Sistema inteligente para el tratamiento de ruidos

G.M.Barrera, F.D.Goldenstein, D.M.López de Luise
Universidad de Palermo (Tel.: 54-11-5199-4520, aigroup@palermo.edu)

1. Objetivos y alcance

El principal objetivo es lograr recuperar sonidos similares automáticamente de una base de datos con sonidos previamente almacenados. Se utiliza un sonido como argumento de la consulta y se obtienen los resultados con un grado de confiabilidad parametrizable.

Este estudio se desarrolla en el marco de las investigaciones realizadas en el laboratorio AIGroup, bajo el nombre de proyecto FIC.

En la propuesta del sistema se incluye la capacidad de distinción de ruidos, música, voz y combinación de los anteriores.

2. Antecedentes

La manipulación más frecuente de archivos de sonidos es por índices con descripciones textuales. Cada archivo de audio suele ser descripto por palabras cargadas manualmente [1]. Estas palabras (típicamente denominadas metadatos) son utilizadas como índices para realizar las búsquedas contra la base de datos. La dificultad radica en describir mediante palabras ciertos sonidos tales como ruidos. Un inconveniente adicional a tener en cuenta suele ser la gran cantidad de tiempo necesario para realizar esta descripción.

Otro mecanismo ampliamente usado en las búsquedas consiste en el filtrado de archivos candidatos de audio con redes neuronales [2]. Esta aproximación requiere la preparación de tres lotes de datos: lote de entrenamiento de dichas redes, de ajustes y de validación. La efectividad de este método depende de que los datos procesados sean reales y representativos de la población.

También cabe mencionar los modelos estadísticos como el Hidden Markov Models (HMMs)[3], que se utilizan para clasificar voz a través del reconocimiento de patrones. Sin embargo este método no es aplicable a otros tipos de sonidos tales como ruidos ya que estos no cuentan con patrones.

El enfoque que más se aproxima al empleado en este trabajo, es la caracterización de sonidos utilizando Fast Fourier Transformation (FFT). Esta técnica permite caracterizar los sonidos, filtrar ruidos y comparar sonidos a través del análisis de sus frecuencias y amplitudes. La misma se utiliza para validar archivos de audio pero no es determinante al momento de buscar sonidos con características similares [4].

Si bien es factible mencionar que otras estrategias de manipulación tales como Wavelets [2] y Filtros de Kalman [5] aportan un tratamiento de mayor complejidad, se prefirió postergar su estudio dando prioridad al refinamiento de la lógica basada en Algoritmos Genéticos.

3. Proyecto FIC

3.1 Tecnología utilizada

El lenguaje de programación elegido es Java por su portabilidad y por tratarse de un lenguaje orientado a objetos en evolución constante. El motor de la base de datos fue desarrollado en XML específicamente para el proyecto. En la Fig. 1 se puede ver un modelo típico de registro de datos empleado en esta base.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <data>
  - <wav filename="bang_1.wav">
      <SD value="629013.7855331028" />
      <average value="915844.6285963663" />
      <cv value="0.6868127692107955" />
      <fft value="4185830.8486146666" />
    + <frequencies>
    </wav>
  - <wav filename="bang_2.wav">
      <SD value="1289503.817826549" />
      <average value="1363796.2654998596" />
      <cv value="0.9455252594887545" />
      <fft value="7835232.135229253" />
    - <frequencies>
        <frequency>257004.0</frequency>
        <frequency>115831.90765923387</frequency>
        <frequency>97307.79849039715</frequency>
        <frequency>82185.08356303671</frequency>
```

Fig.1: Registro de datos.

La metodología de análisis de datos es por medio de un Algoritmo Genético [6], debido a su flexibilidad en el manejo de parámetros múltiples y complejos, consiguiendo de esta manera soluciones óptimas rápidamente (gran capacidad de convergencia).

3.2 Arquitectura

La aplicación Java desarrollada tiene una arquitectura modular (ver Fig.2), y además cada módulo es paramétrico. Esta última característica es la que permite cambiar la configuración del motor de indexación en base a la categoría de sonido con la que se trabaja. A su vez, facilita la ampliación y el mantenimiento del sistema, permitiendo cambiar un módulo sin afectar el funcionamiento del resto.

Los componentes arquitectónicos de alto nivel son:

- Sound Utilities: Manipulación de sonidos.
- FIC Engine: Grabación y búsqueda de sonidos similares.
- Search Engine: Búsqueda de sonidos.
- DBMS: Motor de base de datos en formato XML.
- DB: Base de datos.

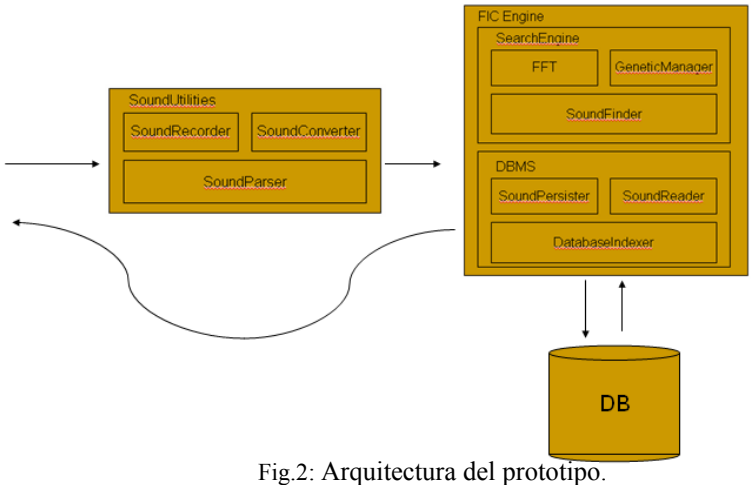

Fig.2: Arquitectura del prototipo.

3.3 Implementación

Los sonidos se categorizan en: voz, ruido, música y combinación de los anteriores. Esto se debe a que cada categoría tiene características distintivas y, en búsqueda optimal, es conveniente realizar un análisis acorde a las mismas. Un ejemplo de dicha importancia es el hecho de que un filtro de ruidos para analizar voz, sería destructivo para el procesamiento de ruidos puesto que se perdería información esencial.

El formato de audio utilizado es PCM de 16 bits y 22Khz sin compresión, debido a que es un formato estándar en la manipulación de audio, a contraposición al formato MP3, tan ampliamente difundido para escuchar música, pero que por su alta compresión pierde información valiosa.

Para el tratamiento de los sonidos se utilizan las características distintivas de los mismos, siendo éstas las frecuencias, el desvío estándar, y la media de las amplitudes de las frecuencias. Esta información es almacenada en Nodos de un archivo XML, en donde cada característica es un nodo y su valor es un atributo del mismo.

El tratamiento de sonidos se divide en dos procesos: la grabación y la búsqueda de sonidos en la base de datos.

El proceso de grabación de sonidos en la base de datos (ver Fig.3) comienza con la digitalización de un sonido proveniente de un micrófono o con la lectura de un sonido previamente almacenado en un archivo con formato de audio. Aquellos sonidos que no tengan el formato de audio definido en la aplicación son procesados por el componente de software *SoundConverter* para generar un sonido en el formato de audio preestablecido en la aplicación.

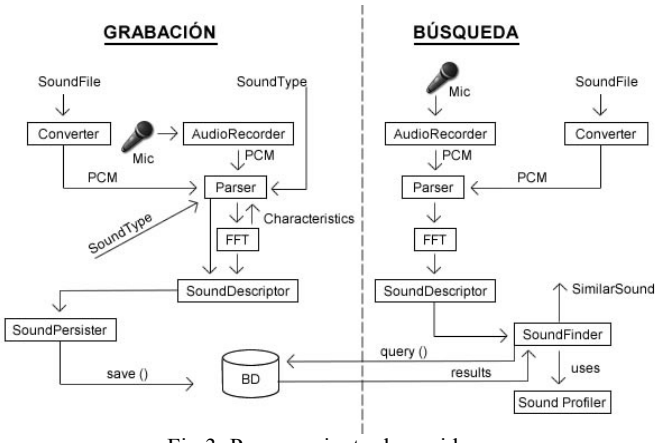

Fig.3: Procesamiento de sonidos.

El resultado del proceso de grabación es almacenado en un archivo XML con el formato previamente descripto.

Una vez categorizados, los archivos son convertidos en vectores de números enteros en memoria, que representan la amplitud de la onda sonora en función del tiempo. La dependencia del tiempo resulta perjudicial para la comparación de sonidos debido a que dos sonidos iguales pero desfasados en el tiempo serían considerados distintos por un analizador temporal. Para solucionar este inconveniente se aplica la Transformación Rápida de Fourier (FFT) y se obtienen las amplitudes de la onda en función de las frecuencias.

En cuanto al proceso de búsqueda, éste comienza de manera similar: se arma el vector de frecuencias independientemente del origen del sonido, se aplica FFT y se obtienen las características del sonido. Se realiza una comparación de las características del sonido en cuestión con los sonidos de la misma categoría que se encuentran almacenados en la base de datos. Esta comparación consta de tres etapas (ver Fig. 4):

-Pre-filtrado: se descartan los sonidos candidatos de la base de datos que tengan características que no se consideren suficientemente similares (en base a la configuración del motor indexador) conforme a las características del sonido ingresado por el usuario. Para ésto, se define una función de aceptación (ver Ec. 1) que considera una métrica de distancia y una ponderación para cada característica de los sonidos. En base a la imagen de esta función y la configuración del motor de

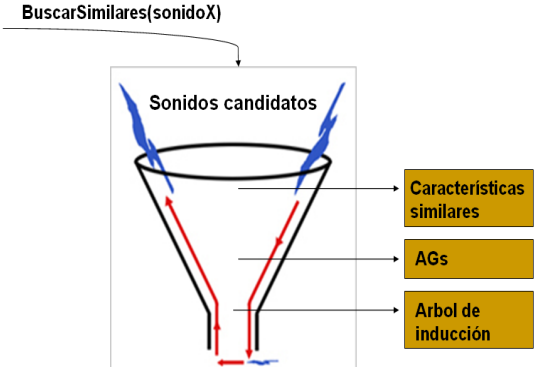

Fig.4: Etapas de búsqueda.

indexación, se descartan los sonidos que no satisfagan las condiciones definidas en el motor.

$$f(x) = \left| \frac{\sum \frac{frec_B}{frec_A}}{N-1} - radio \right| \cdot p_1 + |sd - radio| \cdot p_2 + |media\_frec - radio| \cdot p_3$$

Ecuación 1: Función de aceptación

-Filtrado: selecciona los sonidos candidatos utilizando algoritmos genéticos (AGs). Los sonidos se codifican en cromosomas utilizando el Código Gray (Gray Code) [7] porque esta codificación binaria particular es especialmente útil para implementar la mutación de genes ya que cambiando un bit se puede pasar al número decimal sucesor o al predecesor.. También se tiene en cuenta que las características de los sonidos están agrupadas en segmentos al momento de aplicar las operaciones de AGs de mutación, cruza y selección [6]. El éxito de este proceso evolutivo de la población depende principalmente de la función de optimización elegida (función de fitness), en la cual se ponderan las características de los sonidos. La Ec. 2 representa la función mencionada deducida para los ruidos. En la cual p_i corresponde a la ponderación de la característica del término, frq1(i) es la frecuencia i-ésima del sonido buscado, frq2(i) es la frecuencia i-ésima del sonido analizado de la base de datos, SD corresponde al desvío estándar y M a la media de frecuencias.

$$F(x) = p_1 \cdot (|\, \mathrm{frq}_1(i) - \mathrm{frq}_2(i)\,|) + p_2 \cdot |\, \mathrm{SD}_1 \;\; \mathrm{SD}_2\,| + p_3 \cdot |\, \mathrm{M}_1 \;\; \mathrm{M}_2\,|$$

Ecuación 2: Función de fitness

-Refinamiento: se jerarquizan los resultados obtenidos en los procesos anteriores utilizando un árbol de inducción que se basa en la función de fitness [8].

La aplicación Java (ver Fig. 5) que se utiliza tanto en el proceso de grabación como en el proceso búsqueda permite categorizar el sonido con el que se va a trabajar.

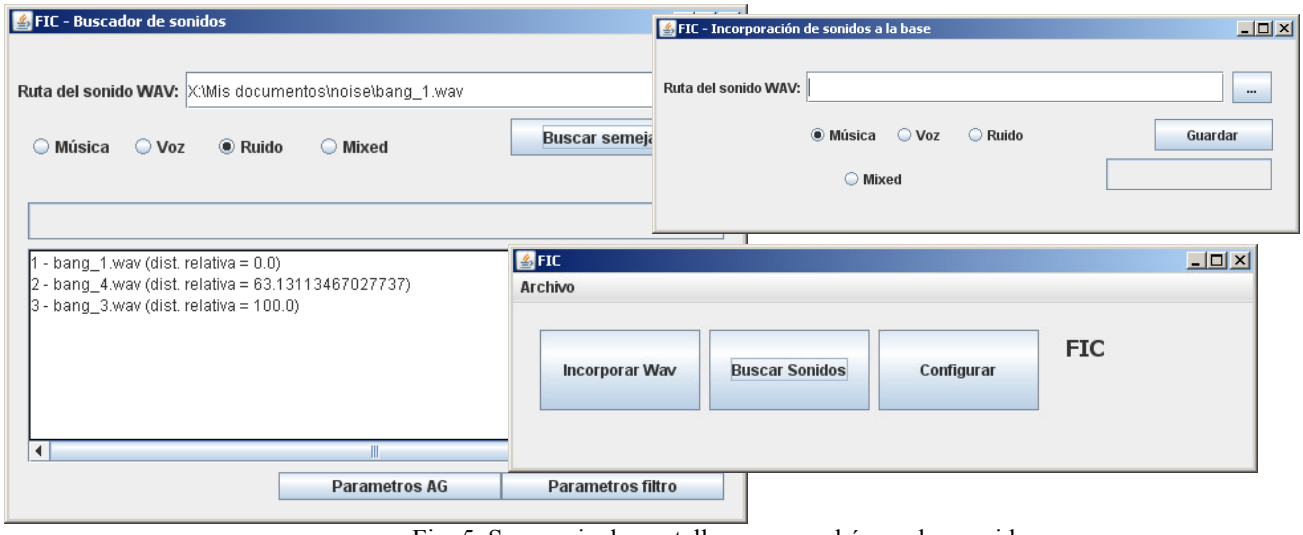

Fig. 5. Secuencia de pantallas para una búsqueda se ruido.

5. Trabajo a futuro

Tal como se mencionó en los antecedentes del trabajo, se investigarán otras estrategias de manipulación de sonidos como Wavelets y Filtros de Kalman a fin de estudiar su eficiencia relativa. El objetivo va ser el desarrollo de módulos alternativos de sonidos que permitan evaluar el

rendimiento del motor de indexación. Gracias al hecho de que la arquitectura del software desarrollado es totalmente modular, va a resultar relativamente directo adaptar la nueva estrategia de manipulación.

Posteriormente, se realizará un estudio estadístico de los resultados de las búsquedas realizadas con el motor con el fin de evaluar y refinar la función de fitness.

Teniendo en cuenta el interés de empresas y universidades acerca de la puesta en producción de software desarrollado, se implementarán interfases con fuentes de entrada/salida no convencionales como sondas, sensores y membranas capturadoras de sonidos. Una aplicación directa de esta ampliación del trabajo es la posibilidad de utilizar el motor de indexación para reconocer ruidos en un ambiente y activar alarmas en base al sonido detectado.

6. Referencias

[1] MPEG-7 Overview (version 10), http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm

[2] “Audio content description with wavelets and neural nets”. S. Rein, M. Reisslein, T. Sikora. Proceedings of ICASSP '04 (IEEE International Conference on Acoustics, Speech and Signal Processing). pp. 341-344. 2004.

[3] “Hidden Markov Models for Speech Recognition”. B. H. Juang, L. R. Rabiner. Technometrics Vol 33 number 3. pp. 251-272. 1991.

[4] “What is the Fast Fourier Transform?”. W. T. Cochram, et al. Proc. of the IEEE Vol. 55 number 10. 1967.

[5] “An Introduction to the Kalman Filter”. G. Welch, G. Bishop. TR 95-041, Department of Computer Science, University of North Carolina. 2002.

[6] “Una Introducción a la Computación Evolutiva”. A. Pérez Serrada. http://surf.de.uu.net/research/softcomp/EC/EA/. 1996.

[7] "Gray Codes and Paths on the -Cube.". E. N. Gilbert. Bell System Tech. J. 37. pp. 815-826. 1958.

[8] “Datamining: Practical Machine Learning Tools and Technics”. I. H. Witten, E. Frank, Editorial Morgan Kaufmann, 2nd edition. 2005.