A Television Channel Real-Time Detector using Smartphones

Igor Bisio, *Member, IEEE*, Alessandro Delfino, Fabio Lavagetto, and Mario Marchese

Abstract—Recently, people have been interested in sharing what they are watching on TV, allowing the development of Social TV Applications often based on mobile devices. In this context, this paper proposes IRTR (Improved Real-Time TV-channel Recognition): a new method aimed at recognizing in real time (live) what people are watching on TV without any active user interaction. IRTR uses the audio signal of the TV program recorded by smartphones and is performed through two steps: i) fingerprint extraction and ii) TV channel real-time identification. Step i) is based on the computation of the Audio Fingerprint (AF). The AF computation has been taken from the literature and has been improved in terms of power consumption and computation speed to make the smartphone implementation feasible by using an ad hoc cost function aimed at selecting the best set of AF parameters. Step ii) is aimed at deciding the TV channel the user is watching. This step is performed using a likelihood estimation algorithm proposed in this paper. The consumed power, computation and response time, and correct decision rate of IRTR, evaluated through experimental measures, show very satisfying results such as a correct decision rate of about 95%, about $2s$ of computation time, and above 90% power saving with respect to the literature.

Index Terms—TV channel recognition, audience real-time detection, audio fingerprint, smartphone, energy saving

1 Introduction

The rapid growth of social networks and the desire to share personal information and events with other people are the reason why more and more people are getting interested in sharing what they are watching on television. This allows the development of Social TV Applications (STVAs) often based on mobile devices (i.e., Smartphones) because of their pervasiveness and widespread use, as shown in the following. Among the available solutions, the most interesting for our purposes are applications that allow recognizing what a person is watching on TV [1] without active user interaction.

1.1 Future Social TV Applications (STVAs)

Future STVAs are usually based on patented sound-recognition technologies that listen to the audio of the TV, computer or other devices, automatically recognize the broadcast program, and allow sharing it with friends on Twitter© or Facebook©. On the other hand, the Cloud Computing (CC) paradigm, which has grown much in the recent past, represents a perfect operating environment for STVAs and can be used to generate new services and experiences. The CC paradigm is clearly defined in [2]. The future full and efficient use of STVAs is strictly connected to the evolution of Cloud Computing, as also confirmed by projects such as, among others, IntoNow [1], Tunerfish [3], Miso [4], and Yap.Tv [5]. Platforms (e.g. specific operating system) and Infrastructures (e.g. the Internet, databases, servers) are Services for STVAs. The smartphones are also part of the Cloud Platform and Infrastructure and are tools used by STVAs. STVAs are implemented in the Cloud, so they are Software Services for the Cloud. Recognized TV channel data represent a service for Cloud Users: they are shared among cloud users and represent a resource pool to be accessed. Concerning STVAs, even more important than CC has been the growth of mobile devices because of their impressive diffusion and for their increasing data processing capabilities. Many applications have exploited the proliferation of smartphones and mobile operating systems, and used the smartphone in many different ways, e.g., as a terminal, a node of a network, a sensor, in some cases at the same time. [6] proposes a content sharing mechanism based on Delay Tolerant Networks (DTNs). [7] exploits opportunistic communications to facilitate information dissemination in the emerging Mobile Social Networks (MoSoNets). [8] uses the smartphone as a driving aid, giving the driver useful information in order to reduce fuel consumption and trip duration. An important aspect concerning smartphone-based applications is the need to pay attention to power consumption. Applications that are very efficient on plugged-in devices may be unusable on battery-supplied smartphones. Power saving is essential because the energy resource is very limited. An interesting example is the Chameleon application [9], a color-adaptive web browser capable of reducing the power consumption of Organic Light-Emitting Diode (OLED) displays.

1.2 Paper Contribution

In the evolutionary context of mobile devices and cloud computing, our paper introduces a TV channel real-time

• *The authors are with the Department of Telecommunications, Electronic, Electrical, and Naval Engineering, University of Genoa, Genoa 16145, Italy. E-mail: {igor.bisio, alessandro.delfino, fabio.lavagetto, mario.marchese}@unige.it.*

Manuscript received 21 Dec. 2012; revised 13 May 2013; accepted 14 June 2013. Date of publication 19 June 2013; date of current version 26 Nov. 2014. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier 10.1109/TMC.2013.79

detector called IRTR, where the TV audio signal is received and processed by a smartphone that computes its fingerprint and transmits it through a telecommunication network such as the Internet to a server which recognizes the TV channel by comparing the received fingerprint with a set of reference fingerprints through a likelihood-estimation-based process. The proposed IRTR solution, preliminarily introduced in [10], automatizes the process of Audience detection, thus increasing the value for advertisers, and it focuses on power saving both within the algorithm and through the use of the CC architecture, which allows a significant portion of the required tasks to be performed by remote servers, saving important smartphone computation and energy resources while keeping the recognition process reliable and fast. The proposed IRTR solution cannot distinguish between a TV channel and its High Definition (HD) version since their audio components are identical and the IRTR algorithm is based on audio content without employing user cooperation or watermarking techniques. If, on one hand, this could represent a limitation on the proposed solution, on the other hand, the opportunistic nature of the IRTR solution, i.e., no cooperation is required, may help to extend the set of potential users involved in a possible audience-monitoring application, in which it is not necessary to distinguish between a TV channel and its HD version.

The rest of the paper is organized as follows. In Section 2 the motivation behind the IRTR development are outlined. Section 3 describes related scientific works. In Section 4 the IRTR architecture is proposed.The IRTR Audio Fingerprint extraction algorithm and its improvement to better fit mobile devices' needs, in particular in terms of battery life, is described in Section 5. Section 6 focuses on the proposed IRTR channel recognition algorithm. Section 7 shows IRTR performance evaluation.

2 Motivation for IRTR Development

In recent years both academia and industry have been developing solutions and technologies aiming at making the television more interactive. A study by Yahoo and Nielsen [11] proves that 86% of mobile Internet users (and 92% of the people in the 13 - 24 age range) use their mobile devices while watching TV. This presents a compelling opportunity for content providers and advertisers to integrate TV experience and mobile platforms in order to offer new services. Social TeleVision Applications (STVA) are often smartphone applications that synchronize the content shown on the device screen (called "second-screen") with the current TV content (visible on the "main screen") by automatically recognizing what the user is watching on TV [1]. The European project NoTube [12] is focused on connecting TV and Web content through Linked Open Data to enhance the TV experience, for example by recommending programs of interest to the user, including custom advertisements. The TV channel detection is performed by the user in the mentioned cases. IRTR could be easily integrated with these applications by providing the automatic detection of the channel and, also, collecting statistics about the audience of a show without any user interaction and any additional device. Currently, these kind of statistics are collected by specific companies using *ad hoc* meters. This approach limits the number of monitored users because of the meter availability. For example, in the U.K. the Broadcasters' Audience Research Board (BARB) [13], which is the company providing the industry standard television audience measurement service for broadcasters and the advertising industry, monitors a reporting panel of 5100 homes, selected to be representative of 25200000 TV householders. According to [14] there are currently an estimated 12000000 adult consumers in Great Britain who use a smartphone. All these people might become part of the reporting panel by only installing IRTR on their smartphones without any direct intervention nor any additional device installed at home.

3 Related Scientific Work

3.1 Audio Information Retrieval

In the last decade audio recognition in the form of Audio Information Retrieval (AIR) has drawn attention from academia and industry. AIR is the action of recovering/recognizing audio data within a set of pre-recorded audio traces. Most scientific works related to AIR deals with music content identification, which means identifying a song among those available in a pre-computed database. Although many systems are extremely efficient in recognizing a song from a short noisy snippet in a database containing more than 100 000 songs, these systems cannot recognize a TV program in real time for the reasons described in the remainder of this section. Two processes required by AIR, also used in IRTR, are fundamental for audio recognition: audio identification, based on the concept of Audio Fingerprint (AF), and audio recognition strategy, to establish a correspondence between the audio that must be recognized and the set of pre-recorded audio traces available in a database.

3.2 Audio Fingerprint (AF)

Most work investigating the problem of Audio Information Retrieval is based on Audio Fingerprints (AFs) [15]–[19]. An AF is a condensed digital summary, deterministically generated from an audio signal, which can be used to identify an audio sample or quickly locate similar items in an audio database. A fingerprint function F maps an audio object X, composed of a large number of bits, to a fingerprint of only a limited number of bits. An AF is compact because it is significantly smaller, in terms of bits, than the audio it comes from. AFs have gathered attention since they allow the identification of audio independently of its format and without the need of meta-data or watermarking. The AF-extracting methods proposed over the years share the same basic two-step processing of the audio samples: *i*) linear transform and *ii*) feature extraction. Regarding the different choices in the employed linear transformation, many solutions use the Fast Fourier Transform (FFT) [15], [20]. [21] uses Karhunen Loeve (KL) or Singular Value Decomposition (SVD). [22] and [23] use power measures, which can be seen as an integration over the whole frequency domain of a time-to-frequency transform. [24] applies the Modified Discrete Cosine Transform

(MDCT). Wavelet transforms are also employed, e.g. in [25] and [26].

As for feature extraction, [27] and [28] use the Mel-Frequency cepstrum. [29] is based on the Spectral Flatness Measure and [30] on information given by the position of spectral centroids. As already mentioned the smartphone implementation of the application forces us to take into account power issues: each CPU cycle leads to a decrease in battery life.

3.3 Audio Recognition Strategy

Concerning the recognition strategy, all presented related work employs static databases of a limited and somehow defined number of elements. The issue of database retrieval has been addressed in efficient and elegant ways, mainly by means of hash tables [15], which allow to short-list a subset of candidates among which one is chosen, according to some kind of metric (e.g. the Hamming distance in the case of [15]). [31] instead utilizes information about the position both in time and frequency of spectral peaks to build the AF, while the recognition strategy relies on a likelihood estimation based on the number of matching peaks. These retrieval processes are efficient in the case of music identification because the reference database is static and old reference AFs always represent a valid entry (i.e., a user may always listen to an old song). On the other hand, they are not as suitable for real-time TV recognition, since *i*) the reference database should be constantly updated in real time through the fingerprints computed from the signals received by the monitored channels, and *ii*) generally, reference AFs older than a given time (defined empirically in this paper in the next section) are not useful and must be discarded because the recognition algorithm operates with a small delay, with respect to real-time TV channels. Such delay is typically introduced only by the communication network (e.g. the Internet) and by delayed-TV systems such as TiVo [32]. In short, TV channel audio recordings lose their importance for TV channel recognition proportionally to their distance from the current time.

3.4 IRTR Relation With the State of the Art

IRTR aims at implementing a system based on a client-server architecture capable of labeling noisy audio recordings with the name of the TV channel that is watched by a user. Audio recording and feature extraction duties are carried out by a smartphone device. For this purpose IRTR uses Philips' Audio Fingerprinting (AF) algorithm widely used for music information retrieval [15], modified by tuning the basic parameters to decrease the computational load, making it more suitable for smartphone processing platforms. An *ad hoc* cost function has been proposed to select the best set of parameters, as explained in the following. A novel time-shift-invariant likelihood estimation algorithm is proposed to obtain real-time TV channel audio recognition. While music identification is a well investigated topic, to the best of the authors' knowledge, real-time detection of the audio of a TV program is still an unexplored subject for what concerns the state of the art of signal processing.

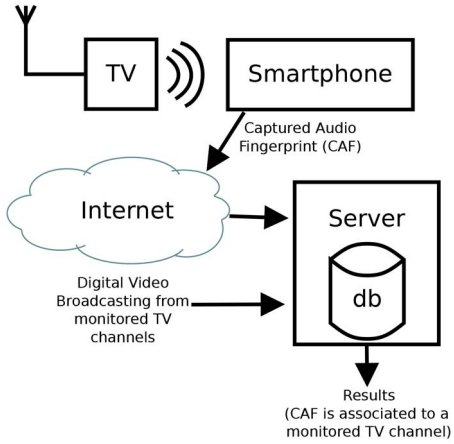

Fig. 1. Overview of the IRTR system architecture.

4 IRTR System Architecture

4.1 General Scheme

The IRTR system architecture is shown in Fig. 1. The IRTR method is based on a client-server model. Clients are installed on smartphones. Android-based smartphones have been used to test IRTR, as described in Section 7, but the IRTR implementation is independent of the specific smartphone technology. The server is implemented on a computer running Linux OS in our case. Client and server are connected through the Internet. This architecture may be extended by adding multiple servers, each dedicated to specific contents, such as Live Radio Broadcast, movies or songs. The fingerprint extracted by the smartphone can be shared, exploiting the advantages of cloud computing, by all these servers. In any case, the extension to multiple servers is left for future investigation. The employed architecture was chosen by following previous similar solutions in the literature, such as [15]. An alternative architecture might avoid using a server-side database by placing most of the effort on the smartphones. Nevertheless the proposed client/server architecture was preferred for the following reasons.

i) The smartphone cannot reasonably constantly monitor TV channels. It would require that a TV receiver be always active on the smartphone, as well as the reference fingerprints extractor. It would cause a relevant consumption of smartphone computational resources and energy. Moreover, considering that the only way to receive TV transmissions with a smartphone (avoiding additional hardware) is by using the mobile connection (2G/3G), the reception of many simultaneous TV streams would imply the saturation of the mobile connection capacity. Additionally, although the number of reference fingerprints in the database is limited (from less than ten fingerprints, as in the case of this paper, to no more than hundreds), it must be continuously stored and updated. These actions require memory, which is particularly precious in smartphones. Eventually, moving the task of monitoring TV channels to the smartphone implies that the action must be performed by all clients. On the other hand, this task, which is identical for all the clients, may be performed only once by the server.

ii) The server could be employed for collecting users' statistics. It is crucial for audience monitoring applications in

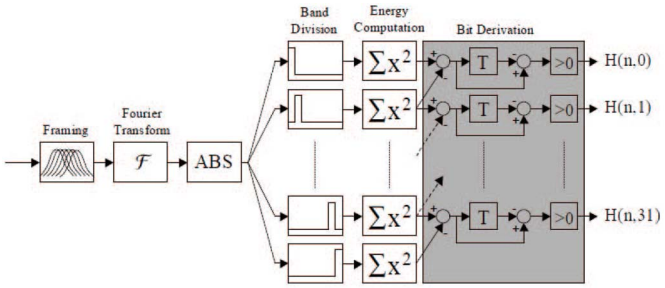

Fig. 2. Block diagram of the audio fingerprint extraction algorithm presented in [15].

which the proposed solution can be employed. The detailed role of smartphone and server within the IRTR architecture is detailed below.

4.2 Smartphone

The smartphone records a short snippet of the TV audio channel using its microphone, it processes it by extracting the AF through the AF extraction algorithm described in Section 5.1 and it sends the AF to the Server for labeling. The fingerprint is a two-dimensional matrix whose number of rows is proportional to the length of the audio from which it is extracted. The fingerprint obtained by the smartphone and sent to the server is called Captured Audio Fingerprint (CAF).

4.3 Server

The server, in order to monitor TV programs in real time, must be capable of receiving Digital Video Broadcasting signals, extracting the audio of each channel to be monitored, computing its fingerprint in real-time, and updating the database with the fingerprints computed from all monitored channels. The fingerprints contained in the database are called Reference Audio Fingerprints (RAF). The server implements the AF-based audio recognition strategy by searching for a correspondence between the CAF sent by the client and one of the RAFs, and possibly labeling the CAF with the name of the channel from which it was extracted. Obviously this last step, which is the aim of audio recognition, may fail or lead to a wrong choice. To avoid the infinite growth of the database the oldest fingerprints must be dropped after a certain time. In our tests, the database contains the RAFs extracted from the last 5 minutes of the audio of each monitored channel and it is updated by using a First In First Out (FIFO) policy, where the oldest fingerprint row is dropped every time a new one is created. The 5-minutes temporal limit is an empirical value that allows the system to also detect slightly delayed transmissions. The IRTR system can face delays, whose order of magnitude ranges from milliseconds to a few seconds, due to network access and transmission technologies (e.g., terrestrial, satellite or cable broadcasters). The reference database stores fingerprints representing the last 5 minutes, with respect to the current time, of each TV channel, as reported at the end of Section 4.3. Considering that the captured fingerprint represents a few seconds of recorded audio, the stored reference fingerprints surely contain the acquired one. Actually, the 5-minutes recordings represent, in practice, a sort of temporal buffer aimed at absorbing possible delays and they are employed to avoid possible declines in performance due to delay to access to the Cloud when the captured fingerprint is sent to the server.

$H(1,1)$	$\cdots$	$H(1, M_{bins}-1)$
$\vdots$	$H(n,m)$	$\vdots$
$H(N_{frames},1)$	$\cdots$	$H(N_{frames}, M_{bins}-1)$

Fig. 3. Audio fingerprint structure.

5 IRTR Audio Fingerprint Computation

5.1 Basic Computation

The fingerprint extraction algorithm used by IRTR is based on the approach introduced by Philips Research [15], reported in the following and shown in Fig. 2. The following steps are performed in order:

a) An audio recording of $L_R[s]$ of duration sampled at a rate of $R_S[Hz]$ is divided into frames of $L_F \simeq 0.37s$, with an overlap factor OL.
b) Each frame is filtered by means of a Hanning window function [33], in order to smooth the signal and to reduce spurious frequency components.
c) The Fast Fourier Transform (FFT) and squared modulus are applied to each frame in order to obtain the energy spectrum of each frame.
d) The spectrum is divided into M_{bins} logarithmically spaced frequency bins and the energy is computed for each bin. The logarithmic spacing is chosen because of the similarity with the Human Auditory System [34].
e) By denoting the energy of band m of frame n by $E(n,m)$, the output of the fingerprint extraction block is defined as in (1). A set of features $H(n,m)$ is computed for every frame n.

$$H(n,m) = \begin{cases} \text{if } E(n,m) - E(n,m+1) - (E(n-1,m) \\ \quad -E(n-1,m+1)) \geq 0 \\ \\ 0 \text{ if } E(n,m) - E(n,m+1) - (E(n-1,m) \\ \quad -E(n-1,m+1)) < 0. \end{cases} \quad (1)$$

Features $H(n,m)$ can assume only binary values: either "0" or "1". The features extracted for each frame compose a row of the fingerprint, thus the fingerprint is a bi-dimensional matrix with $M_{bins}-1$ columns and N_f rows, where N_f is the overall number of considered frames. The content of this fingerprint is shown in Fig. 3. The m-th column is the contribution of frequency m for each frame. The n-th row is the contribution of frame n for each frequency bin, from 1 to $M_{bins}-1$, and it may be defined as the fingerprint of frame n. Since the CAF and the RAFs refer to signals of different duration, the number of frames in CAF and RAFis obviously different. In the following N_f^C and N_f^R denote, respectively, the number of CAF and RAF frames. The parameter values employed in [15] are listed in Table 1.

TABLE 1
Fingerprint Parameters

Recording Length (L_R)	10 [s] (CAF) and 5 [min] (RAFs)
Overlap (OL)	31/32
Sample Rate (R_S)	44100 [Hz]
Frame Length (L_F)	$\sim$ 0.37 [s]
Number of Sub-Bands (M_{bins})	33

5.2 Modifications Aimed at Reducing the Computational Load and Consequent Energy Consumption

An important part of the IRTR method is the improvement of the AF extraction process, summarized in 5.1, in terms of computational load. Our aim is to make the extraction process as fast as possible in order to reduce its computational load, keeping the performances of the system high in terms of correct detections. The aim is to have a trade-off between two attributes: detection accuracy and computational time. The fingerprint extraction algorithm can be modified by setting the following parameters: duration of the recording, L_R, overlapping factor between two consecutive frames, OL, and sample rate of the audio acquisition, R_S. Two different performance metrics are of particular interest: detection rate $R_D(L_R, OL, R_S)$ and fingerprint extraction duration $\tau(L_R, OL, R_S)$, which represent the detection accuracy and computational time, respectively. Both can be considered as a function of the AF algorithm parameters. The detection rate is defined as the number of correct decisions divided by the total number of trials. The fingerprint extraction time is the time required by the client (i.e. the smartphone) to compute the fingerprint. Our aim is to reduce the computational time while limiting the decrease in detection accuracy by acting on L_R, OL and R_S. We define two different cost functions, each one evaluating a performance metric:

$$v_1 = \frac{1}{R_D(L_{R,i}, OL_j, R_{S,k})} - 1$$
$$v_2 = \frac{\tau(L_{R,i}, OL_j, R_{S,k})}{\max \tau(L_{R,i}, OL_j, R_{S,k})}. \quad (2)$$

A property of functions v_1 and v_2 is that they equal to 0 when the ideal metric value is reached, $R_D = 1$ and $\tau = 0$ respectively. It was decided to merge the two cost functions by summing them, therefore the final cost function is:

$$C(L_{R,i}, OL_j, R_{S,k}) = v_1 + v_2. \quad (3)$$

Given its definition, shown in (3), minimizing the cost function $C(\cdot)$ is equivalent to finding a trade-off between detection accuracy and computational time, since $C(\cdot)$ decreases when $R_D(\cdot)$ increases and when $\tau(\cdot)$ decreases.

Weights ω_1 and ω_2 such that $\omega_1 + \omega_2 = 1$ with $0 \leq \omega_1 \leq 1$ and $0 \leq \omega_2 \leq 1$ may be used as multiplicative factors for v_1 and v_2, respectively, in (3), to give more or less importance to one of the metrics depending on the performance aims. Weights investigation is left to further research. Cost function (3) ensures that the ideal cost $C(L_{R,i}, OL_j, R_{S,k}) = 0$ is obtained with a perfect detection rate, $R_D(L_{R,i}, OL_j, R_{S,k}) = 1$, and a null computational time $\tau(L_{R,i}, OL_j, R_{S,k}) = 0$, as evident in 4.

$$C(L_{R,i}, OL_j, R_{S,k}) = \left(\frac{1}{R_D(L_{R,i}, OL_j, R_{S,k})} - 1\right) + \\ + \frac{\tau(L_{R,i}, OL_j, R_{S,k})}{\max \tau(L_{R,i}, OL_j, R_{S,k})}. \quad (4)$$

Since functions $R_D(L_{R,i}, OL_j, R_{S,k})$ and $\tau(L_{R,i}, OL_j, R_{S,k})$ are not known in closed form, they can only be measured. Optimization methods such as Multi-Objective Optimization (MOP) [35] can not be applied. The fingerprint extraction parameters are chosen through a brute-force minimization of the cost function, since the dimensionality of the problem is small enough to permit such an approach. The numerical results of the minimization process are reported in Section 7.1. The chosen parameters are not globally optimal but only the best among all the evaluated ones. The obtained values should be reasonably close to the optimal ones but formal optimality cannot be proven. The estimation of a closed form for functions $R_D(L_{R,i}, OL_j, R_{S,k})$ and $\tau(L_{R,i}, OL_j, R_{S,k})$ will be a future development of this work. The closed form shall allow using non-linear programming methods and finding the optimal set of parameters on a continuous scale.

6 IRTR TV Channel Recognition

The recognition problem consists of deciding which channel the received fingerprint belongs to. The problem of Live TV detection does not allow building Look Up Tables as in the Music Identification problem because, as previously stated, the database contents change continuously (about every 100 ms with the best parameters setting) and old AFs provide limited contribution. A suitable algorithm must therefore be developed. Once the server receives a new fingerprint (CAF) it must compare it to the fingerprints stored in its database (RAFs) to find a possible correspondence. The first required step is to provide a measure of the level of similarity between audio fingerprints, in order to estimate the likelihood that the two fingerprints belong to the same audio content. It is important to take into account the lack of synchronization between the audio stream that the server uses to build the AF database and the audio captured by the smartphone. It is in practice very difficult to calculate the transmission delay and to predict the exact time difference between the server-database and the client-smartphone. Such asynchrony calls for a suitable likelihood estimation algorithm robust to time shifting. Two likelihood estimation algorithms are presented in Section 6.1 and compared in Section 7.5. The first one called "Direct Method" is simply based on the computation of the Hamming Distance between fingerprints and it is described in Section 6.1.1. The second one, called "Reduced Complexity (RC) method", is introduced in this paper in Section 6.1.2 and is aimed at reducing the complexity of the Direct method. An audience-based scheme is proposed in 6.1.3 to implement a quick search in the RAF database to find a correspondence. In content retrieval architectures detection errors weigh differently, whether they are false alarms or missed detection errors. When a user is watching a TV channel a missed detection occurs if the system states that the user is not watching any of the monitored channels, while a false alarm occurs when the channel recognized by the system is not

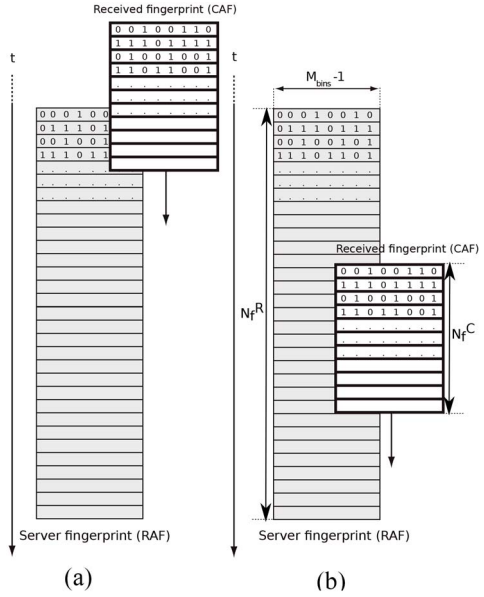

Fig. 4. $H_C(n, m)$ shifted over the overall length of $H_R(n, m)$ with partial (a) and total (b) overlap.

the one the user is actually watching. In the false alarm case the error is twice as costly, since, if the user is watching channel A, taking the wrong decision implies *i*) rejecting the hypothesis of channel A and *ii*) accepting the wrong hypothesis of channel B. Furthermore, for the goal of IRTR having a missed detection is more acceptable than recognizing a wrong channel. A method is used to set a threshold θ on the likelihood in order to decide whether the received fingerprint is taken from a channel or not. The overall decision method used by IRTR is presented in Section 6.1.4. The threshold θ is computed by considering that an important goal of the IRTR system is to keep the false alarm probability P_{FA}, formally defined in Section 6.2, as low as possible. The threshold θ applied in the likelihood estimation has been set in order to keep the false alarm probability below $3 \cdot 10^{-3}$ ($P_{FA} < 3 \cdot 10^{-3}$) and it has been computed by using a data set of 300 trials employed to estimate the Probability Density Function (PDF) explicitly considered in Section 6.2.

6.1 Likelihood Estimation

6.1.1 Likelihood Estimation Algorithm: Direct Method

Since the fingerprints are binary matrices, a reasonable metric of the similarity between fingerprints is the Hamming distance between them. $H_R(n, m)$ and $H_C(n, m)$ are the reference and the captured fingerprints respectively. It is worth remembering that $H_R(n, m)$ and $H_C(n, m)$ have different dimensions, as they both have $M_{bins} - 1$ columns, but the number of $H_R(n, m)$ rows N_f^R is much larger than N_f^C, the number of rows of $H_C(n, m)$. Therefore, the likelihood estimation algorithm must search for the portion of $H_R(n, m)$ most similar to $H_C(n, m)$. This is achieved by shifting the captured fingerprint $H_C(n, m)$ over the overall length of $H_R(n, m)$ (or vice-versa if mathematically simpler) as shown in Fig. 4(a) and (b), and, for each shift value k, computing the Hamming distance $H_d(k)$ as in (5). To include in the similarity evaluation also partial overlaps of $H_C(n, m)$ and $H_R(n, m)$, shown in Fig. 4(a), the fingerprint $H_R(n, m)$ is extended $N_f^C - 1$ rows before and $N_f^C - 1$ rows after the original $H_R(n, m)$ by using zero-padding so to have an overall length of $N_f^R + 2N_f^C - 2$.

$$H_d(k) = \sum_{n=k}^{k+N_f^C} \sum_{m=1}^{M_{bins}-1} H_R(n-k, m) \oplus H_C(n, m), \quad \forall k \in [-N_f^C + 2, N_f^R - 1]. \tag{5}$$

The Hamming distance $H_d(k)$ is the number of positions at which the corresponding symbols in the two overlapped matrices, H_C and extended H_R, for each shift k, are different. Having a Hamming distance value $H_d(k)$ for each shift, the Hamming distance N_{bit}^D between $H_R(n, m)$ and $H_C(n, m)$ is defined as the minimum $H_d(k)$ over the shift k as in (6).

$$N_{bit}^D = \min_k \left(H_d(k)\right), \forall k \in [-N_f^C + 2, N_f^R - 1]. \tag{6}$$

The complexity of (5) is $\mathcal{O}\left(N_f^C(M_{bins} - 1)\right)$ for each time shift k, thus the total complexity of the formula is $\mathcal{O}\left(N_f^C(M_{bins} - 1)(N_f^C + N_f^R - 2)\right) = \mathcal{O}\left((M_{bins} - 1)\left({N_f^C}^2 + (N_f^R - 2)N_f^C\right)\right)$.

6.1.2 Likelihood Estimation Algorithm: Reduced Complexity (RC) Method

The real-time TV detection requires a quick server response and, therefore, a quick likelihood computation. To reduce the complexity of the computations, the metric L we measure is a modification of the Hamming distance. Let $L(k)$ be the difference between the number of positions where the bits of the two fingerprints are equal $N_{bit}^E(k)$ and the number of the positions where the two fingerprints differ $H_d(k)$. Given $N_{bit}^{TOT} = N_f^C \cdot (M_b ins - 1)$ the dimension, in bits, of the smaller fingerprint:

$$L(k) = |N_{bit}^E(k) - H_d(k)|. \tag{7}$$

Having a $L(k)$ value for each shift we proceed similarly to the previous case by using (6):

$$\begin{aligned} N_{bit}^D &= \min_k \left(H_d(k)\right), \forall k \in [-N_f^C + 2, N_f^R - 1] \\ N_{bit}^E &= \max_k \left(N_{bit}^E(k)\right), \forall k \in [-N_f^C + 2, N_f^R - 1] \end{aligned} \tag{8}$$

therefore,

$$\begin{aligned} L &= \max_k |N_{bit}^E(k) - H_d(k)| = |N_{bit}^E - N_{bit}^D| = \\ &= |N_{bit}^{TOT} - 2N_{bit}^D| = |2N_{bit}^E - N_{bit}^{TOT}|, \\ &\forall k \in [-N_f^C + 2, N_f^R - 1]. \end{aligned} \tag{9}$$

Such metric is 0 when $N_{bit}^E = N_{bit}^D$, i.e. when the Hamming distance between the two fingerprints is $\frac{N_{bit}^{TOT}}{2}$. Both when $N_{bit}^D = 0$ and when $N_{bit}^D = N_{bit}^{TOT}$, $L = N_{bit}^{TOT}$. This is correct, in our view, because a fingerprint and its negated convey, in fact, the same information. Adding the $|\cdot|$ operator is a way to consider this within the likelihood estimation algorithm. The likelihood value L can be normalized between 0 and 1 as in (10).

$$L_n = L/N_{bit}^{TOT}. \tag{10}$$

The RC likelihood estimation algorithm exploits the fact that the metric L can be computed very efficiently through the cross-correlation of two sequences by using the Fast Fourier Transform (FFT). The first step is defining $H'(n,m)$ as the fingerprint $H(n,m)$ where the "0"s are replaced with "−1"s, as shown in (11).

$$H'(n,m) = 2 \cdot H(n,m) - 1, \forall n, m. \tag{11}$$

The product between an element of the converted matrix $H'_R(n,m)$ and another element of the converted matrix $H'_C(n,m)$ is 1 if they are equal and −1 if they are different. This allows to compute the metric value $L(k)$ for each shift k by multiplying, element by element, the shifted matrix $H'_R(n-k,m)$ and the matrix $H'_C(n,m)$ and then adding the results of each element product:

$$L = \max_k (L(k)) = \max_k \left(\left| \sum_{m=1}^{M_{bins}-1} \sum_{n=-N_f^C+2}^{N_f^R+2N_f^C-1} H'_R(n-k,m) \cdot H'_C(n,m) \right| \right), \quad , k \in [-N_f^C+2, N_f^R-1] \tag{12}$$

The summation over n in (12) is the one-dimensional cross-correlation $R_{H'_R H'_C}(k,m)$ between each column of $H'_R(n,m)$ and the corresponding $H'_C(n,m)$ column. In general this operation equals the convolution between the columns of $H'_R(n,m)$ and the corresponding columns of $H'^*_C(-n,m)$, the complex conjugate of $H'_C(n,m)$. In our case, being $H'_C(n,m)$ real for all possible values of m and n, $H'^*_C(n,m) \equiv H'_C(n,m)$, (14) is true.

$$\begin{aligned} R_{H'_R H'_C}(k,m) &= \sum_{n=-N_f^C+2}^{N_f^R+2N_f^C-1} H'_R(n-k,m) \cdot H'_C(n,m) \\ &= H'_R(n,m) * H'^*_C(-n,m) \\ &= H'_R(n,m) * H'_C(-n,m). \end{aligned} \tag{13}$$

Merging (12) and (14) we can re-write (12) as follows:

$$L = \max_k \left(\left| \sum_{m=1}^{M_{bins}-1} H'_R(n,m) * H'_C(-n,m) \right| \right), \quad , k \in [-N_f^C+2, N_f^R-1] \tag{14}$$

Since cross-correlation $R_{H'_R H'_C}$ is a convolution, it can be computed efficiently by using Fast Fourier Transform (FFT) algorithms. The theoretical computational complexity of the FFT for a sequence of length N is $\mathcal{O}\left(\frac{N}{2}\log_2(N)\right)$, and in practice, since FFT algorithms process sequences whose lengths are powers of two, $\mathcal{O}\left(\frac{2^{\lceil \log_2 N \rceil}}{2}\lceil \log_2(N) \rceil\right)$. The one-dimensional convolution of two sequences of length N is calculated in the frequency domain by multiplying element by element the Fourier transform of both sequences and then computing the Inverse Fourier transform of the product sequence. The complexity of this operation is due to 3 FFTs performed on sequences of length $2^{\lceil \log_2 N \rceil}$ and to $2^{\lceil \log_2 N \rceil}$ multiplications. Therefore the overall complexity is $\mathcal{O}\left(\frac{3}{2} \cdot 2^{\lceil \log_2 N \rceil} \lceil \log_2 N \rceil + 2^{\lceil \log_2 N \rceil}\right)$. In our case, since the shorter sequence is zero-padded up until the length of the longer one and the one-dimensional cross-correlation has to be computed for each of the $(M_{bins}-1)$ columns of $H'_R(n,m)$ and $H'_C(n,m)$, the overall complexity is $\mathcal{O}\left((M_{bins}-1) \cdot \left(\frac{3}{2} \cdot 2^{\lceil \log_2 N_f^R \rceil} \lceil \log_2 N_f^R \rceil + 2^{\lceil \log_2 N_f^R \rceil}\right)\right)$. This value must be compared with $\mathcal{O}\left((M_{bins}-1)\left(N_f^{C^2} + (N_f^R - 2)N_f^C\right)\right)$, which is the complexity of the Direct method. A full numerical comparison is reported in Section 7.5 but it is important to observe from the start that RC method complexity does not depend on the length of the CAF fingerprint.

TABLE 2
Italian Share of Audience Statistics During the Daily Time Slot Lasting from 20.30 to 22.30, October 2011, [36]

Channel Name	Share of TV watchers
Rai 1	18.55%
Canale 5	16.39%
Rai 3	9.29%
Italia 1	8.88%
Rai 2	8.67%
Rete 4	6.39%
La 7	5.42%
Total	73.59%

6.1.3 Search Algorithm

The idea concerning the database search is based on the observation that television audience ratings are far from being equally distributed. For example, what currently happens in Italy is that, though more than 150 channels are broadcast, over 70% of the audience is gathered around no more than 7 channels. Table 2, for example, shows the average share of TV watchers for the 7 most watched Italian TV networks during October 2011 [36]. Actually, the values are not time-invariant and are referred to a particular time slot (from 20.30 to 22.30) and to a specific month. Nevertheless, using audience statistics allows to sort the channels to be matched in the database from the most- to the least-watched. Such searching policy should reduce the time required to find the correct channel, on average, therefore speeding up the recognition process.

6.1.4 Decision Algorithm

The overall decision algorithm applied by IRTR is described by the flowchart in Fig. 5.

When a query fingerprint is received, the server compares the received fingerprint with the first fingerprint of the database (the most watched channel according to the available statistics). If the likelihood score L is higher than the threshold value θ, whose computation is explained in the next section, the server decides that the user is watching that channel, otherwise it compares the received fingerprint with the following fingerprint stored in the database. If, after comparing the received fingerprint with all the fingerprints of the N monitored channels from the most- to the least-watched channel, the system has not found a match, it concludes that the user is not watching any of the

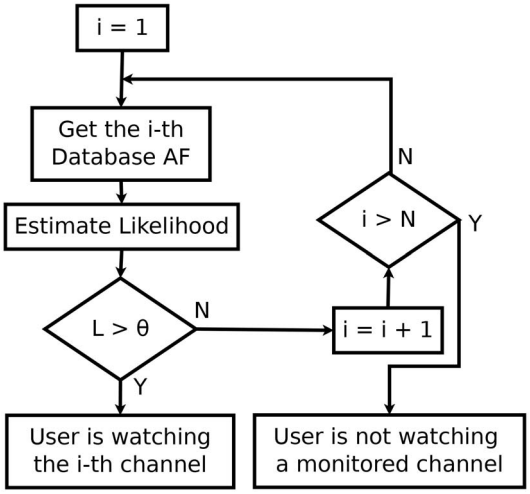

Fig. 5. IRTR decision algorithm flowchart.

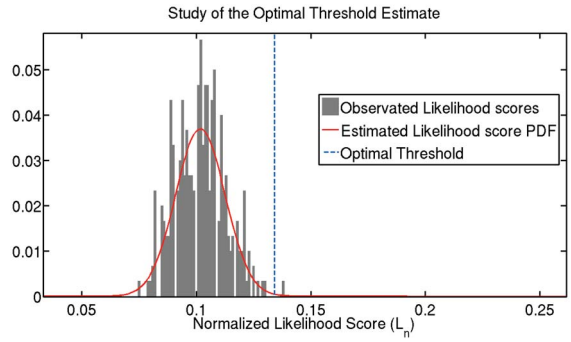

Fig. 6. Probability density function of normalized likelihood scores L_n, as in (10) referred to H_0 hypothesis.

monitored channels. As described in Section 6.1.3, the availability of audience statistics and, consequently, of a priori probabilities of audience values may improve the computational speed. Table 2 is only an example referred to the Italian case used to produce the results shown in this paper but it may be helpful to understand: without having any information on the audio captured by the smartphone of the user, the probability (from Table 2) that the user is watching one of the seven most-watched Italian Television channels is over 73%. This means that by sorting the channels fingerprints in the database from most- to least-watched, more than 73 times out of 100, the search will stop within 7 cycles of the algorithm, avoiding useless operations and saving time retrieving a quicker response. Although the estimation of these a priori probabilities is out of the scope of this paper, it is evident that the system can collect statistics on the channels watched by the users and therefore estimate the a priori probabilities.

6.2 Optimal Threshold

Within a system monitoring N TV channels, the problem of deciding which channel the user is watching is composed of N sub-problems: for each i-th ($i \in [1, N]$) stage of the decision process the system must decide between hypothesis H_1 (the user is watching the i-th channel) or hypothesis H_0 (the user is not watching the i-th channel). As can be seen from the flow chart in Fig. 5, the threshold value θ plays a crucial role by directly affecting the detection rate. The system can fail by assessing the user is watching the i-th channel (H_1) while the user is not watching it (H_0) so causing a False Alarm, or, vice-versa, it can fail by assessing that the user is not watching the i-th channel (H_0), while the truth is that the user is watching it (H_1), causing a Missed Detection. As said, Missed Detections are much more acceptable than False Positives. The Maximum Likelihood criterion cannot be applied because, even if statistics on the likelihood values L referred to H_0 class can be inferred, we do not have general information about the L values of the H_1 class since they are strongly dependent on the audio Signal-to-Noise Ratio (SNR). As anticipated in Section 6.1, the reason for the setting of θ is to maximize the detection probability while keeping the false alarm probability P_{FA} below an acceptable value ($3 \cdot 10^{-3}$). We can define the false alarm probability as the probability of deciding H_1 while the ground truth is H_0, $P(H_1|H_0)$. The Probability Density Function (PDF) of the likelihood value L given H_0 $p_L(L|H_0)$, assumed normally distributed, has been estimated through a finite number of observations of H_0 class realizations. To do so the likelihood values L obtained when the client is listening to the audio of a non-monitored channel have been collected. Obviously this corresponds to the event $(H_1|H_0)$ because the CAF is related to a non-monitored unknown channel. Being L_i the L value measured during the i-th observation and $N_o = 300$ the total number of observations the sample mean $\mu = \bar{L} = \frac{1}{N_o}\sum_{i=1}^{N_o} L_i$ and the sample variance $\sigma^2 = s^2 = \frac{1}{N_o}\sum_{i=1}^{N_o}(L_i - \mu)^2$ have been used to obtain the estimated normally distributed PDF (15). The assumption that the PDF is normally distributed is qualitatively justified in Fig. 6, and is tested numerically in Section 6.2.1, where the results of the Lilliefors Test for Normality are reported.

$$P_{FA} = P(H_1|H_0) = P\{L > \theta|H_0\} = \\ = \int_{\theta}^{+\infty} p_L(x|H_0)\mathrm{d}x = \int_{\theta}^{+\infty} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}\,\mathrm{d}x. \quad (15)$$

The threshold θ has been set to assure that $P_{FA} < 3 \cdot 10^{-3}$ by applying the "3σ Rule". In general, this rule is employed to obtain an approximate probability estimate of a quantity, given its standard deviation, if the reference population is assumed normal. This criterion was chosen because it is widely used in statistics. The "3σ Rule" states that, for a normal distribution, nearly all values lie within a distance of 3 standard deviations from the mean. In mathematical terms, the the area of the two tails starting from 3σ from the mean of a normal distribution is $3 \cdot 10^{-3}$. Considering a single Gaussian tail for the sake of simplicity:

$$\int_{\mu+3\sigma}^{+\infty} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}\,\mathrm{d}x = 1.5 \cdot 10^{-3}. \quad (16)$$

Therefore the threshold value has been set to $\theta = \mu + 3\sigma$.

6.2.1 Normality Test

We previously assumed the Probability Density Function of the H_0 class to be normally distributed. The proof is given in this section by performing the Lilliefors Test for Normality [37]. For the sake of briefness, the mathematical detail of the test is not reported.

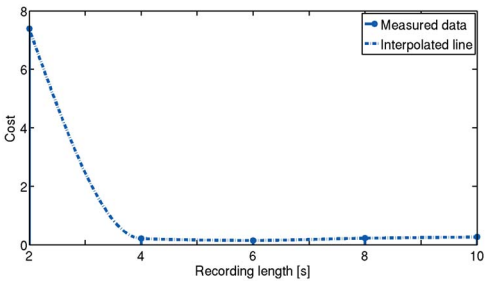

Fig. 7. Cost function values with variable L_R ($OL = \frac{24}{32}$, $R_S = 22050$ [Hz]).

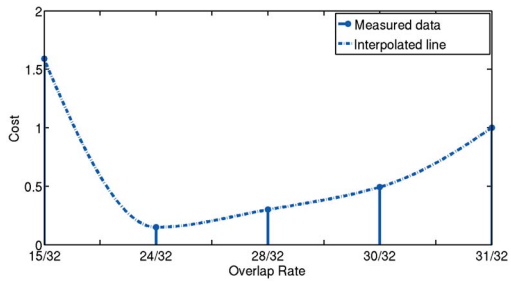

Fig. 8. Cost function values with variable OL ($L_R = 6$ [s], $R_S = 22050$ [Hz]).

The Lilliefors Test is an adaptation of the Komolgorov-Smirnov Test and it is used to test the hypothesis that data come from a normally distributed population, i.e., that there is no difference between the observed distribution and a normal distribution with mean and variance equal to the sample mean and sample variance of the population. The alternative hypothesis is that the population is not normally distributed. The normality test is based on the concept of statistical significance, denoted by α, which is the rejection probability of the normality hypothesis when it is true. Defining, coherently with the literature in the field, the amount of evidence against the normality hypothesis as p-value then it is possible to state that data come from a normally distributed population when the p-value is smaller than the significance level α, with a confidence level higher than $1-\alpha$. Though the selection of the significance level is arbitrary, in our paper a significance level of $\alpha = 0.05$ was chosen, as conventionally done for most applications. The Lilliefors Test was run - using an *ad hoc* MatLab tool - on the observation vector from which the histogram in Fig. 6 was produced. The result is a p-value equal to 0.0437, strictly lower than reference α. This result allows assuming that the PDF of the H_0 class is normally distributed with a confidence level of at least 95%.

7 Performance Investigation

An algorithm executed on mobile devices such as smartphones has to deal with battery lifetime. It is therefore crucial to limit the amount of CPU operations. For these reasons, in Section 5.2, we described the method we applied to find an improved set of parameters aimed at reducing the energy consumption for the fingerprint computation algorithm, which is the main task performed by the smartphone from the computational load viewpoint. The results regarding the parameter configuration, as well as the selection of the best tested parameter configuration, are presented in Section 7.1. Section 7.2 reports the comparison, concerning the overall system performance, between IRTR when the best parameter configuration analyzed in Section 7.1 is employed and IRTR with the parameter configuration used in the Philips approach [15]. The performance is evaluated in terms of:

- Detection rate R_D: number of correct decisions divided by the number of trials;
- AF computation time τ: time the client needs to compute the fingerprint;
- Consumed power and its integral Consumed energy by the Client, measured for one trial (from the channel detection triggering to the server response);
- Response time: time elapsing from the channel detection triggering to the server response.

In Section 7.4 the power consumption of IRTR implementation is compared with the commercial application IntoNow [1]. The client employed for all the tests described in this paper, on which te application implementing the IRTR algorithm and the IntoNow application were run, was a Samsung Galaxy S Android smartphone. The likelihood estimation algorithm, implemented in the server, has been tested in Section 7.5: the proposed RC method for likelihood estimation is compared with the Direct method. The results highlight the advantages in terms of computational complexity and, consequently, of computational time.

7.1 Fingerprint Parameters Configuration

Section 5.2 describes the changes to the fingerprint extraction algorithm by setting the following parameters: L_R, duration of the recording; OL, overlap of two consecutive frames; R_S, sample rate of the audio acquisition. This section presents the results of this process by showing the value of the cost function C obtained by changing the AF parameter values (L_R, OL, R_S).

With the aim of finding a suitable parameters combination, the cost function C was minimized by applying a brute force procedure. All possible combinations of the tested parameters, defined below, were employed and the values assumed by the cost function were evaluated. Each parameter combination was evaluated by 20 trials (i.e., each point reported in the graphs is the average over 20 trials). So, since 75 combinations were evaluated, as detailed below, 1500 trials have been completed.

In more detail, the recording length L_R was varied from 2 [s] to 10 [s] (with a 2 [s] step); the following overlap rates OL were considered: $\frac{15}{32}$, $\frac{24}{32}$, $\frac{28}{32}$, $\frac{30}{32}$ and $\frac{31}{32}$; three possible values of sample rate R_S were applied: 11025 [Hz], 22050 [Hz] and 44100 [Hz]. Being TV audio the reference signal of this paper, these sample rates have been chosen because they are usually employed by algorithms for music identification aimed at tracking all possible human audible sounds and extending the applicability of the solutions beyond typical *speech*-based applications working at 8000 [Hz]. As shown below, the best trade-off between accuracy and complexity is reached with $R_S = 22050$ [Hz]. The employment

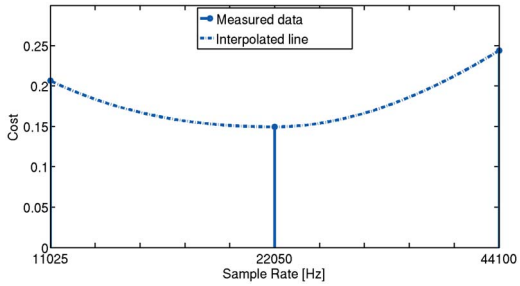

Fig. 9. Cost function values with variable R_S ($L_R = 6$ [s], $OL = \frac{24}{32}$).

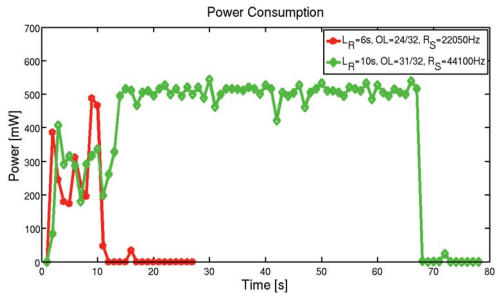

Fig. 10. Power consumption of the IRTR algorithm with the following parameter configurations: ($L_R = 6$s, $OL = \frac{24}{32}$, $R_S = 22050$Hz) and ($L_R = 10$s, $OL = \frac{31}{32}$ and $R_S = 44100$Hz).

of different sample rates implies higher values of the cost function.

The most meaningful results (extracted from all tested combinations) of this procedure may be seen in Figs. 7 to 9. Each figure reports the behavior of the cost function C obtained by varying a single parameter (the recording length in Fig. 7, the overlap rate in Fig. 8, and the sample rate in Fig. 9) while keeping constant, at the value assuring the minimum C, the others. These results show that the parameter combination that minimizes the cost function is $L_R = 6$s, $OL = \frac{24}{32}$ and $R_S = 22050$Hz.

7.2 Comparison With the Parameter Configuration Used in the Original Philips Approach

Table 3 compares the values of the cost function C, of the detection rate R_D, of the computation time τ, and of the consumed energy for each trial by using the IRTR parameter configuration assuring the best performance ($L_R = 6$s, $OL = \frac{24}{32}$, $R_S = 22050$Hz) among the tested ones and by using the combination $L_R = 10$s, $OL = \frac{31}{32}$ and $R_S = 44100$Hz adopted by the traditional Philips AF approach. Table 3 also shows the percentage gain provided by IRTR with respect to the configuration used by the Philips approach, which because of the longer recording length, the almost complete overlap, and the higher sample rate is very accurate and assures a detection rate of about 100%, although, in this case, the Samsung Galaxy S Android smartphone requires more than 1 minute to complete the fingerprint computation and more than 30000 mJ for each trial. Our best configuration guarantees a detection rate of about 95% but requires less than 2.5s to compute the fingerprint, and less than 3000 mJ for each trial. This result meets real-time requirements as well as those imposed by smartphone platforms and is very satisfying from an operative viewpoint. The improvement in percentage with respect to the original Philips approach is $\sim 96.3\%$ concerning time and $\sim 91\%$ concerning energy. Fig. 10 shows the instant power consumption during a trial for the two considered parameter configurations, with the area under the curve being the energy consumed, reported in Table 3. The tool we used to evaluate power consumption is PowerTutor, an application for Android phones developed jointly by the University of Michigan and Google, which displays the power consumed by major system components such as the CPU, network interfaces, the display and the GPS receiver for every running application. [38] shows the reliability of PowerTutor: it provides measures, for 10-second intervals, with an average error of 0.8% and a maximum error of 2.5% with respect to the measurements taken by a hardware meter.

TABLE 3
Comparison Between IRTR Applying Original ($L_R = 10$s, $OL = \frac{31}{32}$, $R_S = 44100$Hz) and Improved ($L_R = 6$s, $OL = \frac{24}{32}$, $R_S = 22050$Hz) Audio Fingerprint Parameters

	Original settings	Improved settings	Gain
R_D	$\sim 100\%$	$\sim 95\%$	-5%
τ	~ 62s	~ 2.3s	$+96.3\%$
C	~ 1	~ 0.0986	$+90.1\%$
Energy consumed for each trial	30681 mJ	2756 mJ	$+91.0\%$

7.3 IRTR Performance with Different Signal-to-Noise Ratios

The accuracy of the IRTR system depends on the Signal-to-Noise Ratio (SNR). IRTR has been tested by disturbing the TV audio with white noise, which is easily reproducible and leads to non-ambiguous easily repeatable test conditions. White noise represents a worst-case scenario because it is distributed over all frequencies while, everyday background noise (e.g., people talking, ringing phones) is concentrated within given and limited frequency ranges. SNR is computed by using an IEC 651 standard-compliant sound level meter that measures the value of $1+SNR$. The adopted procedure measures the signal-plus-noise level $S + N$ and, after switching off the audio source, the noise level N. In the described conditions the obtained results, in terms of detection rate R_D, are reported in Table 4. Reported values show that, if the noise level is equivalent to the signal level, the IRTR performance is still satisfactory.

TABLE 4
IRTR Accuracy Performance with Different Signal to Noise Ratios

SNR [dB]	-5	0	3	6	10
R_D (%)	30	95	100	100	100

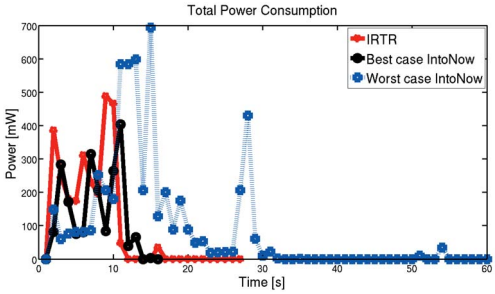

Fig. 11. Total power consumption of IRTR and IntoNow algorithms.

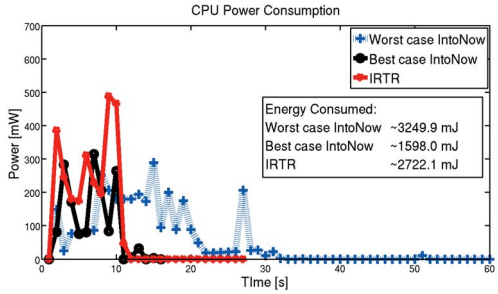

Fig. 12. CPU power consumption of IRTR and IntoNow algorithms.

7.4 Power Consumption Comparison with Existing IntoNow Platform

IntoNow [1] is a successful consumer tech company that offers a solution (based on the patented SoundPrint platform) providing users with the ability to quickly recognize TV contents. The performance of the IRTR and IntoNow solutions are compared in this section, as far as power consumption is concerned, by using PowerTutor, as in the previous case. It is in practice very difficult to fairly compare IntoNow and IRTR in terms of recognition accuracy because their databases are different: IRTR monitors Italian channels while IntoNow operates on U.S. channels so the comparison is limited to power consumption in this paper. The IntoNow algorithm is not known but some of its features can be inferred by the trials' results. The IntoNow platform response time is not fixed because, as far as we can understand, it does not send all the data at the end of the processing waiting for the response, but it implements a progressive sending mechanism. It starts sending data during audio acquisition and it stops when a response is received; if no response is received within 20 s it concludes that it is not able to detect the channel. We have tested the IntoNow solution in two scenarios: the Best Case (BC) and the Worst Case (WC). BC occurs when IntoNow immediately detects the channel, WC when IntoNow does not recognize the channel. Concerning IRTR, WC and BC lead to the same energy consumption because IRTR does not implement any progressive sending mechanism. The implementation of this type of mechanism can be a future development of IRTR aimed at further energy saving. Fig. 11 shows the power consumption over time of IRTR, obviously using the best parameter configuration ($L_R = 6$s, $OL = \frac{24}{32}$, $R_S = 22050$Hz) and IntoNow in the Best and Worst Case scenarios, defined above. Table 5 summarizes Fig. 11 by explicitly showing the response time and consumed energy of the solutions compared in Fig. 11. The IRTR performance is close to the Best Case of IntoNow in terms of energy consumption and response time. A possible further improvement for IRTR may be the implementation of the fingerprint progressive sending mechanism.

Because of the difference between the IRTR and IntoNow implementations it is interesting to evaluate separately the energy consumed by the CPU and by the network interface. The network interface we need to monitor in our test is the WiFi card since the smartphones are connected to the Internet through the WiFi network of the laboratory. Figs. 12 and 13 show the power consumption profile of the CPU and the WiFi interface, respectively. Both figures contain the measured power in [mW] over time, as well as the value of the consumed energy, for IRTR, IntoNow BC, and IntoNow WC. Concerning the IRTR implementation, 98.77% of the energy is consumed by the smartphone's CPU and only 1.23% by the Wifi interface. IntoNow transmits more data and 77.20% of the consumed energy is spent by the CPU and 22.80% by the Wifi interface, in the BC, and 59.37% by the CPU and 40.63% by the Wifi card, in the WC. The IRTR implementation does not involve the network infrastructure as much, thus avoiding overloads and requiring less bandwidth, but its CPU consumption curve has a peak at the fingerprint computation. This result may be improved since our implementation is not yet engineered, i.e., the code is not optimized, since at the moment it is a product of academic research. We do not know what operation corresponds to the IntoNow CPU peaks since the algorithm is unknown. Moreover, it is reasonable to assume that the IntoNow code is fully engineered because it is a successful commercial application.

TABLE 5
Comparison Between IRTR and IntoNow

Metrics	IRTR	IntoNow WC	IntoNow BC
Response Time	$\sim 12s$	$\sim 20s$	$\sim 11s$
% of IntoNow WC	$\sim 60\%$	100%	$\sim 55\%$
Energy Consumed	$\sim 2756mJ$	$\sim 5474mJ$	$\sim 2070mJ$
% of IntoNow WC	$\sim 50.35\%$	100%	$\sim 37.82\%$

7.5 Likelihood Computation Algorithm Evaluation

In this section the performance of the Likelihood Estimation algorithm introduced in Section 6 is evaluated. As previously mentioned in Section 6.1.1 the computational complexity of the Direct method is $\mathcal{O}\left((M_{bins}-1)\left(N_f^{C^2}+(N_f^R-2)N_f^C\right)\right)$, while the complexity

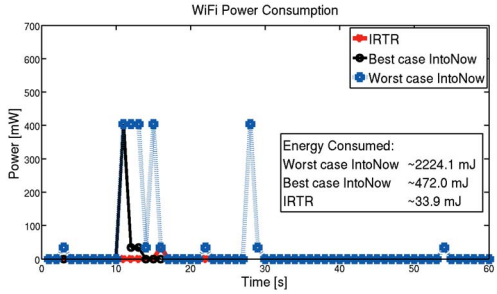

Fig. 13. WiFi interface power consumption of IRTR and IntoNow algorithms.

TABLE 6
Likelihood Estimation Computation Time and Related Gain, $N_f^R = 3200$, $N_f^C = 60$

	Direct method	RC method
Computation Time	79.7ms	60.8ms
Gain	0%	23%

of the RC method proposed in this work is of the order of $\mathcal{O}\left((M_{bins}-1)\cdot\left(\frac{3}{2}\cdot 2^{\lceil \log_2 N_f^R\rceil}\lceil \log_2 N_f^R\rceil + 2^{\lceil \log_2 N_f^R\rceil}\right)\right)$. In the current IRTR configuration N_f^C and N_f^R are fixed values, but, in future developments variable N_f^C and N_f^R may help to further improve the performance. For example N_f^C is proportional to the duration of the audio recording acquired by the smartphone. A variable recording length would lead to a variable N_f^C. N_f^R is proportional to the length of the audio of the monitored channels. In our work, as previously said, the recognition takes into account the last 5 minutes of each monitored channel, thus, leading to approximately 3200 RAF rows. In the best parameter setting, the recording length is 6 seconds, consequently N_f^C is approximately 60 rows. Table 6 shows the time required to compute the likelihood estimation, N_{bit}^D in (6) and L in (9), for the Direct and RC methods respectively as well as the percentage gain of RC with respect of the Direct method by using the setting $N_f^R = 3200$, $N_f^C = 60$.

As said, N_f^C and N_f^R may be variable, therefore it is important to show how the performance of the likelihood estimation algorithm changes based on these values. Fig. 14 compares the values $(M_{bins}-1)\left(N_f^{C2} + (N_f^R - 2)N_f^C\right)$ and $(M_{bins}-1)\cdot\left(\frac{3}{2}\cdot 2^{\lceil \log_2 N_f^R\rceil}\lceil \log_2 N_f^R\rceil + 2^{\lceil \log_2 N_f^R\rceil}\right)$ related to the theoretical complexity of the Direct and RC methods, respectively, by fixing the N_f^C value to 60 and by varying N_f^R. Fig. 15 shows the measured time required to compute the likelihood estimation into the two cases (as done in Table 6) again with $N_f^C = 60$ and varying N_f^R. The shown time measurements are taken by using MatLab on a PC running on a 32-bit Linux operating system and using an Intel Xeon(R) CPU W3520 @ 2.67GHz × 4 processor with 3.9 GiB RAM memory. The shown values are the average over 10 trials for each value of N_f^R.

Figs. 16 and 17 show the same quantities of Figs. 14 and 15 but fixing $N_f^R = 3200$ and varying N_f^C.

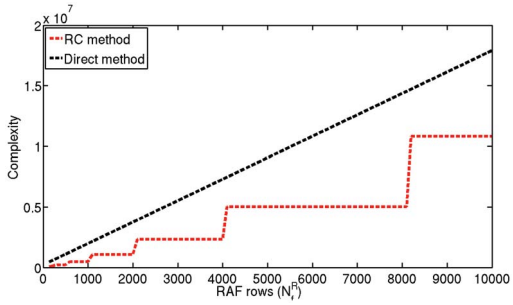

Fig. 14. Likelihood theoretical complexity of RC and Direct method versus RAF length.

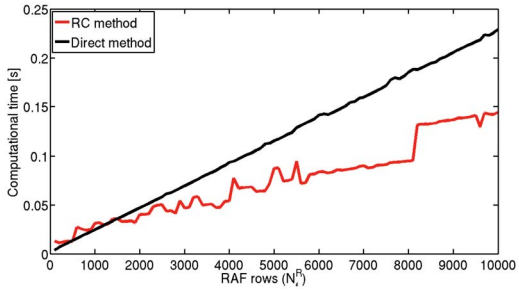

Fig. 15. Measured time to compute the likelihood estimation for RC and Direct methods versus RAF length.

It is clear that the proposed method is significantly less complex than the Direct method. It is worth remarking that if the fingerprint sent by the smartphone is taken from a longer audio snippet RC computational time remains unchanged because the RC complexity is independent of N_f^C. This will allow a future implementation of a flexible recording policy,in which the client acquires audio snippets of time-varying length without requiring any increase of computational time. On one hand, while using shorter fingerprints (or hash values) in the Direct Method leads to shorter computational times on the server side, on the other hand the proposed approach is not affected, always as far as the computational time on the server side is concerned, by the captured fingerprint size, as clearly shown in Figs. 16 and 17. This represents, from the authors' viewpoint, one of the strengths of the proposal because the size of the captured fingerprint also has a significant impact on the recognition performance. At the same time it is also true that a reduced fingerprint size implies a complexity reduction on the client side, as shown in Section 7. As a consequence, the employed size, i.e., the size of a possible hash value, should be a compromise in terms of computational complexity and detection rate: the proposed method can employ larger fingerprints, therefore increasing the recognition accuracy, without affecting the computational complexity of the matching phase (on the server side). In practice, it is possible to improve the reliability of the IRTR response without affecting its real-time functionality.

8 Conclusion and Future Work

A full system, called IRTR, capable of detecting the TV channel in real time through a short audio snippet has been introduced. We have presented an improvement of

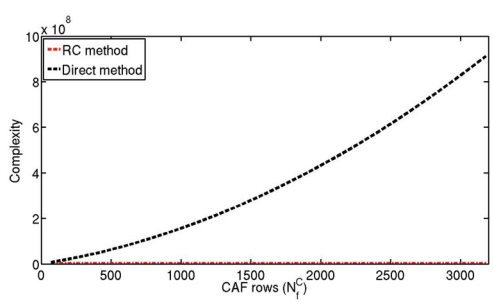

Fig. 16. Likelihood theoretical complexity of RC and Direct method versus CAF length.

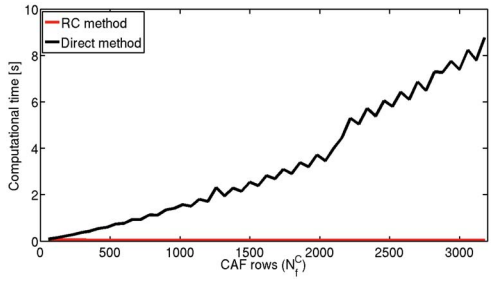

Fig. 17. Measured time to compute the likelihood estimation for RC and Direct methods versus CAF length.

the parameter configuration used by the Philips audio fingerprint computation algorithm in order to reduce the computational load and consequent energy consumption in the smartphone client. Experimental results show a significant computational time and power consumption reduction of more than 90% with a limited decrease in recognition performance. The overall performance of the system, with the selected settings, reaches a 95% correct decision rate. A comparison with the existing commercial software IntoNow has been carried out concerning energy efficiency and computational speed. The performance of IRTR is comparable to the IntoNow Best Case scenario. Moreover, an efficient likelihood estimation method running on the server and called RC (Reduced Complexity) has been proposed. The RC method allows reducing the computational complexity of the likelihood estimation of more than 20% compared to the state of the art. The computational complexity of RC has the peculiarity of being independent from the length of the audio recorded by the smartphone, and is therefore well suited for variable-recording-length architectures. Future improvements may derive from fingerprint progressive sending and from the use of users' individual watch history to compute a priori probabilities and further optimize the searching algorithm.

REFERENCES

[1] [Online]. Available: http://www.intonow.com/ci

[2] P. Mell and T. Grance, "The NIST definition of cloud computing (draft) recommendations of the National Institute of Standards and Technology," *Nist Special Pub.*, vol. 145, no. 6, pp. 1–7, 2011 [Online]. Available: http://csrc.nist.gov/publications/drafts/800-145/Draft-SP-800-145_cloud-definition.pdf

[3] [Online]. Available: http://www.tunerfish.com/

[4] [Online]. Available: http://gomiso.com/

[5] [Online]. Available: http://www.yap.tv/

[6] E. Talipov, Y. Chon, and H. Cha, "Content sharing over smartphone-based delay-tolerant networks," *IEEE Trans. Mobile Comput.*, vol. 12, no. 3, pp. 581–595, Jan. 2012.

[7] B. Han *et al.*, "Mobile data offloading through opportunistic communications and social participation," *IEEE Trans. Mobile Comput.*, vol. 11, no. 5, pp. 821–834, May 2012.

[8] E. Koukoumidis, M. Martonosi, and L.-S. Peh, "Leveraging smartphone cameras for collaborative road advisories," *IEEE Trans. Mobile Comput.*, vol. 11, no. 5, pp. 707–723, May 2012.

[9] M. Dong and L. Zhong, "Chameleon: A color-adaptive web browser for mobile OLED displays," *IEEE Trans. Mobile Comput.*, vol. 11, no. 5, pp. 724–738, May 2012.

[10] I. Bisio *et al.*, "Opportunistic estimation of television audience through smartphones," in *Proc. SPECTS*, Genoa, Italy, Jul. 2012, pp. 1–5.

[11] Yahoo and Nielsen. (2010). "Mobile shopping framework, the role of mobile devices in shopping process," Tech. Rep. [Online]. Available: http://www.slideshare.net/ashmeed25/mobileshoppingframeworkstudy2010whitepaper-final

[12] [Online]. Available: http://www.notube.tv/

[13] [Online]. Available: http://www.barb.co.uk/

[14] Ofcom. (2011). "Communications Market Report: U.K.," Tech. Rep. [Online]. Available: http://stakeholders.ofcom.org.uk/binaries/research/cmr/cmr11/UK_CMR_2011_FINAL.pdf

[15] J. Haitsma and T. Kalker, "A highly robust audio fingerprinting system with an efficient search strategy," *J. New Music Res.*, vol. 32, no. 2, pp. 211–221, 2003 [Online]. Available: http://dx.doi.org/10.1076/jnmr.32.2.211.16746

[16] L. Ghouti, A. Bouridane, and M. Ibrahim, "A fingerprinting system for musical content," in *Proc. IEEE Int. Conf. Multimedia Expo.*, Toronto, ON, USA, Jul. 2006, pp. 1989–1992.

[17] C.-S. Lu, "Audio fingerprinting based on analyzing time-frequency localization of signals," in *Proc. IEEE Workshop Multimedia Signal Process.*, Dec. 2002, pp. 174–177.

[18] Y. Ke, D. Hoiem, and R. Sukthankar, "Computer vision for music identification," in *Proc. IEEE CVPR* , vol. 1. Washington, DC, USA, Jun. 2005, pp. 597–604.

[19] S. Baluja and M. Covell, "Content fingerprinting using wavelets," in *Proc. CVMP*, London, U.K., Nov. 2006, pp. 198–207.

[20] P. Cano, E. Batle, T. Kalker, and J. Haitsma, "A review of algorithms for audio fingerprinting," in *Proc. IEEE Workshop Multimedia Signal Process.*, New York, NY, USA, Dec. 2002, pp. 169–173.

[21] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, 3rd ed. Orlando, FL, USA: Academic, 2006.

[22] J. Lourens, "Detection and logging advertisements using its sound," *IEEE Trans. Broadcast.*, vol. 36, no. 3, pp. 231–233, Sep. 1990.

[23] F. Kurth, A. Ribbrock, and M. Clausen, "Identification of highly distorted audio material for querying large scale data bases," in *Proc. Audio Eng. Soc. Convention*, 2002 [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=11325

[24] Y. Z. Qian, H. J. Dou, and Y. Feng, "A novel algorithm for audio information retrieval based on audio fingerprint," in *Proc. ICINA*, vol. 1. Kunming, China, Oct. 2010, pp. V1–266 –V1–270.

[25] J. Cerquides, "A real time audio fingerprinting system for advertisement tracking and reporting in FM radio," in *Proc. 17th Radioelektronika Int. Conf.*, Brno, Czech Republic, Apr. 2007, pp. 1–4.

[26] D. Mukherjee, T. Chattopadhyay, S. Bhattacharya, A. Ghose, and P. Misra, "An architecture for real time television audience measurement," in *Proc. IEEE ISCI*, Kuala Lumpur, Malaysia, Mar. 2011, pp. 611–616.

[27] T. L. Blum, D. F. Keislar, J. A. Wheaton, and E. H. Wold, "Method and article of manufacture for contentbased analysis, storage, retrieval, and segmentation of audio information," U.S. Patent No. 5 918 223, Jun. 1999.

[28] P. Cano, E. Batlle, H. Mayer, and H. Neuschmied, "Robust sound modeling for song detection in broadcast audio," in *Proc. AES 112th Int. Conf.*, 2002, pp. 1–7.

[29] E. Allamanche, "Content-based identification of audio material using MPEG-7 low level description," in *Proc. ISMIR*, 2001.

[30] J. Seo *et al.*, "Audio fingerprinting based on normalized spectral subband centroids," in *Proc. ICASSP*, vol. 3. Mar. 2005, pp. 213–216.

[31] A. L. Wang, "An industrial-strength audio search algorithm," in *Proc. 4th ISMIR*, S. Choudhury and S. Manus, Eds. Int. Soc. Music Inf. Retrieval. Oct. 2003, pp. 7–13 [Online]. Available: http://www.ismir.net: ISMIR; http://www.ee.columbia.edu/~dpwe/papers/Wang03-shazam.pdf

[32] M. R. Jim Barton. *Tivo Official Website* [Online]. Available: http://www.tivo.com

[33] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*. Englewood Cliffs, NJ, USA: Prentice–Hall, 1975.

[34] H. Fastl and E. Zwicker, *Psychoacoustics: Facts and Models*. New York, NY, USA: Springer, 2006.

[35] K. M. Miettinen, *Nonlinear Multiobjective Optimization*. Boston, MA, USA: Kluwer Academic, 1998.

[36] (2011, Oct.). *Auditel, Sintesi Mensile 1A* (in Italian) [Online]. Available: http://www.auditel.it/dati/

[37] H. W. Lilliefors, "On the Kolmogorov-Smirnov test for normality with mean and variance unknown," *J. Amer. Statist. Assoc.*, vol. 62, no. 318, pp. 399–402, 1967 [Online]. Available: http://www.jstor.org/stable/2283970

[38] L. Zhang *et al.*, "Accurate online power estimation and automatic battery behavior based power model generation for smartphones," in *Proc. IEEE/ACM/IFIP CODES+ISSS*, New York, NY, USA, Oct. 2010, pp. 105–114.

Igor Bisio (S'04-M'08) received the Laurea and PhD degrees in Telecommunication Engineering from the University of Genoa, Italy, in 2002 and 2006, respectively. He is currently an Assistant Professor and he is a member of the Digital Signal Processing (DSP) and Satellite Communications and Networking (SCNL) Laboratories at the University of Genoa. His current research interests include resource allocation and management for satellite and space communication systems, signal processing over smartphones. He is a member of the IEEE.

Alessandro Delfino received the B.Sc degree in Telecommunication Engineering in 2007 and the M.Sc degree, also in Telecommunication Engineering, in 2010, with a thesis on Audio Fingerprinting, both from the University of Genoa. In 2010, he worked on MAC protocols for Cognitive Radio at European funded Joint Research Center (JRC) of Ispra, Italy. He is currently a PhD student at the University of Genoa, and his current research interests include audio fingerprinting and audio information retrieval and cognitive radio.

Fabio Lavagetto is currently a full professor in Telecommunications at the DITEN Department of the University of Genoa. Since 2008, he has been Vice-Chancellor with responsibility for Research and Technology Transfer at the University of Genoa. Since 2005, he has been Vice-Chair of the Institute for Advanced Studies in Information Technology and Communication. Since 1995, he has been the head of research of the Digital Signal Processing Laboratory of the University of Genoa. He was General Chair of several international scientific conferences and has authored over 100 scientific publications in international journals and conferences.

Mario Marchese (S'94-M'97-SM'04) received the Laurea degree (cum laude) in electronic engineering and PhD degree in telecommunications from the University of Genoa, Genoa, Italy, in 1992 and 1996, respectively. He is currently an Associate Professor with the DITEN Department, University of Genoa. His current research interests include satellite and radio networks, transport layer over satellite and wireless networks, quality of service and data transport over heterogeneous networks, and applications for smartphones.