CLASIFICADOR DE MÚSICA POR GÉNERO UTILIZANDO REDES NEURONALES ARTIFICIALES

ELKIN EDUARDO GARCÍA DÍAZ

GERMÁN ANDRÉS MANCERA MÉNDEZ

JORGE GUILLERMO PACHECO DÍAZ

Trabajo de Grado presentado como requisito

para optar al título de Ingeniero Electrónico

DIRECTOR : FERNANDO ENRIQUE LOZANO MARTÍNEZ Ph.D.

PONTIFICIA UNIVERSIDAD JAVERIANA

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE ELECTRÓNICA

BOGOTÁ D.C.

2003

PONTIFICIA UNIVERSIDAD JAVERIANA

FACULTAD DE INGENIERÍA

CARRERA DE INGENIERÍA ELECTRÓNICA

RECTOR MAGNÍFICO	R.P. Gerardo Remolina S.J.
DECANO ACADÉMICO	Ing. Roberto Enrique Montoya Villa
DECANO DEL MEDIO UNIVERSITARIO	R.P. Antonio Sarmiento Nova S.J.
DIRECTOR DE CARRERA	Ing. Juan Carlos Giraldo Carvajal
DIRECTOR DEL PROYECTO	Fernando Lozano Martínez Ph.D.

ARTÍCULO 23 DE LA RESOLUCIÓN No. 13 DE JUNIO DE 1946

"La universidad no se hace responsable de los conceptos emitidos por sus alumnos en sus proyectos de grado.

Sólo velará por que no se publique nada contrario al dogma y a la moral católica y por que los trabajos no contengan ataques o polémicas puramente personales. Antes bien, que se vea en ellos el anhelo de buscar la verdad y la justicia."

A mi mamá,

por su apoyo, cariño,

comprensión y afecto

a lo largo de mi vida.

Elkin Eduardo García Díaz.

A los que incondicionalmente me han brindado

su amor y apoyo siempre:

Dios, papi, mami y Lilianita.

Germán Andrés Mancera Méndez

A todas las personas que hicieron esto posible.

En especial a mi papá a mi mamá y a mi hermana.

Jorge Guillermo Pacheco Díaz.

CONTENIDO

LISTA DE TABLAS

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE GRÁFICAS

Pág.

LISTA DE ANEXOS

INTRODUCCIÓN

Bases de datos de música en formato digital cuyo tamaño era difícil imaginar hace unos años se comparten hoy en día con gran facilidad a través de las redes de computadores (por ejemplo, Internet), creando la necesidad de disponer de métodos eficientes que permitan la búsqueda y organización de dichas bases. En los últimos años, y debido al creciente fenómeno de distribución de música a través de Internet, ha surgido la necesidad de crear sistemas capaces de clasificar música de manera automática. Sin embargo, son muchos los estudios que se han venido realizando con el fin de establecer cuáles son aquellas características que constituyen un estilo musical, qué parámetros musicales son relevantes para hacer la clasificación y cuáles son las técnicas de aprendizaje de máquina más efectivas para procesar tal información **[1],[2],[3].**

Este proyecto está enfocado hacia la clasificación de canciones en tres géneros a saber: Merengue, Salsa y Vallenato. Existen varios maneras de abordar el problema de la clasificación de música. Éste estudio se centra en la realización de tal proceso mediante la identificación y extracción de características relevantes que pueden ser usadas como entradas a una Red Neuronal Artificial. Una vez se han identificado tales características, se procede a realizar el aprendizaje de máquina. Se usan dos topologías básicas: Perceptrón Multinivel y Redes de Funciones de Base Radial (Radial Basis Functions). También se ha estudiado el efecto que tiene la aplicación del algoritmo AdaBoost sobre el desempeño final del clasificador.

Los estudios que se han realizado al respecto son pocos. Seth Golub **[2]** realizó una tesis de maestría en la cual probó tres tipos diferentes de clasificadores aplicados a dos géneros similares, obteniendo un porcentaje de clasificación correcta del 77%. Paul Scott **[3]** elaboró un sistema clasificador usando un perceptrón multinivel para 4 géneros bastante diferentes, y obtuvo un porcentaje del 94.8%. Numerosos investigadores han realizado trabajos relacionados con la identificación de características relevantes a partir de la música. Por ejemplo, Foote **[4]** empleó técnicas espectrales para distinguir entre voz y música con un alto grado de exactitud, mientras que Soltau **[5]** entrenó una red neuronal auto–asociativa usando la técnica de Foote para los géneros rock, pop, tecno y clásico. Su razón de clasificación correcta fue similar a la obtenida por Golub.

En el primer capítulo de este libro se presenta una descripción de la teoría relacionada con la extracción de características, preprocesamiento y redes neuronales artificiales. En el segundo capítulo se considera todo lo referente a las especificaciones del proyecto y se hace referencia a la arquitectura, el formato de la canciones y las características de la base de datos. En el tercer capítulo se describe detalladamente la forma en la cual se desarrolló el proyecto en sus etapas de identificación, extracción y preprocesamiento de parámetros, seguido del entrenamiento y optimización del clasificador. En el cuarto capítulo se hace un análisis de los resultados obtenidos para todas las pruebas que fueron realizadas. En el quinto capítulo de exponen las conclusiones resultantes del proyecto. Finalmente se brinda una bibliografía que tiene como objetivo permitir que el lector vaya a las fuentes primarias utilizadas por los realizadores del proyecto.

1. MARCO TEÓRICO

1.1 EXTRACCIÓN DE CARACTERÍSTICAS Y PREPROCESAMIENTO

El diagrama que se presenta en la *Figura 1* permite visualizar de forma global el problema de la clasificación de música de acuerdo al género. Esta clasificación taxonómica fue realizada en primera instancia por David Gerhard **[1]** en un estudio realizado acerca de la clasificación de señales de audio. En la *Figura 1* se encuentra resaltado el cuadro correspondiente a Género, y puede verse que junto con Cultura, Compositor y Cantante, son las formas de clasificar música por Tipo. Igualmente puede observarse que la música puede ser clasificada según su contenido dependiendo de la Melodía y de las Notas.

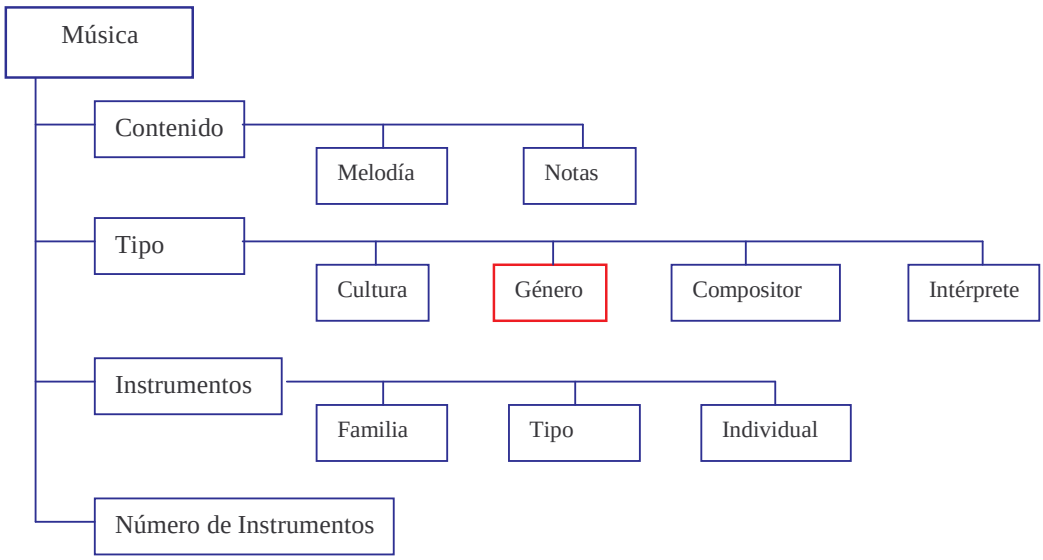


Figura 1. Taxonomía de la clasificación de música.

La razón por la cual se ha implementado este sistema se debe a que la búsqueda de una canción por género, es la forma más natural y frecuente en la cual se realiza este proceso cuando no se dispone de información más específica como el título o el intérprete. De hecho, ésta es la forma en la que cualquier persona buscaría en una tienda de música o en tiendas especializadas de Internet, en donde los discos se encuentran organizados de acuerdo al género musical por rock, pop, jazz, clásica, tropical, vallenato, etc. **[1]**

La *Figura 2* presenta un diagrama de bloques general del proyecto (ampliado en el numeral 2.1). Como se muestra, el sistema está compuesto de dos bloques interconectados entre sí. Un primer bloque se encarga de llevar a cabo la extracción de características relevantes a partir del archivo de música, mientras que el segundo bloque es el sistema clasificador, encargado de realizar la tarea de categorización una vez se ha obtenido la información necesaria y pertinente del bloque anterior.

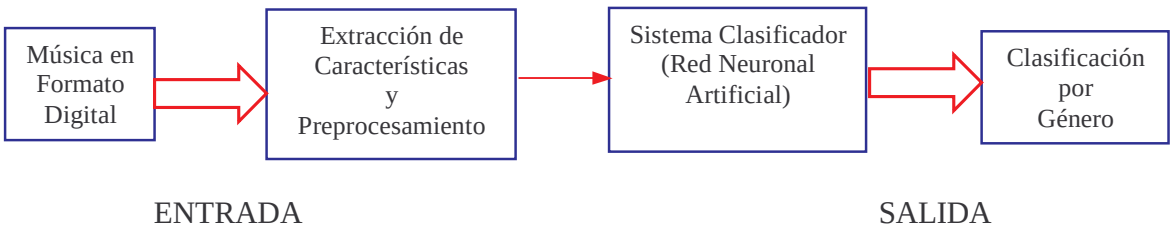


Figura 2. Diagrama de bloques general del clasificador de música.

Christopher Bishop **[6]** habla acerca de la importancia que tiene la utilización del bloque de extracción de características y preprocesamiento, y muestra el impacto del mismo sobre el rendimiento final del sistema de clasificación. El primer objetivo es, como su nombre lo

indica, extraer parámetros que permitan diferenciar entre los conjuntos que son usados como entradas al sistema completo. Esta extracción de características debe ser tal, que permita el posterior entrenamiento de la Red Neuronal Artificial, para conseguir éxito en la clasificación. El otro objetivo es llevar a cabo la reducción de la dimensionalidad. Esto es, proporcionar a la Red Neuronal un número adecuado de entradas que permita reducir la complejidad del modelo, mejorar la clasificación y reducir el tiempo necesario para llevar a cabo el entrenamiento de la Red Neuronal, puesto que se descarta información irrelevante y poco informativa. En el caso de éste proyecto es claro que la utilización de la canción completa como entrada a la Red sería altamente ineficiente y conduciría a tiempos de cómputo inmanejables para la fase de entrenamiento. Por ejemplo, una canción en formato digital WAV monofónica muestreada a 44100 Hz produciría una cantidad de entradas no inferior a 5’000.000.

Con base en el análisis de las formas de onda de diversas canciones y empleando herramientas que permitan analizar las señales, es posible extraer un número adecuado de parámetros en esta parte del sistema. Algunas técnicas de análisis en tiempo y frecuencia permitieron obtener características que contienen información relevante de tipo musical (intensidad, duración, timbre, tono, compás, pulso, escalas melódicas, instrumentos) que está relacionada con el género musical al que pertenecen las muestras analizadas.

1.1.1. Extracción de características a partir del dominio del tiempo

Para realizar la tarea de clasificación de música se pueden tener en cuenta diversas características de su forma de onda, como son Espectro de frecuencia, Potencia, Amplitud en Tiempo, etc. Dentro de las características de tiempo, existen muchos tipos de parámetros como lo son, por ejemplo, el volumen promedio y el volumen dinámico.

El volumen Promedio se calcula como el valor RMS de la señal de onda, es decir tomamos un número N de muestras y a partir de este calculamos este valor de la siguiente forma:

$$V = \sqrt{\frac{1}{N}\sum_{i=0}^{N-1} s_n^2(i)} \qquad \textbf{(1.1)}$$

El volumen promedio permite tener una noción del valor promedio de las muestras, es decir un promedio de la potencia de energía, parámetro que por sí solo no permite distinguir un ritmo de otro, pero que en combinación con otros es de utilidad.

El volumen Dinámico se calcula tomando el valor máximo y el valor mínimo dentro de la muestra y sacando una relación porcentual de su variación

$$VDR = \frac{V_{\max} - V_{\min}}{V_{\max}} \qquad \textbf{(1.2)}$$

Estos dos datos permiten conocer la acumulación de diferencias existentes sobre ventanas de tiempo tomadas sobre la misma muestra, que son sucesivas.

En la ejecución de la música, el intérprete usa las variaciones de tiempo, dinámica y articulación. La teoría indica que esas variaciones se encuentran cercanamente relacionadas con la estructura musical **[7]**. Una de las herramientas de análisis de datos es usar la correlación con el objetivo de encontrar regularidades, las cuales, de acuerdo con la hipótesis indicada anteriormente, corresponden a dicha estructura.

Cualquier desviación de una distribución típica de frecuencia es vista como una modificación estructural, la cual permite diferenciar un ritmo de otro. Es necesario analizar dichos cambios en diferentes tiempos, para observar si son cambios estructuralmente planeados y repetitivos que entregan un análisis de la forma de onda o si son cambios meramente aleatorios y que no presentan periodicidad a través del tiempo.

Una canción es mezcla de muchas frecuencias fundamentales que entregan el ritmo. Por esta razón se debe analizar el espectro de ésta y ver como varía a través del tiempo. Para esto se realiza la correlación del espectro de frecuencia con él mismo a lo largo de diferentes muestras de tiempo sobre una canción. Esto entrega una medida del intervalo de frecuencia en el cual existe el mayor valor de potencia, es decir la media de la autocorrelación y la forma en que decae dicho valor a través de la frecuencia, lo cual equivale a la desviación estándar. **[7]**

$$\mathrm{R}_{xy} = E\left[X_m Y_n\right] \qquad \textbf{(1.3)}$$

Estos parámetros permiten conocer una tasa de variación de una muestra a otra dentro de una misma canción, tanto en amplitud, como en contenido de frecuencia, lo cual a través

del análisis de muestras de varios géneros permite identificar dichas características como propias de cada uno de los ritmos a analizar**.**

1.1.2. Extracción de características a partir del dominio de la frecuencia

Hay una amplia gama de herramientas matemáticas que permiten realizar análisis de señales en el dominio de la frecuencia. La Transformada de Fourier es una de las más ampliamente utilizadas en este tipo de estudios **[2][3]**, y su importancia radica en el hecho que permite descomponer un amplia gama de señales en términos de funciones senoidales **[8]**. La transformada de Fourier de una señal de energía finita está definida como

$$X(\omega) = \sum_{n=-\infty}^{\infty} x(n) e^{-j\omega n} \qquad \textbf{(1.4)}$$

Una canción es muestreada a una frecuencia determinada, que para el caso de señales de audio es generalmente 44100 Hz. La razón por la cual se muestrea a esta tasa se fundamenta en el Criterio de Nyquist, el cual establece que para recuperar una señal a partir de sus muestras, éstas deben ser tomadas a una frecuencia igual a superior a dos veces la frecuencia máxima contenida en dicha señal. Una señal de audio tiene un espectro que no sobrepasa los 20 kHz, razón por la cual es posible recuperarla si la tasa de muestreo es 44.1 kHz.

Hacia mediados de los años 60's se desarrolló un algoritmo conocido como Transformada Rápida de Fourier (Fast Fourier Transform FFT) que ayudó a realizar una implementación digital y redujo en varios órdenes de magnitud el tiempo computacional que se necesitaba para realizar tal transformación **[9]**. Por esta razón y por el hecho que es una función incluida en Matlab, ésta fue ampliamente utilizada en el proyecto para obtener las transformadas. Este algoritmo toma como entrada una señal y obtiene su transformada discreta de N puntos mediante la ecuación

$$X(k) = \sum_{i=1}^{N} x(i)\,\omega_N^{(i-1)(k-1)} \qquad \textbf{(1.5)}$$

En donde $\omega_N = e^{-(j2\pi)/N}$

1.1.3. Extracción de características a partir de la transformada Wavelet

Hasta el momento se ha hablado del dominio del tiempo y del dominio de la frecuencia por separado, sin embargo surge la duda de si es necesario poseer información que relacione tiempo y frecuencia para una señal dada. Para responder se hace necesario diferenciar si la señal es estacionaria o no, puesto que una señal no estacionaria cambia su contenido en frecuencia con el transcurso del tiempo y en este caso sería útil establecer una relación entre tiempo y frecuencia. Este es el caso de la música, la cual es un tipo de señal no estacionaria.

La transformada Wavelet proporciona información que relaciona tiempo y frecuencia, es decir qué bandas de frecuencias están presentes en determinado intervalo de tiempo. Es importante anotar que es imposible establecer en qué tiempo está presente una componente

de frecuencia, pues mientras se tiene mayor precisión en el tiempo se pierde precisión en frecuencia, esto se conoce como el principio de incertidumbre.

Adicionalmente, la transformada Wavelet es multiresolución; esto es que analiza la señal a diferentes frecuencias con diferentes resoluciones, en este caso una alta resolución en tiempo y pobre resolución en frecuencia a frecuencias altas y baja resolución en tiempo y alta resolución en frecuencia a frecuencias bajas. **[10].**

1.1.3.1. Transformada continua Wavelet

La transformada continua Wavelet (CWT por sus siglas en inglés) está definida para una señal $x(t)$ de la siguiente forma

$$\Psi(\tau, s) = s^{-1/2} \int x(t) \psi\left(\frac{t-\tau}{s}\right) dt \qquad \textbf{(1.6)}$$

Donde la transformada inversa es

$$x(t) = K \iint \frac{1}{s^2} \Psi(\tau, s) \psi\left(\frac{t-\tau}{s}\right) ds d\tau \qquad \textbf{(1.7)}$$

Con la constante de normalización *K* dada por

$$K = \int \frac{|W(\psi)|^2}{|\psi|} d\omega \qquad \textbf{(1.8)}$$

Siendo $W(\psi)$ la transformada de Fourier de $\psi(t)$. **[11]**.

La transformada Wavelet $\Psi(\tau, s)$ es una función de dos variables, τ o traslación que hace referencia al desplazamiento de la función madre o prototipo $\psi(t)$ y s o escala que se refiere al ancho de dicha función. Es importante resaltar que aunque no exista la variable frecuencia dentro de la definición de la transformada Wavelet, la variable escala se relaciona con este parámetro, donde altas escalas hacen referencia a bajas frecuencias mientras bajas escalas hacen referencia a altas frecuencias **[10]**.

Existen múltiples familias de funciones Wavelet $\psi(t)$, como la función Morlet o el sombrero mejicano, entre otras. Pruebas preliminares realizadas en este estudio demostraron que los mejores resultados se obtienen utilizando el sombrero mejicano definido como

$$\psi(t) = A\left(1 - \frac{x^2}{\sigma^2}\right) e^{-x^2/2\sigma^2} \qquad \textbf{(1.9)}$$

1.1.3.2. Transformada discreta Wavelet

Utilizando la transformada Discreta Wavelet (DWT por sus siglas en inglés) es posible expandir una función $x(t)$ por medio de una serie

$$x(t) = \sum_{\tau=-\infty}^{\infty} \sum_{s=-\infty}^{\infty} a_{\tau,s} \psi_{\tau,s}(t) \qquad \textbf{(1.10)}$$

donde

$$\psi_{\tau,s}(t) = 2^{s/2} \psi\left(2^s t - \tau\right) \qquad \textbf{(1.11)}$$

Así el grupo de coeficientes bidimensionales $a_{\tau,s}$ es llamado transformada discreta Wavelet. **[11]**.

1.1.4. Preprocesamiento

Con el fin de obtener mejores resultados durante el proceso de entrenamiento de la Red Neuronal, es necesario realizar una labor de preprocesamiento para transformar los parámetros extraídos de las canciones en una nueva representación. De la misma manera, la incorporación de conocimiento previo aumenta el desempeño general del sistema. El conocimiento previo hace referencia a información relevante que puede ser usada para desarrollar una solución, y que es adicional a aquella que se encuentra en el conjunto de entrenamiento **[6]**.

La reducción de la dimensionalidad de los datos de entrada es una de las funciones primordiales de la tarea del preprocesamiento. La extracción de parámetros relevantes a partir de las entradas se conoce como "Extracción de Características" y su importancia, radica en el hecho que permite disminuir la complejidad del modelo y aumentar la capacidad de generalización de la red. Esta disminución conlleva una mejora de los tiempos necesarios para poder entrenar la Red Neuronal, debido a que se descarta contenido poco informativo. Además es adecuado realizar otro proceso conocido como escalamiento lineal, mediante el cual se pretende que todos los parámetros de entrada tengan valores en el mismo rango. Esta transformación es especialmente adecuada si las entradas tienen valores que difieren en varios órdenes de magnitud. Para el proyecto se realizó una

normalización de todas las variables con respecto a los valores que son obtenidos para todas las muestras, haciendo que todas las entradas quedaran dentro del intervalo [-1,1].

Con el fin de reducir la dimensionalidad de las entradas, es útil seleccionar un subconjunto de ellas y descartar el otro, partiendo del supuesto que hay ciertas entradas que contienen mayor información que otras. Un procedimiento que seleccione las entradas adecuadas debe contar con un criterio que permita discernir qué variables son mejores que otras. El subconjunto de entradas óptimo dependerá, en gran medida, del sistema clasificador que será empleado. Así mismo, este proceso de selección debería idealmente ser realizado entrenando la Red en repetidas ocasiones para diferentes conjuntos de entrada y evaluando los resultados que se obtienen para cada uno de ellos. Una aproximación como ésta resulta computacionalmente inadecuada por cuanto el entrenamiento debería ser realizado demasiadas veces. Por esta razón se realizó una transformación conocida como blanqueo **[12]** o KLT (Ver Numeral 3.1.5.). Inicialmente se agrupan las variables de entrada x_i en un vector $x = (x_1 , x_2 , \dots , x_d)^T$ que tiene como media un vector y como covarianza una matriz con respecto a todos las muestras del conjunto de entrenamiento dados por:

$$\bar{x} = \frac{1}{N}\sum_{n=1}^{N} x^n \qquad \textbf{(1.12)}$$

$$\Sigma = \frac{1}{N-1}\left(x^n - \bar{x}\right)\left(x^n - \bar{x}\right)^T \qquad \textbf{(1.13)}$$

Haciendo uso de la ecuación de los valores propios para la matriz de covarianza

$$\Sigma u_j = \lambda_j u_j \qquad \textbf{(1.14)}$$

Puede definirse una transformación lineal de la entradas que está dada por

$$\tilde{x}^n = \Lambda^{-1/2} U^T \left(x^n - \bar{x} \right) \quad \textbf{(1.15)}$$

En donde

$$U = (u_1, ..., u_d) \quad \textbf{(1.16)}$$

$$\Lambda = diag(\lambda_1, ..., \lambda_d) \quad \textbf{(1.17)}$$

En el nuevo espacio de coordenadas que ha sido generado, los datos tienen una media de cero y una matriz de covarianza dada por la identidad.

1.2 REDES NEURONALES ARTIFICIALES

Las Redes Neuronales surgieron como una aproximación a la Inteligencia Artificial hecha por parte de la Ingeniería mediante la cual se trata de modelar (o imitar) el funcionamiento físico del cerebro. Actualmente han encontrado gran acogida en procesos de "bajo nivel" como el control de motores o el reconocimiento de patrones (clasificación). Los estudios han revelado que un ser humano promedio tiene cerca de 10^9 neuronas en su cerebro, unidas entre sí por un número cercano a 10^{11} conexiones. Se presume que todas estas unidades realizan operaciones sencillas, razón por la cual un proceso cualquiera es un fenómeno emergente que se da gracias a la interacción de muchas partes.

En este enfoque, una neurona es considerada como un dispositivo computacional que tiene 2 clases de entradas: Excitatorias e Inhibitorias, así como una salida (*Figura 3*). Las Redes Neuronales han sido aplicadas con gran éxito en el área del Reconocimiento de Patrones, debido al hecho que se ha demostrado que si se dispone de un conjunto suficientemente amplio de entradas correctamente clasificadas, es posible llevar a cabo un procedimiento de aprendizaje para discernir cuál ha sido el criterio de clasificación y generalizarlo de forma tal que se consiga éxito en la clasificación de nuevas entradas que no fueron usadas durante el proceso de aprendizaje de la red **[6]**.

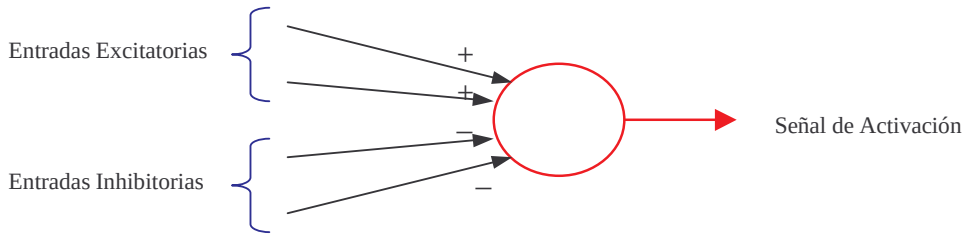


Figura 3. Modelo computacional de una neurona.

Una Red Neuronal Artificial es el segundo gran bloque del proyecto, y tiene como función llevar a cabo la clasificación de las canciones en los géneros Merengue, Salsa y Vallenato. Con el fin de entrenar la red se utilizó Aprendizaje Supervisado. En este método de aprendizaje, el entrenamiento se realiza a partir de ejemplos, es decir, se parte de una base de datos de canciones correctamente etiquetadas (es decir que se conoce el género de cada una las canciones contenidas en dicho conjunto) que permite entrenar y evaluar la red. Las entradas y salidas del sistema son conocidas y se pretende encontrar la función desconocida h mediante la cual están relacionadas. Se cuenta con un conjunto de ejemplos de tamaño

finito N y se estima una función $\hat{h}$ a partir de tales datos mediante la utilización de algún algoritmo.

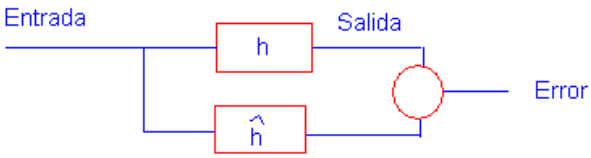


Figura 4. Relación entre la entrada y la salida mediante las funciones h y $\hat{h}$.

Es importante anotar que el objetivo del proceso de entrenamiento no es ajustarse a los datos de entrada, sino que la red no se equivoque en la clasificación de entradas que no fueron utilizadas para el entrenamiento. Esto significa que se busca que la probabilidad de que $\hat{h}(x) \neq y$ sea baja (Idealmente debería ser cero). Para tal fin se asume que los datos del conjunto de entrenamiento y los datos del conjunto de prueba provienen de la misma distribución de probabilidad. Las funciones h y $\hat{h}$ difieren por diversas razones, aún cuando el objetivo del algoritmo del aprendizaje es hacerlas tan parecidas como sea posible.

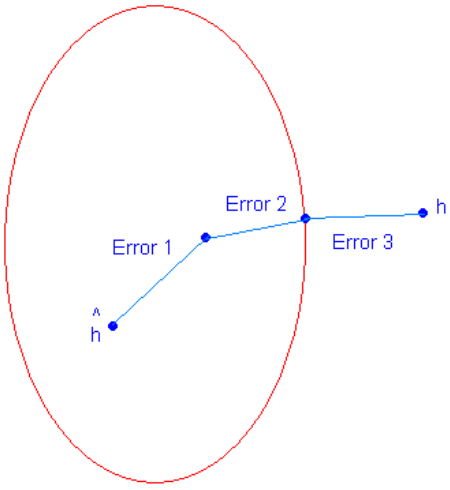


Figura 5. Factores causantes del hecho que la función real y la estimada difieran.

En la *Figura 5* se muestran esquemáticamente los diversos errores que se presentan en el proceso de entrenamiento de una Red Neuronal. El primero de ellos se conoce como error de aproximación y se debe a que el modelo no es lo suficientemente complejo. El segundo se conoce como error de estimación y se debe a que el conjunto de entrenamiento del cual de dispone es finito. El último tipo de error es de tipo computacional y su razón es el tiempo de procesamiento.

En el transcurso de este proyecto de diseñaron, entrenaron y evaluaron los resultados obtenidos para dos algoritmos de aprendizaje ampliamente conocidos: Propagación inversa (Backpropagation) y Funciones de Base Radial (Radial Basis). Finalmente se usó una técnica denominada Boosting que permite aumentar el desempeño de un clasificador débil mediante la disminución del error de clasificación. (Ver Numeral 3.2.)

1.2.1 Red de propagación inversa (Backpropagation)

1.2.1.1. Perceptrón Multinivel

El perceptrón multinivel consiste en una red de procesamiento de elementos ubicados en capas. Típicamente se manejan tres capas, una capa de entrada que acepta las variables de entrada que se usan en el proceso de clasificación, una capa oculta, y una capa de salida con una neurona por cada clase. El principio de la red consiste en que cuando los datos de patrones de entrada son presentados, las neuronas de la de la red realizan cálculos en cada capa y la señal de salida indica cuál es la clase apropiada para el dato de entrada. Se espera

que exista un valor de salida alto en la neurona de la clase correcta y un valor de salida bajo en las demás. Éste pertenece a la clase de redes neuronales con aprendizaje supervisado. Éste es de propósito general, flexible, y permite modelar funciones no lineales, además, la complejidad de esta red, puede ser modificada cambiando el número de capas o el número de unidades por capa. La gran ventaja de esta arquitectura es que si se tienen suficientes unidades escondidas y se tienen suficientes datos, se ha comprobado que la red neuronal puede aproximar cualquier función con cualquier grado de exactitud.

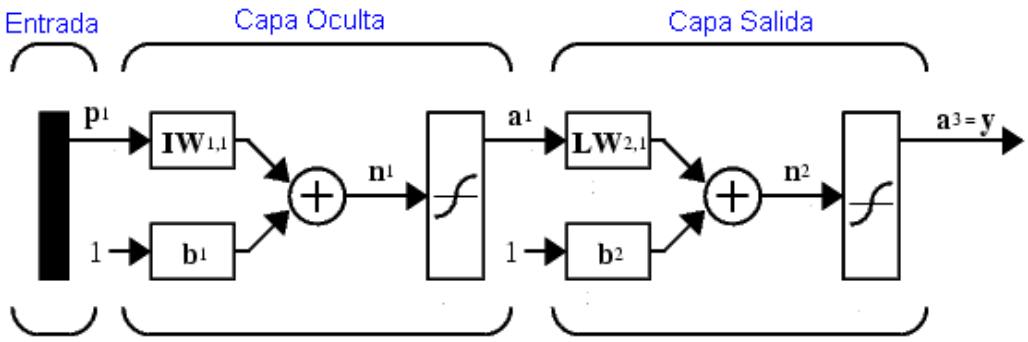


Figura 6. Perceptrón multinivel con una capa oculta. **[10]**

Ahora bien, el objetivo que se desea alcanzar en definitiva, es entrenar la red; intentando disminuir el error cuadrático medio de la salida con respecto a la entrada, para esto se utiliza un algoritmo de entrenamiento, cuya finalidad es precisamente disminuir el error de entrenamiento a través de la variación iterativa de los de los valores de los pesos que conectan las diferentes neuronas, para corregir los valores de las salidas estimadas. Esto se conoce como entrenamiento.

1.2.1.2. Backpropagation

El algoritmo utilizado para entrenar el perceptrón multinivel se conoce con el nombre de Backpropagation o propagación inversa del error. Este algoritmo es una generalización de la regla de aprendizaje *Widrow –Hoff* para redes de múltiples capas y funciones de transferencia diferenciables no lineales. Los vectores de entrada y sus correspondientes salidas son usadas para entrenar una red que puede aproximar una función, asociando un vector de entradas con un vector específico de salidas, o clasificando un vector de entradas en una forma apropiada **[13]**. En el entrenamiento se usa el algoritmo del gradiente descendiente, donde los pesos se mueven en la dirección contraria a la del gradiente de la función de costo. El término Backpropagation se refiere a la forma como se computa el gradiente y como se comporta el error.

Se espera que una red correctamente entrenada produzca respuestas correctas cuando se presentan entradas nunca vistas. Esta propiedad de generalización hace posible entrenar una red con un conjunto representativo de parejas entrada / salida y que se obtengan buenos resultados sin necesitad de entrenar la red con todas las posibles parejas .

El algoritmo de aprendizaje se basa en el gradiente descendiente del error, donde el error es la diferencia de la salida actual del sistema, y la salida esperada **[14]**.

$$E_k = \frac{1}{2}\sum_{i=1}^{M}\left(O_i^{(k)} - O_i\right) \qquad \textbf{(1.18)}$$

Donde M es el número de neuronas, O_i es la salida esperada y $O_i^{(k)}$ es la salida para cada época.

1.2.2. Red de funciones de base radial (radial basis functions)

Este tipo de redes neuronales hacen uso de las funciones de base radial. Éstas son un tipo especial de funciones cuya característica fundamental es el hecho que crecen o decrecen monótonamente a medida que la distancia entre el punto central y otro punto aumenta. La función de base radial más empleada es la función de Gauss que tiene la forma

$$h(x) = \exp\left(\frac{-\|x-c\|^2}{r^2}\right) \qquad \textbf{(1.19)}$$

Hacia 1988, Broomhead y Lowe **[15]** introdujeron el modelo de una red de funciones de base radial que puede ser usada en redes de una o múltiples capas. Sin embargo, este tipo de redes han sido asociadas primordialmente con una topología de una capa escondida, como la que se muestra en la *Figura 7* **[16]**.

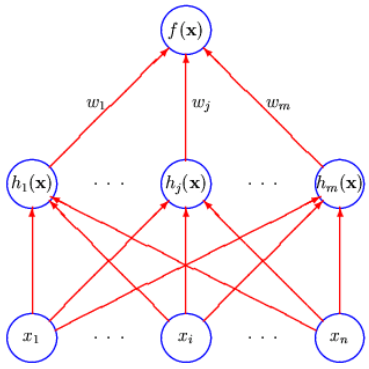


Figura 7. Configuración típica de una Red de funciones de Base Radial.

La relación entre la entrada y la salida para este tipo de redes es

$$y_k(x) = \sum_{j=1}^{M} w_{kj}\phi_j(x) + w_{k0} \qquad \textbf{(1.20)}$$

Para el caso de una función gaussiana se tiene que

$$\phi_j = \exp\left(\frac{-\|x-\mu_j\|^2}{2\sigma_j^2}\right) \qquad \textbf{(1.21)}$$

En donde *x* es un vector cuya dimensión es el número de parámetros de entrada y μ_j es un vector que determina el centro de la función de base radial ϕ_j. El modelo de una neurona para este tipo de red se ilustra en la *Figura 8*, mientras que la arquitectura de estas redes puede ser observada en la *Figura 9*

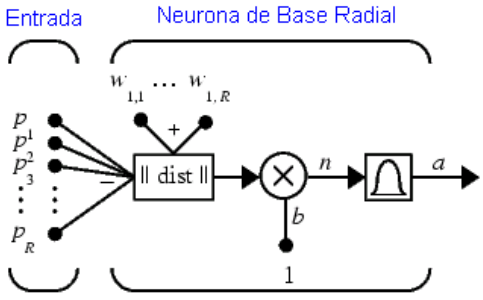


Figura 8. Modelo de una neurona para una red de funciones de base radial **[10].**

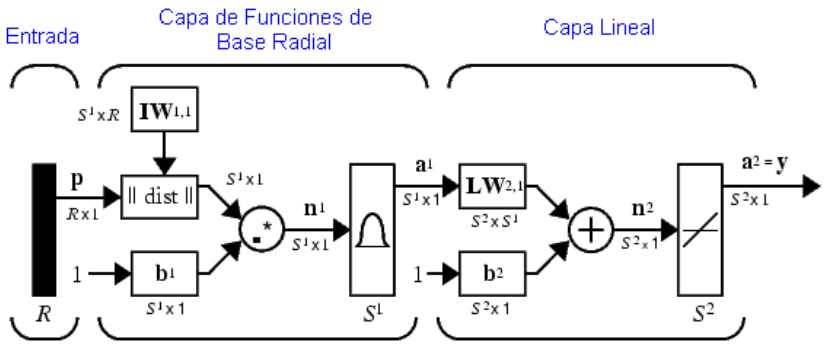


Figura 9. Arquitectura de una red de funciones de base radial **[10]**.

La forma de las redes de funciones de base radial puede ser generalizada para matrices de covarianza arbitrarias. Al hacer esto, la función gaussiana toma la forma

$$\phi_j = \exp\left\{\frac{-1}{2}\left(x-\mu_j\right)^T \Sigma_j^{-1}\left(x-\mu_j\right)\right\} \qquad \textbf{(1.22)}$$

Un aspecto fundamental para este tipo de redes es la diferencia que existe entre los pesos de la capa de funciones de base radial y los pesos de la capa lineal. Esta diferencia radica en el hecho que los pesos de la primera capa son encontrados mediante un proceso de entrenamiento no supervisado que no es óptimo, pero que en general funciona bien. En este primera parte del proceso de aprendizaje se usa el conjunto de los datos de entrada con el fin de determinar los parámetros que definen las funciones de base radial, tratando de modelar la distribución de los datos de entrada. A continuación se procede a encontrar los pesos de la segunda capa (capa lineal). (1.23) puede ser escrita de la forma

$$y_k(x) = \sum_{j=0}^{M} w_{kj}\phi_j(x) \qquad \textbf{(1.23)}$$

En notación matricial puede escribirse

$$y(x) = W\phi \tag{1.24}$$

Los pesos pueden ser optimizados mediante la búsqueda del mínimo de una función de error de suma de cuadrados de la forma

$$E = \frac{1}{2}\sum_{n}\sum_{k}\left\{y_k\left(x^n\right) - t_k^n\right\}^2 \tag{1.25}$$

En donde t_n^k es la etiqueta para la salida k cuando la entrada a la red es el vector x^n. Los pesos son determinados mediante el conjunto de ecuaciones lineales dado por

$$\Phi^T \Phi W^T = \Phi^T T \tag{1.26}$$

La solución formal para los pesos será

$$W^T = \Phi^{\dagger} T \tag{1.27}$$

En donde la notación $\Phi^{\dagger}$ representa la pseudo-inversa de Φ **[6]**.

Con el fin de determinar los parámetros que definen a las funciones de base radial, se utilizan procedimientos conocidos como algoritmos de agrupación (clustering algorithms). El algoritmo de agrupación K-means desarrollado por Moody y Darken **[17]** es uno de los más usados y fue el que se utilizó en este proyecto. El pseudo – código de este algoritmo se encuentra explícito en el numeral 3.2.2.

1.2.3. Selección del orden del modelo utilizando Validación Cruzada.

Un proceso en el cual se busca encontrar el mínimo de alguna función de error no permite encontrar el tamaño óptimo de la Red Neuronal, debido a que son procesos independientes. Sin embargo, otro de los objetivos que se persiguen durante un proceso de entrenamiento es determinar el tamaño de la red que tenga el mejor comportamiento frente a entradas nuevas. Uno de los procedimientos más sencillos consiste en evaluar la red para conjuntos de datos que no fueron empleados durante la fase de entrenamiento. Para realizar esto se entrena una red mediante la minimización de alguna función de error y se emplea un conjunto de validación independiente para probar su desempeño **[6]**.

La técnica de S – Fold Cross Validation **[18]** consiste en dividir el conjunto de entrenamiento en un número total de S subconjuntos. Al realizar tal proceso, la red es entrenada usando S – 1 subconjuntos, de tal forma que su desempeño es validado haciendo uso del subconjunto que no fue empleado en el proceso de entrenamiento. Este proceso se repite para cada una de las S posibilidades, y el error definitivo es promediado sobre los S resultados obtenidos.

Este procedimiento tiene como ventajas su sencillez y el hecho que suele funcionar bien, mientras que sus principales desventajas son dos: El número de datos de entrenamiento se disminuye, y el entrenamiento debe realizarse S veces, lo cual puede ocasionar incrementos notables en el tiempo de procesamiento.

1.2.4. Boosting

Boosting es un método que intenta elevar el desempeño de cualquier algoritmo de aprendizaje **[19]**. Se busca tomar un clasificador débil que sea mejor que adivinar, es decir que su probabilidad de error sea mayor a 0.5 para ser mejorado arbitriamente casi a cualquier valor de exactitud. La idea principal de Boosting es combinar hipótesis simples de clasificación para formar una función con un rendimiento mucho mayor. El principio básico es la combinación lineal de clasificadores débiles para formar una hipótesis mejor, siempre y cuando se cuente con un conjunto de datos suficientemente grande. Es decir

$$h_f(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \qquad \textbf{(1.28)}$$

Donde α_t denota el coeficiente con el cual la hipótesis $h_t(x)$ es combinada y ambos parámetros son obtenidos durante el proceso de aprendizaje.

La tarea de la clasificación de parámetros usando Boosting es encontrar una hipótesis que, basada en el conjunto de entrenamiento, asigna una etiqueta a la salida. Se representa el conjunto de entrada como *X* y se denotan las posibles salidas de una hipótesis como *Y*. Es decir se estima una hipótesis cuya función es $h : X \rightarrow Y$, donde se usan las parejas de datos de entrada generados aleatoriamente a través de una distribución de probabilidades *P(x,y)* **[20]**.

Como se mencionó anteriormente, los datos de entrada son generados aleatoriamente a través de una distribución que en un principio es desconocida. Entonces se debe tratar de estimar una función que esté cerca de la óptima basada en la información disponible dentro

del conjunto de entrenamiento para cada una de las hipótesis. Para diseñar esta distribución se inicia para el primer entrenamiento con una distribución uniforme, lo que quiere decir que cualquiera de las muestras del conjunto de entrada se puede elegir con la misma probabilidad para el entrenamiento. Después de cada iteración se realiza una modificación de dicha función de probabilidad, basada en los datos obtenidos y se le asigna una mayor probabilidad a las muestras mal clasificadas, entrenando de nuevo.

AdaBoost es un algoritmo adaptativo de Boosting, y se implementa así: Para cualquier época del entrenamiento se tiene una distribución **d** asignada al conjunto de datos en dicha época, y con base en ésta, se construye un clasificador débil *h(x)*. Estos pesos son actualizados en cada época de acuerdo al error en el cual incurre el clasificador débil, definiendo el error en cada época como **[19]**

$$\varepsilon_t(h_t, d^{(t)}) = \sum_{n=1}^{N} d_n^{(t)} \left(y_n \neq h_t(x_n) \right) \qquad \textbf{(1.29)}$$

Donde N es el número de elementos del conjunto de entrenamiento. Con la hipótesis *h(x)* se calcula el valor de su peso, es decir de la importancia que va a tener este clasificador débil. Se puede inferir que cuanto mayor porcentaje de error haya obtenido un clasificador débil, menor va a ser la credibilidad y menor va a ser su peso dentro de la hipótesis final, este valor α_t es

$$\alpha_t = \frac{1}{2} \log \frac{1-\varepsilon_t}{\varepsilon_t} \qquad \textbf{(1.30)}$$

Donde el valor del error debe ser menor a ½ con el objetivo de tener valores α_t positivos. Una vez se ha calculado el valor α_t, la distribución de probabilidad se actualiza de la siguiente forma **[19]**

$$d_n^{t+1} = \frac{d_n^t e^{-\alpha_t y_n h_t(x_n)}}{z_t} \qquad \textbf{(1.31)}$$

Esto quiere decir que la probabilidad del n-ésimo elemento del conjunto de entrenamiento para la época t+1 va a tener una probabilidad que está dada por la probabilidad para él mismo en la época anterior, la credibilidad de dicho clasificador y si su salida fue correcta o no. La distribución para el tiempo t+1 es tal que el clasificador previo tiene error previo ½. En caso de que haya sido correcto, su probabilidad disminuye con el fin de mostrarle a la red los elementos del conjunto de entrenamiento que no ha podido clasificar correctamente. La constante z_t es de normalización con el fin que $\sum_{n=1}^{N} d_n^{t+1} = 1$.

La salida definitiva o hipótesis final, es una combinación lineal de los diferentes clasificadores débiles, así

$$h_F(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \qquad \textbf{(1.32)}$$

1.2.4.1. Convergencia del error de entrenamiento a cero

Si se presentan las condiciones apropiadas para garantizar que el error de clasificación sea menor a ½ para cada clasificador débil, es posible demostrar que está condición es suficiente para garantizar que el error de la hipótesis final tiende a cero. Esto se debe a que la salida es una combinación lineal de clasificadores débiles y a medida que el número de iteraciones aumenta, el error de entrenamiento baja hasta un punto en el cual se hace cero. Es posible que aunque el error en el conjunto de entrenamiento sea cero, el error de generalización siga disminuyendo con el entrenamiento, sin que la red genere sobreajuste a los parámetros y, por ende, una mala generalización **[19]**.

2. ESPECIFICACIONES

2.1. ARQUITECTURA DEL SISTEMA

El primer aspecto que debe asegurarse en un proceso de aprendizaje de máquina es contar con una base de datos de muestras que estén correctamente clasificadas, de acuerdo a las posibles etiquetas en las cuales serán categorizadas las entradas. Los resultados en la clasificación de muestras que no fueron usadas durante el proceso de entrenamiento serán mejores en cuanto el tamaño de dicha base de datos aumente. Una afirmación como la anterior sugeriría elevar el número de muestras de la base de datos tanto como sea posible. Sin embargo, la limitación está impuesta por el hecho que no se dispone de una capacidad computacional infinita.

Tal y como se mencionó anteriormente, la realización de un procesamiento previo de los parámetros, que permita descartar características que aporten poca información al sistema es el segundo aspecto a tener en cuenta, de tal forma que la clasificación sea basada en aspectos verdaderamente relevantes. Idealmente se busca que el sistema se concentre en aquellos parámetros que permiten a un ser humano discernir entre dos o más clases diferentes. Esto se conoce con el nombre de generalización, y es un objetivo fundamental en cualquier proceso que involucre “aprendizaje de máquina”.

La selección del tipo de clasificador que será usado es el tercer aspecto. Para el caso de este proyecto, se decidió emplear Redes Neuronales Artificiales para tal fin, basándose en la evidencia de trabajos previos que han sido realizados. La escogencia de la topología de la red fue en cambio, un proceso en el cual se evaluaron diferentes alternativas y se realizaron comparaciones con los resultados obtenidos en cada uno de los casos. Se procedió con el entrenamiento para cada una de las redes escogidas con el fin de poder obtener un sistema clasificador que sea capaz de recibir entradas no empleadas anteriormente y realizar su categorización con el porcentaje de éxito más elevado posible. El sistema completo se muestra en la *Figura 10*.

2.2. ESPECIFICACIONES DE LOS GÉNEROS MUSICALES

Como se menciona en un principio, la Red Neuronal se diseña para ser capaz de clasificar canciones en tres géneros: Merengue, Salsa y Vallenato. Son dos las razones que motivan la realización de este sistema: Primero, dentro de la consulta previa que se llevó a cabo acerca del estado del arte en este campo, jamás de encontró un sistema como éste. Los trabajos se han enfocado primordialmente hacia la realización de sistemas reconocedores de Rock, Pop, Trance, Country y otros géneros propios de otras culturas. Aquí es donde nace la segunda motivación: Se desea realizar un sistema que esté de acuerdo con la cultura de nuestro país, en donde la música tropical ocupa un lugar importante en cuanto a la cantidad adeptos.

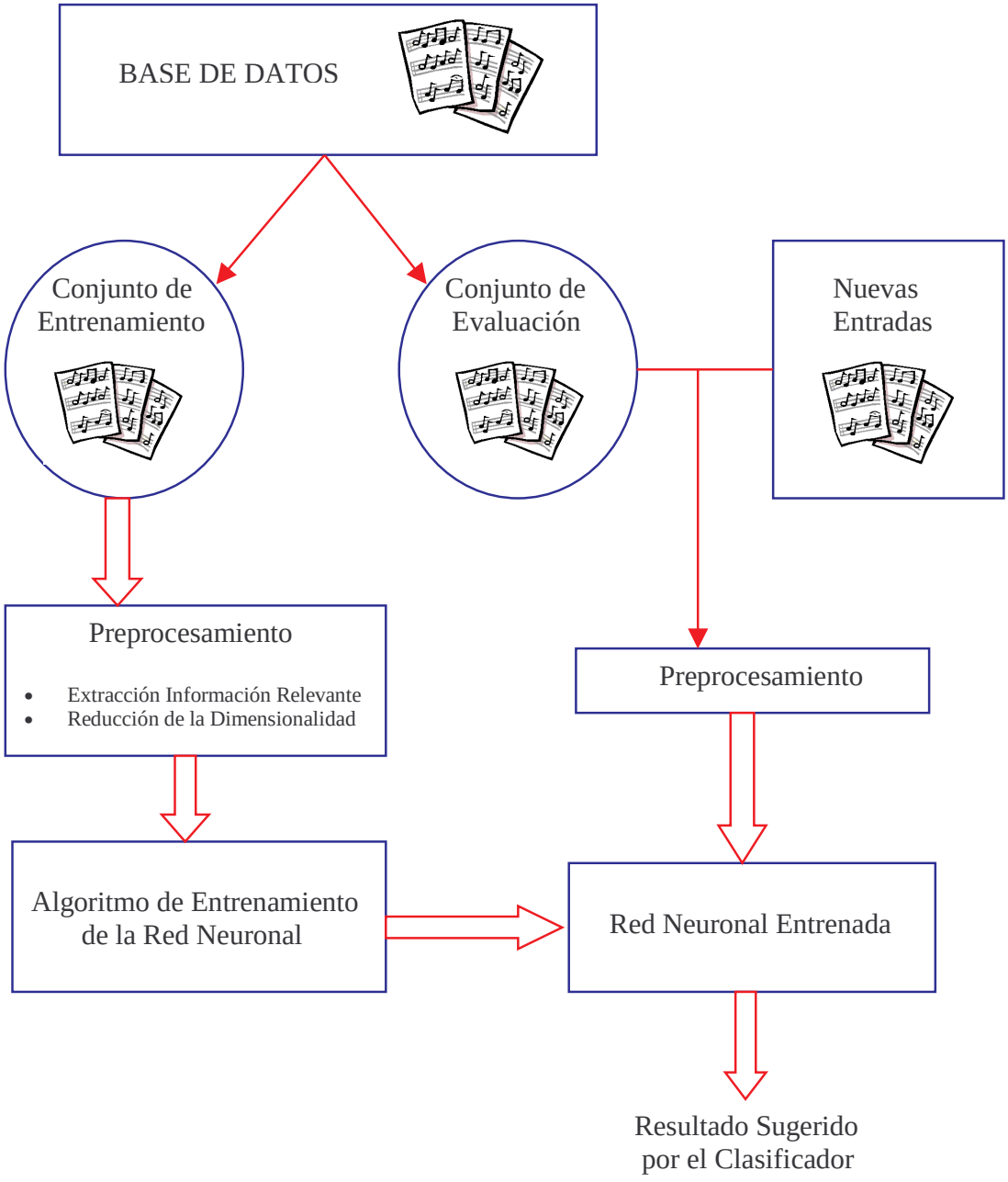


Figura 10. Esquema general del Clasificador de Música.

Una dificultad adicional a la realización del proyecto es el hecho que estos tres géneros musicales tienen raíces semejantes, razón por la cual la tarea de clasificación es un poco más compleja. A continuación se presentan las características fundamentales de cada uno de los géneros que forman parte de este estudio:

- Merengue: El merengue se desarrolló a mediados del siglo XVIII como una variante criolla del baile conocido como la contradanza. La dictadura de Rafael Trujillo en República Dominicana hacia 1930, marca los orígenes de este género y de todos los aspectos de la vida dominicana. Orquestas como la de Alberti, influidas por el jazz y arreglos de mambo, hacían hincapié en el uso de la trompeta y el saxofón, lo cual se hizo muy popular y ha prevalecido en el merengue de hoy en día. **[21]**
- Salsa: Los orígenes de la salsa se encuentran ritmos como la contradanza, la danza, el bolero y la guaracha. La salsa va desarrollándose gradualmente como un nuevo género que mezcla ritmos similares y su fuerza rítmica se fortalece más, al integrar instrumentos como el cuatro. Lentamente, la salsa se convirtió en un lenguaje musical que representa internacionalmente la voz de Latinoamérica, con variantes regionales como la salsa boricua, la salsa cubana, la salsa venezolana, la salsa de Cali y la salsa panameña. **[22]**
- Vallenato: El Vallenato tiene como antecesores, géneros provenientes de América, África y Europa. Su ritmo es generado mediante la utilización de instrumentos de fricción, de percusión y de viento, los cuales se unen para darle vida al vallenato. La guacharaca (instrumento típico de los indígenas americanos), la caja (tambor africano) y el acordeón (instrumento europeo adoptado en la costa atlántica colombiana) conforman la estructura musical de este género. **[23]**

2.3. ESPECIFICACIONES DEL FORMATO DE LAS CANCIONES

Actualmente es posible encontrar varios formatos digitales de audio. En los últimos años ha venido popularizándose el formato MP3 debido a la compresión que efectúa de archivos que antes tenían tamaños 10 veces superiores a los que ahora tienen. Esa es la razón fundamental por la cual grandes cantidades de música en formato digital son compartidas a través de Internet. Otro formato ampliamente difundido es el WAV. Sin embargo, este formato está siendo desplazado debido al tamaño requerido para su almacenamiento en memoria. Desde el punto de vista académico, la extracción de características a partir de un archivo WAV es factible porque contiene los valores de las amplitudes de una canción a través del tiempo. No sucede lo mismo con los archivos MP3, los cuales vienen codificados. El sistema es capaz de clasificar canciones en estos dos formatos, aunque se realiza una conversión a WAV cuando la entrada es un archivo en MP3, mediante la utilización de un algoritmo ideado por un conocido grupo de investigación dedicado a la descompresión de archivos en formato MP3 **[24]**. LAME es la herramienta empleada por programas que realizan conversión de formato, y su distribución a través de Internet es legal y gratuita.

Con el fin de asegurar una mínima calidad dentro de las canciones que componen la base de datos, se emplearon archivos estereofónicos con las siguientes características:

- Canciones en formato MP3: La calidad escogida fue 128 kbps, que corresponde a calidad de disco compacto.
- Canciones en formato WAV: La tasa de muestreo escogida fue 44.1 kHz.

2.4. CARACTERÍSTICAS DE LA BASE DE DATOS

La base de datos con la cual se trabajó consta de 500 canciones, repartidas de la siguiente forma:

- 170 merengues.
- 177 salsas.
- 163 vallenatos.

Este conjunto de dividió en 2 partes con el fin de poder realizar el entrenamiento y la posterior evaluación de la Red Neuronal.

2.4.1. Conjunto de Entrenamiento

Como se mencionó anteriormente, el conjunto de entrenamiento debe ser tan grande como sea posible. La limitación está impuesta básicamente por el tiempo de procesamiento requerido para entrenar una Red Neuronal.

En este caso, se decidió tomar un 90% del número de canciones de la base de datos para formar el conjunto de entrenamiento compuesto por 450 canciones, así:

- 143 merengues.
- 160 salsas.
- 147 vallenatos.

2.4.2. Conjunto de Evaluación

El 10 % restante de la base de datos se utilizó exclusivamente para poder medir el desempeño de la Red, una vez había sido entrenada. Este conjunto está compuesto de 50 canciones distribuidas así:

- 17 merengues
- 17 salsas
- 16 vallenatos

2.5. IMPLEMENTACIÓN

El código necesario para realizar los procesos de extracción de características, preprocesamiento, entrenamiento de cada una de las topologías de red empleadas y evaluación de la mismas se realizó mediante programas escritos en Matlab[1].

Sin embargo, es preciso aclarar que absolutamente todo el código necesario fue implementado sin hacer uso de las funciones del toolbox de Redes Neuronales. Este toolbox presenta opciones excelentes en todo lo relacionado al algoritmo de entrenamiento para redes de propagación inversa, pero tiene muy poca flexibilidad en cuanto a redes de funciones de base radial y no tiene nada relacionado con Boosting. Sin embargo, no se empleó ninguna función del toolbox, porque uno de los objetivos del proyecto era hacer

[1] Matlab. The MathWorks, Inc. Versión 6.1.0.450. Release 12.1.

una comparación entre estos tres tipos de redes bajo las mismas condiciones de prueba para poder comparar los tiempos de procesamiento.

Los algoritmos de entrenamiento se ejecutaron en computadores Dell con las siguientes especificaciones:

- Procesador Pentium 4 de 1.8 GHz.
- 512 MBytes de memoria RAM.
- 20 GBytes de disco duro.

3. DESARROLLOS

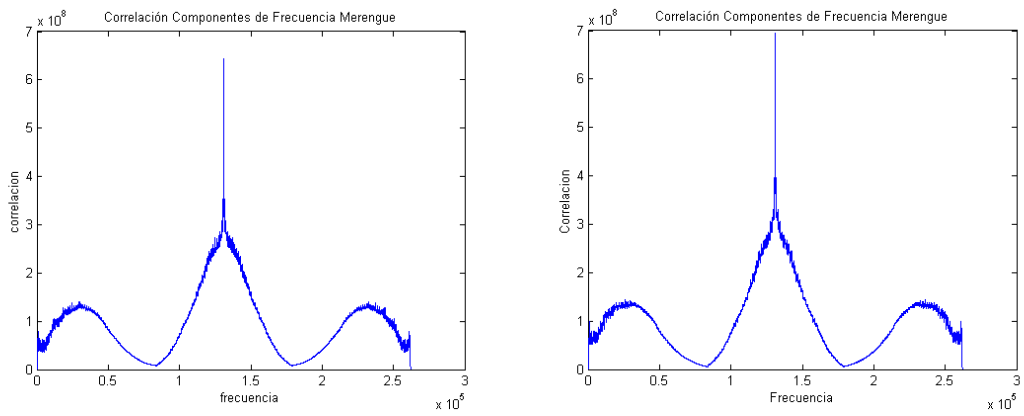
3.1. EXTRACCIÓN DE PARÁMETROS

Las muestras que se toman a partir de las canciones en formato WAV tienen una duración de 1.4861 segundos a partir de canciones cuya tasa de muestreo es 44.1 kHz. Esto significa que en un segundo habrá 44100 valores discretos (si la canción es monofónica), de forma tal que se toman $2^{16} = 65536$ muestras para la extracción de parámetros.

3.1.1. Análisis en tiempo

La autocorrelación es una medida estadística de la similitud de una señal con ella misma tiempo después. Calculando la correlación del espectro de frecuencia con diferentes ventanas de tiempo se obtiene una serie de valores que generan una curva de autocorrelación. Cuando la señal contiene un componente periódico con período P ocurre un pico en la curva en dicho valor. Como la señal está variando con el tiempo, se realiza una toma de muestras de la señal para comparar dichas ventanas de tiempo. Entonces la función de autocorrelación en un tiempo *t* es la autocorrelación del espectro de frecuencia en la ventana que termina en *t*. El tamaño de la ventana depende del retraso, es decir, un cambio en el nivel de un componente de período pequeño va a pasar desapercibido desde que haya otros períodos con mayor ocurrencia contenidos en la ventana.

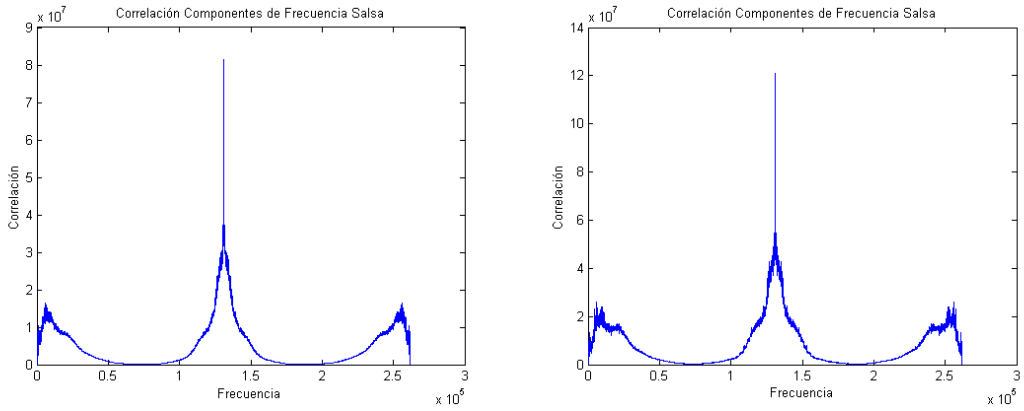
Para que los parámetros mostrados anteriormente sean de utilidad, es necesario que permitan realizar una diferenciación básica de los diferentes ritmos y que mantengan cierto nivel de similitud en el transcurso del tiempo. Por eso se prueba este primer elemento para cada uno de los ritmos. A continuación se presentan las gráficas de cada uno de los ritmos en dos instantes de tiempo diferentes, en donde para calcular cada una de estas muestras se toma un fragmento y se divide en 256 muestras diferentes.



Gráfica 1. Correlación Componentes de Frecuencia para el Merengue "Pégame tu vicio " de Eddie Herrera para 2 tiempos diferentes.

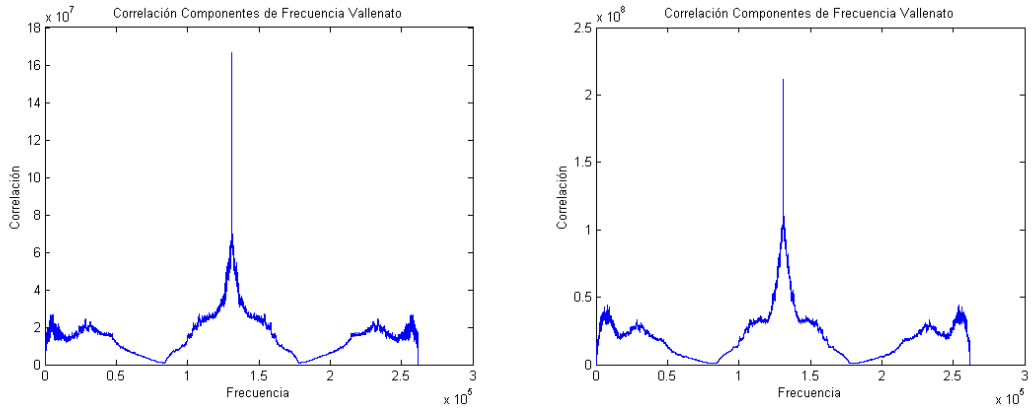
Las gráficas anteriores muestran el comportamiento de una canción perteneciente al conjunto de merengue.

Al observarse las dos gráficas, para los dos diferentes valores de tiempo, se observa que se cumple la primera condición para los parámetros a utilizar, pues no existen variaciones bruscas con el cambio de tiempo del cual se obtiene la muestra de la canción.



Gráfica 2. Correlación Componentes de Frecuencia para la salsa "Cali Ají" de Grupo Niche para 2 tiempos.

Las diferencias existentes entre las gráficas de merengue y de salsa se deben a que la gran mayoría de las muestras de merengue tienen contenido en alta frecuencia, debido a la existencia de instrumentos de viento como la trompeta y el saxo alto, mientras que en la salsa, el ritmo lo lleva la percusión y el bajo, por lo cual los valores más altos de autocorrelación se encuentran en las bajas frecuencias.



Gráfica 3. Correlación Componentes de Frecuencia para Vallenato "La diosa Coronada" de Leandro Diaz para 2 tiempos diferentes.

Por la existencia de un instrumento como el acordeón, que lleva los bajos dentro de la canción y también lleva las frecuencias altas de la canción, se puede ver que las gráficas de vallenato tienen una distribución más uniforme. Una vez se ha visto que la autocorrelación para los componentes de frecuencia entre los tres ritmos es diferente, se obtienen como parámetros dos valores estadístico simples como son la media y la varianza, que permiten realizar algún tipo de distinción entre los tres ritmos.

Los volúmenes promedio y dinámico que se derivan a partir de (1.1) y (1.2) también son empleados como parámetros y se encuentran a partir de la muestra completa.

Otro de los parámetros extraídos fue usado en un estudio previo que realizó Golub **[2]** en un proyecto de naturaleza semejante a éste. Se denomina sonoridad (loudness) y está definido como

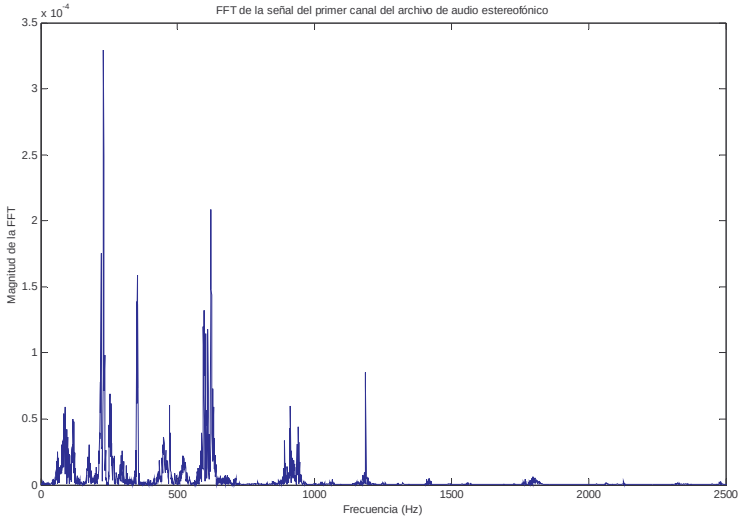
$$loudness = \log_2\left(1+\frac{1}{N}\sum_{t=1}^{N}|a_t|\right) \qquad \textbf{(3.1)}$$

Como se dijo anteriormente, manejar como entrada a algún algoritmo de aprendizaje la canción completa en formato digital traería como consecuencia tiempos de ejecución demasiado altos. Con el fin de evitar este problema se tomaron muestras cuya duración es 1.4864 segundos. La razón de la escogencia de este número de debe al hecho que algunos algoritmos que fueron empleados se encuentran optimizados para determinados valores, como se verá en el numeral 3.1.2.

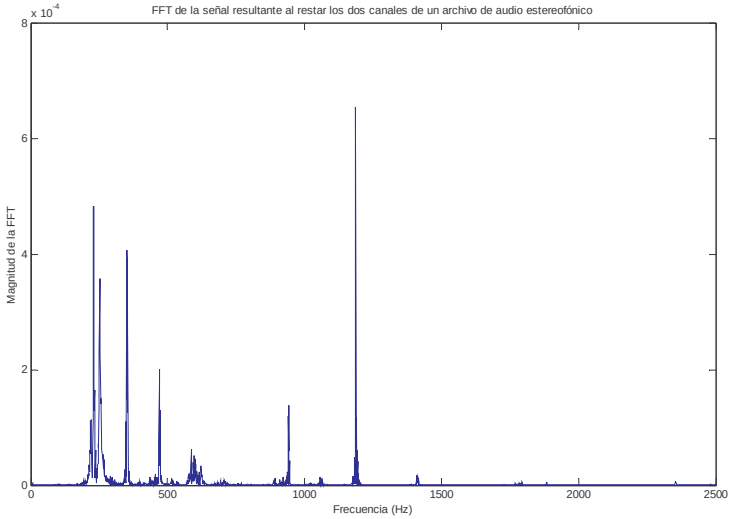
3.1.2. Análisis en frecuencia

La Transformada Rápida de Fourier (FFT) se encuentra optimizada para potencias de 2, y por esta razón se tomaron muestras de duración 1.4861 segundos (2^{16} = 65536 muestras) a partir de canciones muestreadas a 44.1 kHz.

A partir de la realización se numerosas pruebas se concluyó que lo más indicado es tomar como entrada al bloque de extracción de características el promedio de los dos canales del archivo estereofónico. Tomar la resta de los dos, o alguno de ellos resultó en la pérdida de algunas características relevantes. En las *gráficas 4* y *5* se ilustran, para una canción perteneciente al género merengue, las FFT del primer canal y de la resta de ambos canales respectivamente. Allí es claro que el tomar la resta resulta en una pérdida de información contenida mayormente en el rango que va desde 0 hasta 1000 Hz, que es donde se encuentran características tales como la percusión, que ayudan a diferenciar entre diferentes géneros. Mediante la realización de numerosas pruebas también se concluyó que es posible realizar una decimación de orden 2, es decir tomar sólo una de cada dos muestras, sin perder información relevante. Algunos de los parámetros que fueron usados en este proyecto han sido empleados en trabajos previos **[2][25]**.



Gráfica 4. Magnitud de la FFT del primer canal en función de la frecuencia para la canción "Nuestro Amor" de Eddie Herrera.



Gráfica 5. Magnitud de la FFT de la resta de los 2 canales en función de la frecuencia para la canción "Nuestro Amor" de Eddie Herrera.

Estos parámetros son: El centroide, el ancho de banda y el roll-off. El centroide fue explicado por Wold **[26]** como una medida del brillo de una canción y corresponde a una media ponderada por la magnitud de la transformada de Fourier, para todos los componentes de frecuencia de la señal. Matemáticamente

$$centroide = \frac{\sum_{f=1}^{N} f \cdot |H(f)|}{\sum_{f=1}^{N} |H(f)|} \qquad \textbf{(3.2)}$$

El ancho de banda es un parámetro importante que brinda una medida de la distribución de frecuencias y corresponde a una desviación estándar ponderada por la magnitud de la transformada de Fourier. Esto es

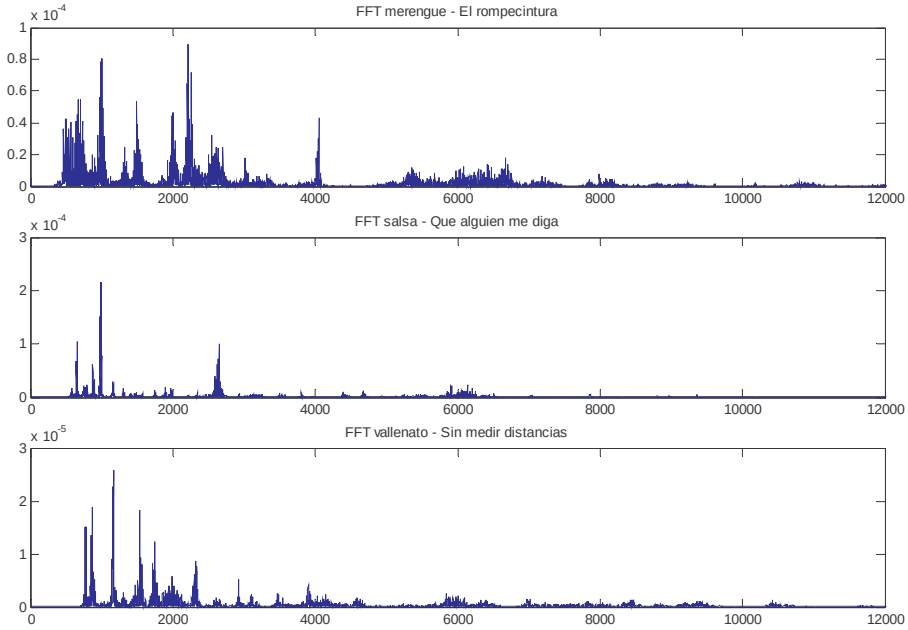
$$bandwidth = \sqrt{\frac{\sum_{f=1}^{N} (centroide - f) \cdot |H(f)|}{\sum_{f=1}^{N} |H(f)|}} \qquad \textbf{(3.3)}$$

El roll-off está relacionado con la forma espectral de una señal, y se define como el número R para el cual se cumple que

$$\sum_{f=1}^{R} |H(f)| = 0.85 \cdot \sum_{f=1}^{N} |H(f)| \qquad \textbf{(3.4)}$$

Hay otro par de parámetros que fueron extraídos a partir de la observación y estudio de las gráficas que se obtienen al calcular la FFT para canciones pertenecientes a los tres géneros que hacen parte del proyecto y que se explicarán a continuación. En la *gráfica 6* se muestra la FFT, que es obtenida típicamente para 3 canciones que pertenecen a géneros distintos. Puede constatarse que la canción de merengue es la que presenta un mayor contenido de armónicas tanto para bajas como para medias frecuencias. Por su parte, la canción de vallenato tiene un mayor contenido de armónicas que la canción de salsa, siendo esta última la que presenta contenidos espectrales más pequeños a lo largo de todo el espectro.

Por esta razón se extrajeron como características los promedios de los valores absolutos de la transformada de Fourier para frecuencias bajas y medias. Vale la pena enfatizar en el hecho de que estos parámetros surgieron de la observación de gráficas para pruebas que se llevaron a cabo con un número elevado de canciones de los tres géneros.



Gráfica 6. Magnitud de la FFT para tres canciones pertenecientes a distintos géneros.

3.1.3. Transformada Wavelet

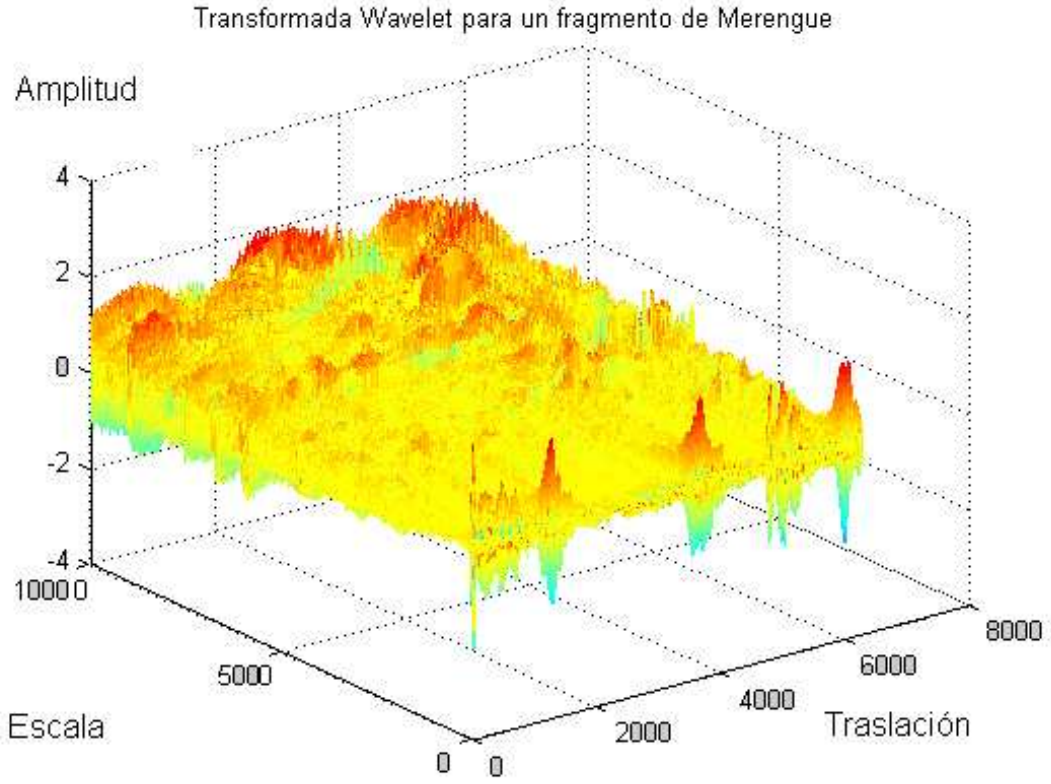
Para este grupo de parámetros se hace uso de los coeficientes bidimensionales $a_{\tau,s}$ de la transformada discreta Wavelet y como función madre, al sombrero mejicano, definido por

$$\psi(t) = \left(\frac{2}{\sqrt{3}} \pi^{-1/4} \right) \left(1 - x^2\right) e^{-x^2/2} \qquad \textbf{(3.5)}$$

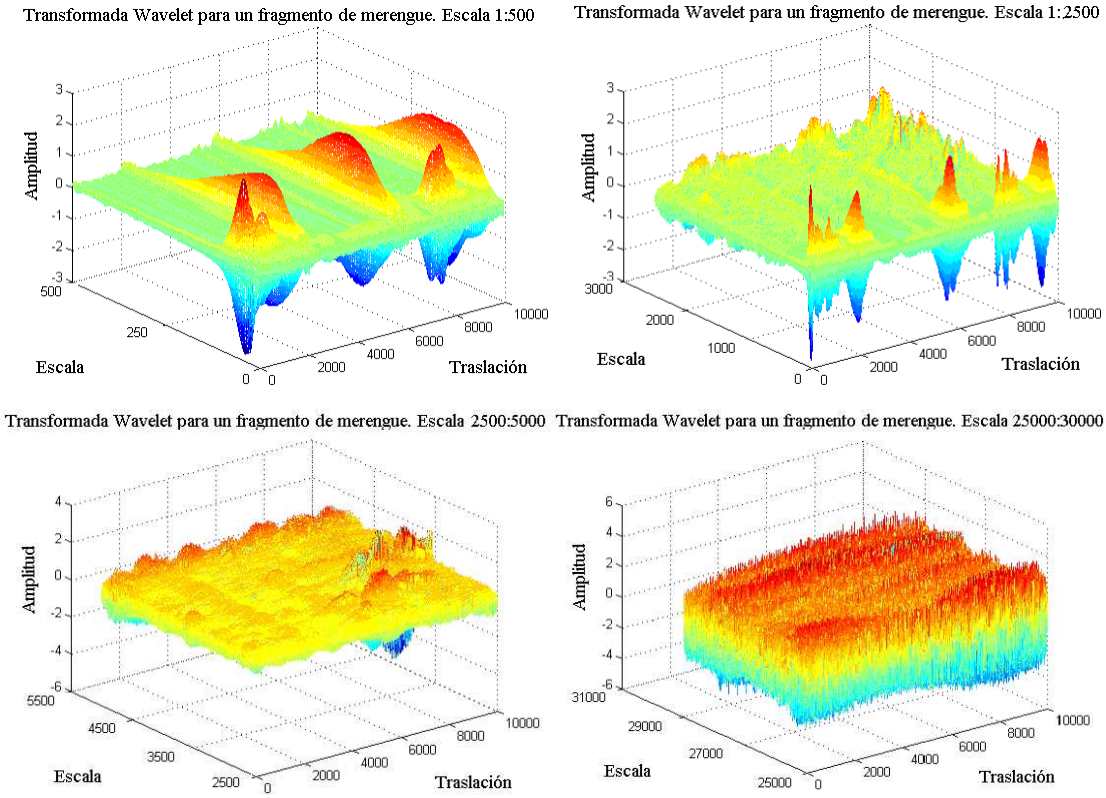
Como se enfatiza en el numeral 1.1.3., la ventaja de la transformada Wavelet sobre la trasformada de Fourier es el poder identificar la ocurrencia, en un intervalo de tiempo determinado, de diferentes intervalos de frecuencia. De esta forma se hace uso de esta propiedad para poder identificar diferentes aportes rítmicos de diversos instrumentos o grupos de estos. Es importante resaltar que, aunque esta separación no tiene en cuenta las características de timbre de los diferentes instrumentos, se logran extraer dos fuentes rítmicas audiblemente diferenciables y que logran marcar diferencia entre los géneros por clasificar.

Una razón adicional para trabajar con una muestra cuya longitud es la dicha en el numeral 3.1., es poder capturar durante más de un periodo los aportes rítmicos de interés en este estudio.

El proceso de búsqueda de características de periodicidad a lo largo del eje traslación para un intervalo determinado de escalas, llamadas singularidades, dentro de la transformada Wavelet, empieza por una búsqueda general en un intervalo muy amplio de escalas, específicamente entre 1 y 10000. En la *gráfica 7* se aprecia la transformada Wavelet para un fragmento típico de merengue, y en la *gráfica 8* se muestran detalles más específicos.



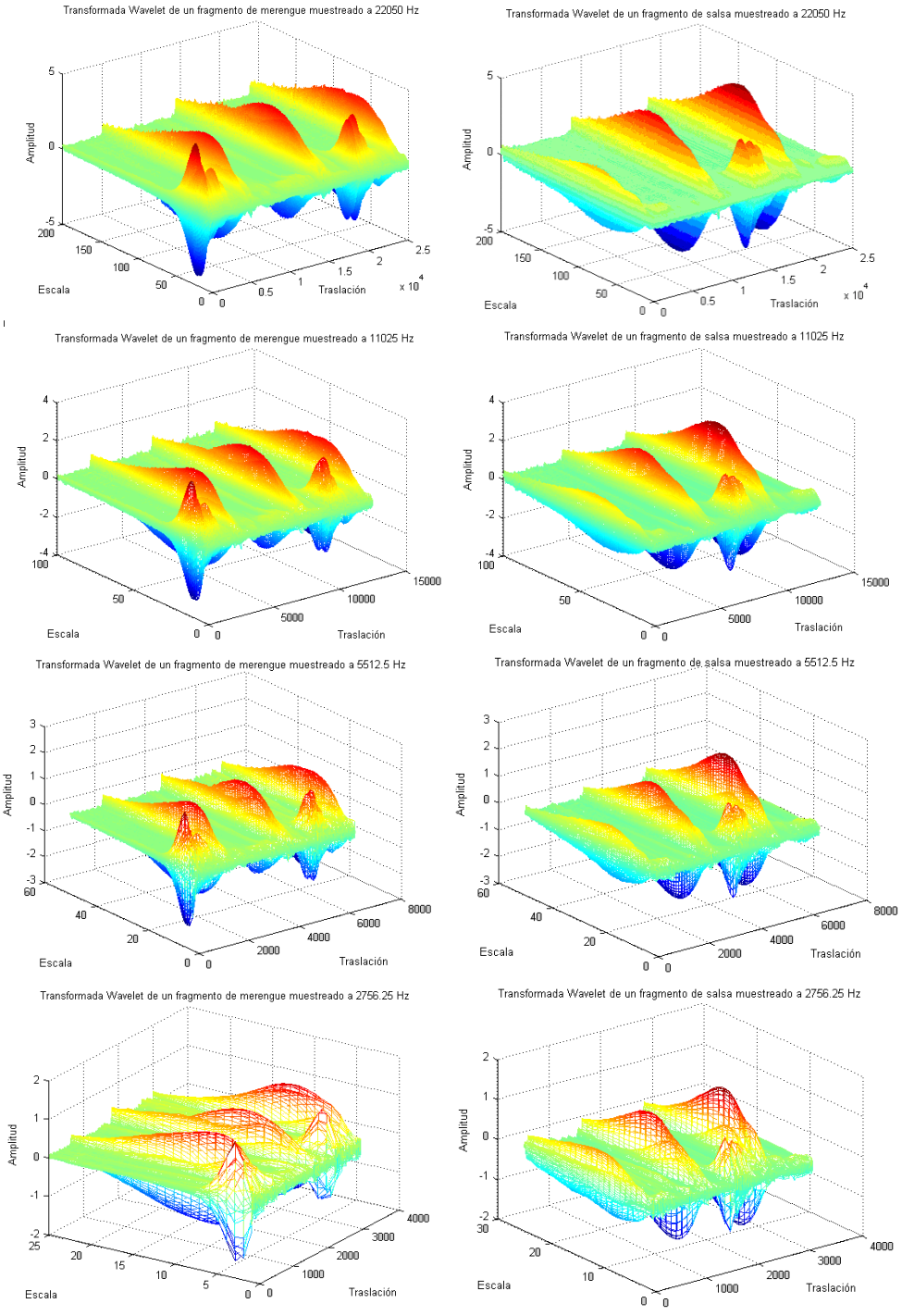
Gráfica 7. Transformada Wavelet del merengue “Nuestro Amor” de Eddie Herrera.



Gráfica 8. Detalles de la transformada Wavelet para un fragmento de merengue.

Basado en la observación de la transformada Wavelet de múltiples canciones de los géneros a clasificar, se encuentra que solamente existe periodicidad en el rango de escalas de [1,400], tal como se aprecia en la *gráfica 8.* Sin embargo, hasta este punto no se ha muestreado la señal, y los tiempos de procesamiento son muy altos, así que se hace necesario reducir el número de muestras de la señal, sin que esto implique reducir el rango de tiempo. Para esto se recurre al muestreo de la señal (decimación).

Aunque el muestrear la señal implica pérdida de información, no solo en la traslación sino también en la escala de la transformada Wavelet, se hicieron pruebas para ver hasta que



Gráfica 9. Transformada Wavelet para 2 canciones del los géneros merengue y salsa con diferentes tasas de muestreo.Columna Izquierda: Merengue. Columna Derecha: Salsa
a) 22050Hz b) 11025Hz c) 5512.5 Hz d) 2756.25 Hz

punto es posible muestrear sin que se pierdan las proporciones de las singularidades encontradas. En la *gráfica 9* se aprecia la transformada Wavelet de la misma señal a diferentes tasas de muestreo dentro del rango de escala mencionado anteriormente.

Tal como se ve en la *gráfica 9*, de las pruebas realizadas para diferentes canciones de los géneros a clasificar, es posible muestrear hasta 5512.5 Hz sin pérdida de las proporciones con respecto a la señal original. Así el número de muestras tomadas se reduce a 8192 (2^{13}), que así mismo cumple con ser potencia de 2 y minimiza el tiempo de procesamiento de este algoritmo.

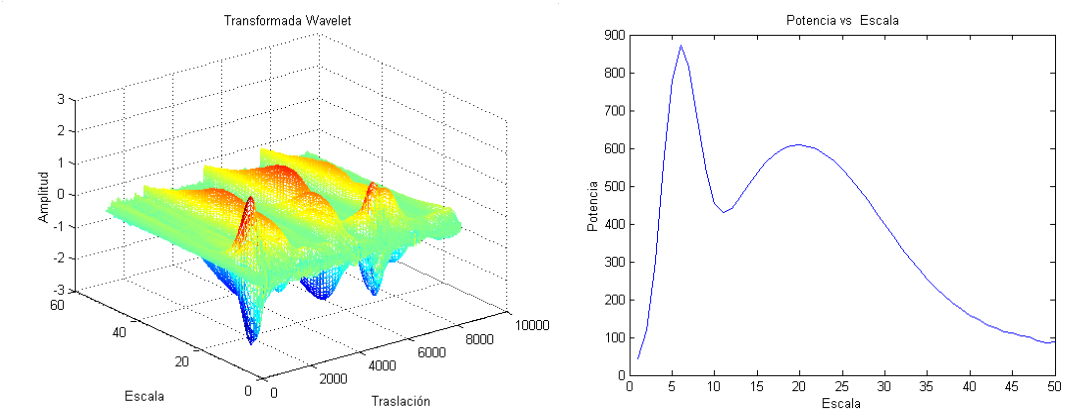
3.1.3.1. Energía por Escala

Con la transformada Wavelet se procede a hallar la energía como función de la escala de acuerdo al teorema de Parseval **[11]**.

$$E(s) = \sum_{\tau} a_{\tau,s} \qquad \textbf{(3.6)}$$

A continuación se presenta en la *gráfica 10.* $a_{\tau,s}$ y $E(s)$ para una canción dada.

Esta función de energía se caracteriza con su media y varianza, además es de interés para cuáles escalas $E(s)$ tiene máximos, así que se hallan los máximos de esta función y se escogen como parámetros los dos mayores.

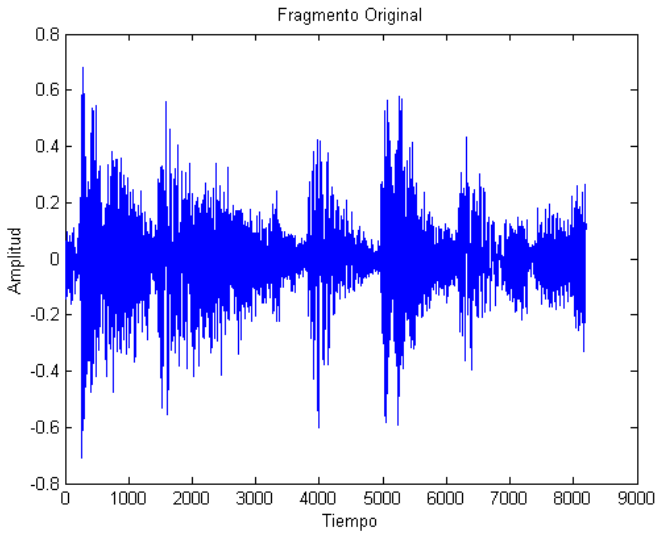


Gráfica 10. Transformada Wavelet y su respectiva Energía para cada escala.

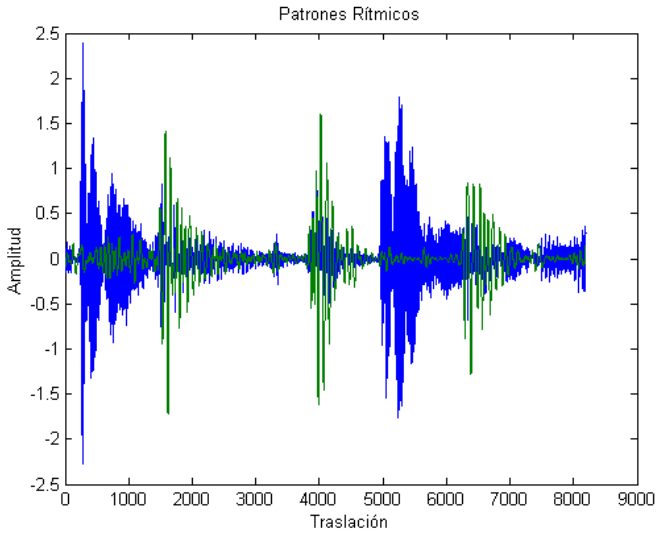
3.1.3.2. Patrones rítmicos

Los máximos hallados en la función energía permiten identificar cuáles escalas y, en consecuencia, cuáles frecuencias son las de mayor importancia a nivel rítmico, ya que una mayor energía implica que se escuchan con mayor intensidad, presentando particular interés para este estudio.

De esta forma, se escogen estas escalas y se selecciona todo el plano de traslación para dichas escalas, cada una de las cuales contiene un patrón rítmico que es audiblemente distinguible y que guarda estrecha relación con las diferentes familias de instrumentos y las voces presentes en dicho fragmento musical. Otra característica importante es que con estos dos patrones rítmicos aún se mantiene la envolvente de la onda sonora como se aprecia en las *gráficas 11* y *12*.



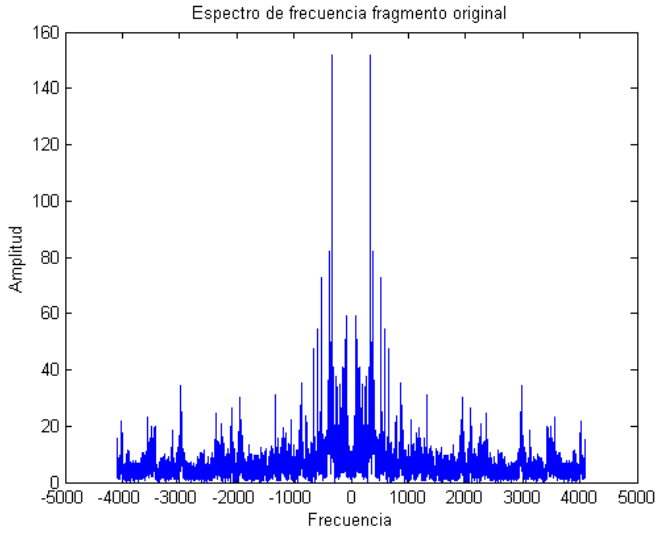
Gráfica 11. Fragmento original antes de procesar.



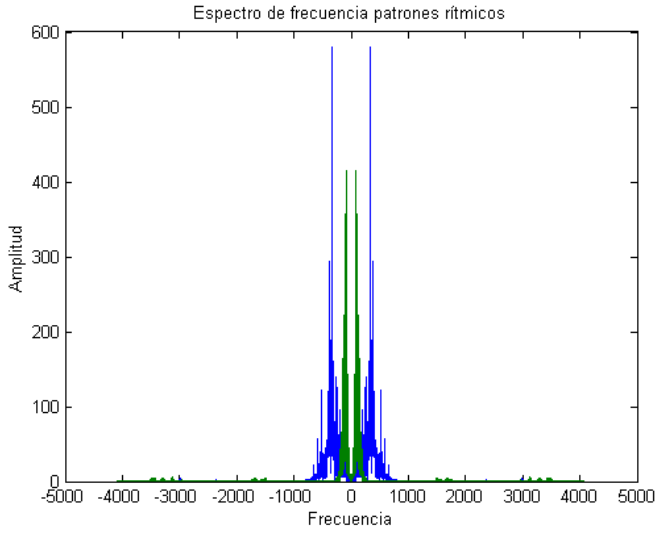
Gráfica 12. Patrones rítmicos extraídos.

Esto no solamente es comprobable por medio de la forma de onda sino también se percibe al escuchar los patrones y el fragmento original. También es muy importante resaltar que cada uno de estos patrones rítmicos es muy puro en frecuencia, es decir que existe un

componente en frecuencia muy relevante y con bajo contenido armónico. En las *gráficas 13* y *14* se presentan los espectros de frecuencia tanto del fragmento original como de los patrones rítmicos.



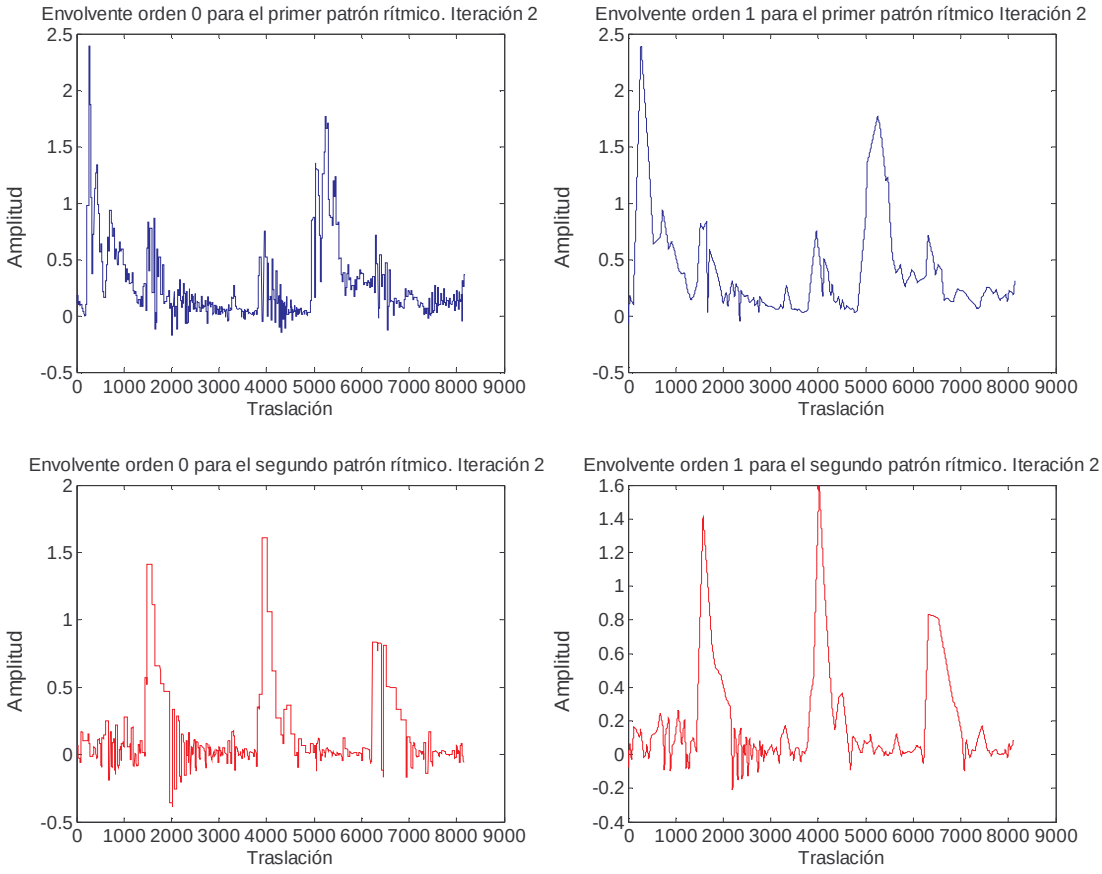
Gráfica 13. Espectro de frecuencia fragmento original.



Gráfica 14. Espectro de frecuencia patrones rítmicos.

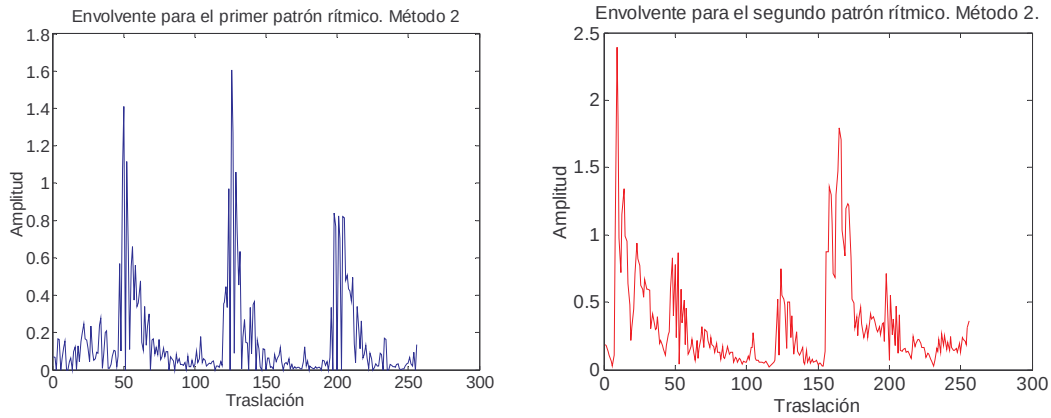
Esta frecuencia fundamental en cada patrón rítmico ya está representada en la escala que fue seleccionada de la función de energía, así que no se incluye como parámetro y se enfoca el estudio en la envolvente dichos patrones, pues allí es donde realmente se manifiesta la percepción del ritmo, pues esta envolvente guarda relación con el volumen.

Para la detección de la envolvente se implementaron dos métodos, el primero se basa en la detección de picos para luego reconstruir la envolvente utilizando aproximaciones de orden cero y orden uno para reconstruir la envolvente. Sin embargo es necesario aplicar este método varias veces para conseguir buenos resultados, como se muestran en la *gráfica 15.*



Gráfica 15. Envolventes de orden cero y uno para los patrones rítmicos.

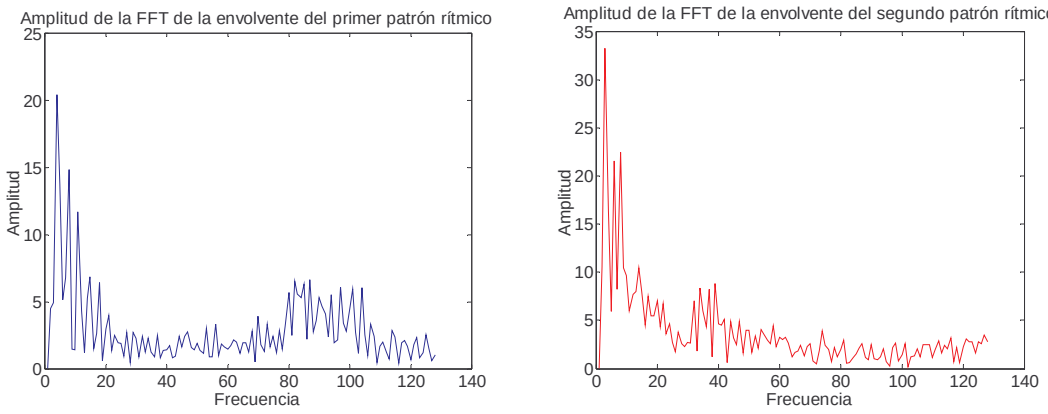
En el segundo método, se hallan máximos locales y consta de dos pasos. En el primer paso, se rectifica la señal y en el segundo se divide la señal en 256 intervalos. La envolvente se conforma a partir de los máximos de cada uno de estos intervalos. Este método además de no necesitar aplicarse varias veces, es mucho más sencillo y el tiempo de procesamiento es mucho menor, los resultados se aprecian en la *gráfica 16.*



Gráfica 16. Envolventes de los patrones rítmicos utilizando el método de máximos locales.

Debido a la simplicidad de este método y a los resultados similares con el primero se utiliza este segundo para la obtención de la envolvente de los patrones rítmicos. Estas funciones envolventes se caracterizan con su media y su varianza.

Ahora se realiza un análisis en frecuencia de cada envolvente utilizando la transformada rápida de Fourier (FFT), sin tener en cuenta su media y tomando solo una de las mitades debido a la simetría de esta transformada. Para encontrar la periodicidad de éstas, se utiliza la amplitud de la FFT (*gráfica 17*).

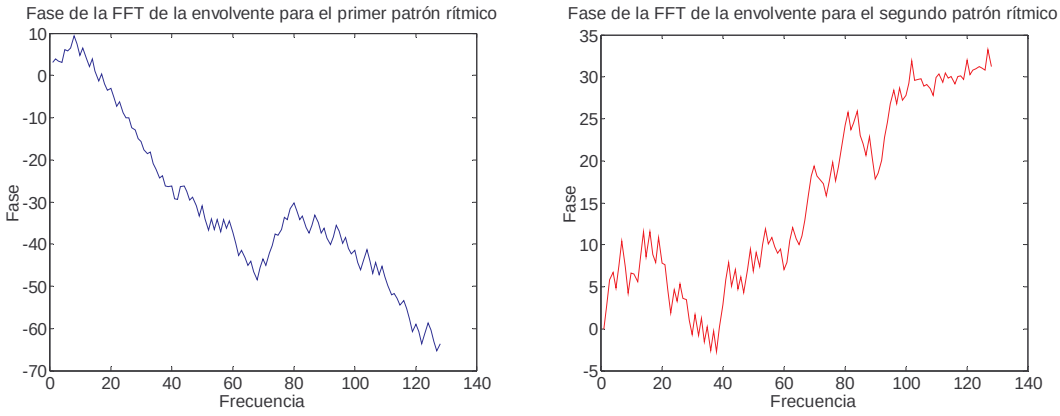


Gráfica 17. Amplitud de la FFT de la envolvente de los patrones rítmicos.

Puede concluirse de la *gráfica 17* que en este caso la envolvente del primer patrón rítmico, y en consecuencia el patrón rítmico, ocurren a una tasa mayor que el segundo patrón, pero el segundo tiene mayor relevancia debido a la mayor amplitud de su fundamental, considerando similares sus armónicas. Esto puede constatarse también con la *gráfica 12* y, adicionalmente de forma práctica, escuchando cada patrón rítmico independientemente.

De lo anterior se extraen como parámetros tanto la frecuencia fundamental de cada envolvente como su respectiva amplitud. Adicionalmente para tener en cuenta el contenido armónico, se utilizan como parámetros la media y la varianza de la FFT de las envolventes.

Por otra parte se analizan las fases de la FFT de las envolventes tal como se presentan en la *Gráfica 18.*



Gráfica 18. Fase de la FFT de la envolvente de los patrones rítmicos.

Estas funciones se caracterizan también con su media y su varianza, adicionalmente es interesante hallar una relación entre ellas, para este fin se toma como parámetro la covarianza entre ellas.

3.1.4. Resumen de parámetros extraídos

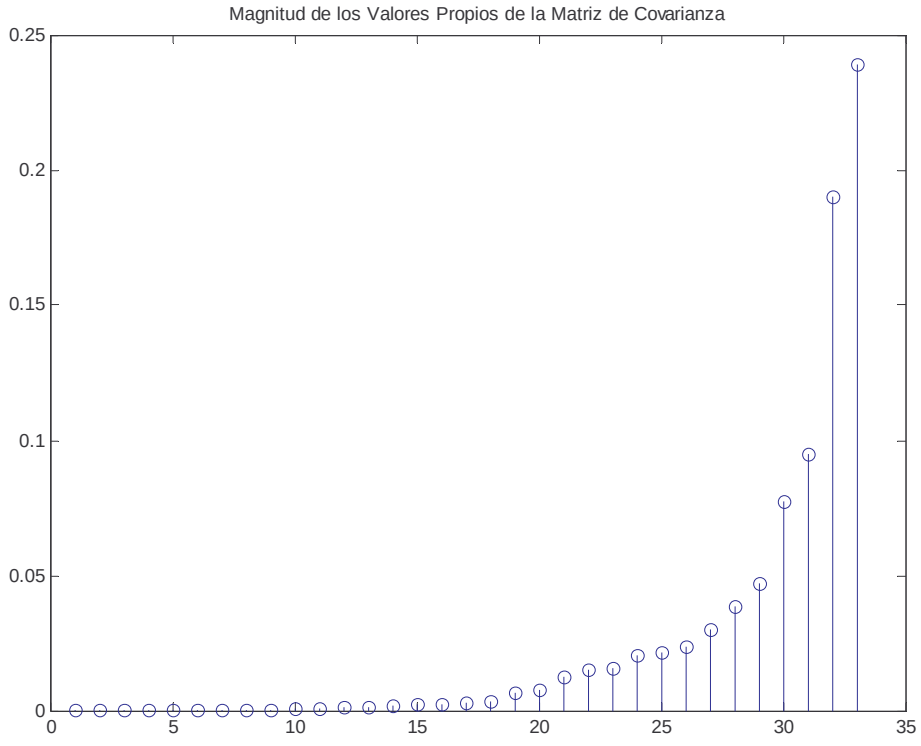
A continuación se presenta una tabla que resume los parámetros que fueron extraídos haciendo uso de las técnicas explicadas anteriormente

Parámetros de tiempo	Volumen promedio
	Volumen dinámico
	Media de la autocorrelación de las componentes de frecuencia.
	Desviación estándar de la autocorrelación de las componentes de frecuencia.
	Sonoridad
Parámetros de frecuencia.	Centroide
	Ancho de banda
	Roll-off
	Media de las componentes de frecuencia entre 0 y 2.5kHz
	Media de las componentes de frecuencia entre 2.5kHz y 10kHz
Parámetros de la Transformada Wavelet	Media de la función energía por escala.
	Varianza de la función energía por escala.
	Primer máximo de la función energía por escala.
	Segundo máximo de la función energía por escala.
	Escala primer máximo de la función energía por escala.
	Escala segundo máximo de la función energía por escala.
	Media de la envolvente del primer patrón rítmico.
	Varianza de la envolvente del primer patrón rítmico.
	Media de la envolvente del segundo patrón rítmico.
	Varianza de la envolvente del segundo patrón rítmico.
	Frecuencia fundamental de la envolvente del primer patrón rítmico.
	Amplitud de la frecuencia fundamental de la envolvente del primer patrón rítmico.
	Frecuencia fundamental de la envolvente del segundo patrón rítmico.
	Amplitud de la frecuencia fundamental de la envolvente del segundo patrón rítmico.
	Media de la Amplitud de las componentes de frecuencia de la envolvente del primer patrón rítmico.
	Varianza de la Amplitud de las componentes de frecuencia de la envolvente del primer patrón rítmico.
	Media de la Amplitud de las componentes de frecuencia de la envolvente del segundo patrón rítmico.
	Varianza de la Amplitud de las componentes de frecuencia de la envolvente del segundo patrón rítmico.
	Media de la Fase de las componentes de frecuencia de la envolvente del primer patrón rítmico.
	Varianza de la Fase de las componentes de frecuencia de la envolvente del primer patrón rítmico.
	Media de la Fase de la las componentes de frecuencia la envolvente del segundo patrón rítmico.
	Varianza de la Fase de la las componentes de frecuencia de la envolvente del segundo patrón rítmico.
	Covarianza de las Fases de las componentes de frecuencia de la envolvente de los patrones rítmicos.

Tabla 1. Resumen de parámetros.

3.1.5. Preprocesamiento

Un total de 33 parámetros son extraídos a partir de las canciones para el caso particular de este estudio. Aplicando el algoritmo de la Transformación de Karhunen – Loéve es posible reducir la dimensionalidad del espacio de entrada a un total de 20 parámetros. La razón de la escogencia de este número de características es debido a que los valores propios de la matriz de covarianza difieren por varios órdenes de magnitud. En la *gráfica 19* se muestran las magnitudes obtenidas para dichos valores y se observa claramente lo que se mencionó anteriormente.



Gráfica 19. Magnitud de los 33 valores propios de la matriz de covarianza.

Como criterio para seleccionar el número de parámetros que fueron usados como entrada a la Red Neuronal Artificial, se tomó el valor máximo de los valores propios y se descartaron todos aquellos que tenían una magnitud 100 veces menor que dicho valor. Esto dio como resultado una dimensionalidad de 20 para el espacio de entrada. Es importante resaltar que la forma ideal de determinar cuáles parámetros deben ser usados como entrada a una Red Neuronal, es usar diferentes subconjuntos como entrada y evaluar el desempeño general del sistema, escogiendo aquel para el cual se presente el porcentaje de clasificación superior. Sin embargo esto es demasiado ineficiente y es una de las razones por las cuales se utilizó la Transformación de Karhunen – Loéve.

3.2. REDES NEURONALES ARTIFICIALES

3.2.1. Redes de Propagación Inversa (Backpropagation)

El pseudo – código del algoritmo de entrenamiento de propagación inversa del error para una capa escondida a partir de la información adquirida es el siguiente

Escoja el número de neuronas de la capa oculta

Inicialice los pesos W.

Repita

Repita para cada uno de los datos X_i

Calcule la salida para la primera capa $G_j = f\left(\sum_{i=1}^{N} X_i W_{hji} + W_{hjB}\right)$

Calcule la salida total $F_j = f\left(\sum_{i=1}^{K} G_i W_{oji} + W_{ojB}\right)$

Fin.

Repita para cada uno de las salidas F_j *para todo j desde 1 hasta M*

Calcule el error como:

$$\varepsilon_{oj} = \left(Y_j - F_j\right)\left[F_j\left(1 - F_j\right)\right]$$

Fin

Hasta condición de terminación

La inicialización de pesos se hizo empleando el algoritmo Nguyen – Widrow **[10]** debido a que esta optimización permite la convergencia más rápida del entrenamiento, mejorando la inicialización de forma aleatoria implementada en un principio. Para la primera capa *H* la función de activación *f(x)*

$$f(x) = \frac{1}{1 + e^{-x}} \qquad \textbf{(3.7)}$$

La salida de la red entrenada se calcula utilizando la distancia mínima euclidiana entre *F* y los vectores correspondientes a las posibles etiquetas T_p.

$$H = T_p \quad tal\ que \left\|T_p - F\right\|^2\ es\ mínima \qquad \textbf{(3.8)}$$

Donde *p* es el número de etiquetas posibles y *H* es el vector de salida.

3.2.2. Redes de funciones de base radial

El pseudo – código del algoritmo de entrenamiento de una Red de Funciones de Base Radial puede verse a continuación:

Escoja el número de neuronas de la capa escondida (Capa de Neuronas con Función de Activación Radial)

Inicialice ω_i *(Centros de las Funciones de Base Radial).*

Repita

Para cada uno de los datos x_i

Asigne x_i *a* θ_i *(cluster) tal que* $\|x_i - \omega_i\|^2$ *sea la mínima entre las posibles.*

Fin.

Para cada uno de los clusters θ_i

$$\omega_i = \frac{1}{\|\theta_j\|} \sum_{j \in \theta_j} x_j$$

Fin.

Hasta que ninguno de los x_i *cambie de cluster.*

Encuentre la varianza de los datos mediante

$$\sigma_i^2 = \frac{1}{|\theta_j| - 1} \sum_{x \in \theta_j} (x - \omega_i)^T (x - \omega_i)$$

Encuentre los pesos de la capa de salida mediante

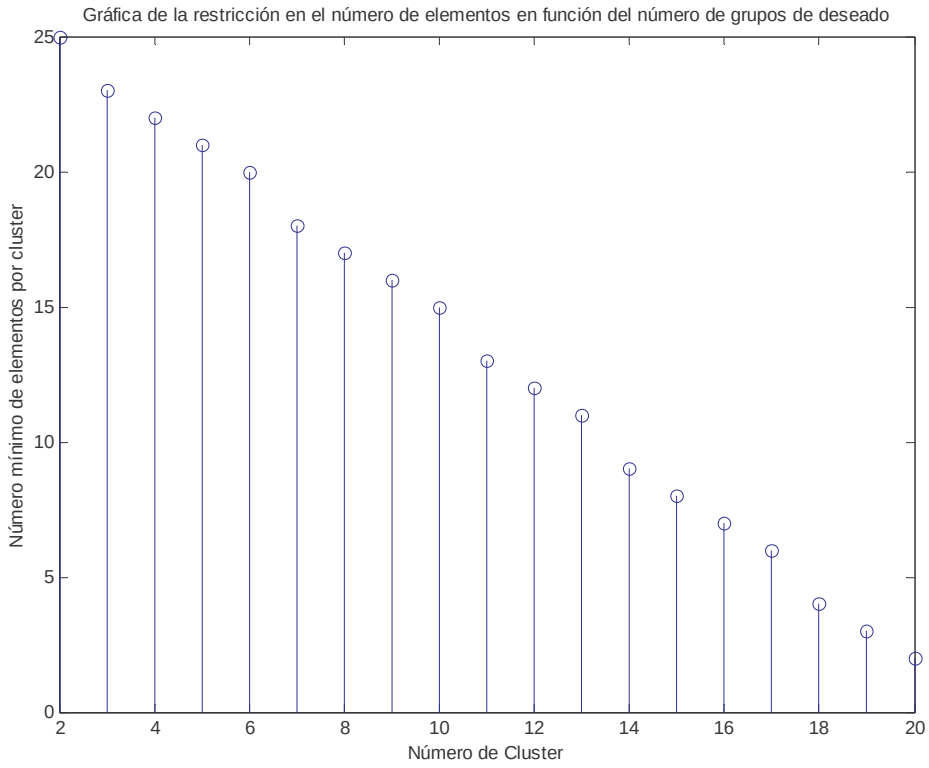
$$W^T = \Phi^{\dagger} T$$

En la implementación del algoritmo fue necesario tomar ciertas precauciones y realizar ciertas restricciones con el fin de garantizar el correcto funcionamiento del mismo. La primera parte del pseudo-código corresponde al algoritmo de agrupación K-means que fue mencionado en el numeral 1.2.2. Es posible ver que los clusters resultantes dependerán de la inicialización de los ω_i que se haga en el primer paso. Mediante la realización de numerosas pruebas fue posible verificar que hay ocasiones en las cuales resultan grupos que tienen 0 o 1 elementos. Estos resultados traen problemas en la determinación de los centros y las varianzas.

Así mismo se encontró que a medida que el número de neuronas de la capa de funciones de base radial sube (esto es, el número de clusters aumenta), el algoritmo necesita un mayor tiempo para poder llegar a la condición de finalización, que es cuando no se presentan variaciones. Con el fin de solucionar este inconveniente se impuso una condición sobre el número mínimo de elementos para cada uno de los clusters así: Si se desea agrupar en 2 conjuntos, el número de elementos por grupo puede ser muy grande, mientras que si el deseo es agrupar en 20 conjuntos, ese número mínimo de elementos debe ser pequeño.

La condición planteada se muestra en la *gráfica 20*, en donde se aprecia que el número de elementos mínimo del grupo disminuye a medida que el número de clusters aumenta. También fue posible observar que el porcentaje final de clasificación correcta para el conjunto de evaluación es superior al introducir este cambio. Esto de debe a que se está realizando una distribución de elementos más uniforme en cada uno de los grupos, mientras

que en un principio podían resultar grupos que tienen un alto número de elementos y otros con muy pocos.



Gráfica 20. Número mínimo de elementos por cluster en función el número de clusters.

Otro aspecto interesante durante este estudio fue observar el desempeño de la red en función del número de neuronas de la capa escondida para determinar el tamaño óptimo de la red . El número de neuronas de la capa de la salida es tres debido que se están clasificando canciones en ese mismo número de géneros.

3.2.3. Selección del orden del modelo

Con el objetivo de encontrar el orden del modelo adecuado para las redes de propagación inversa y de funciones de base radial se realizaron pruebas haciendo uso del proceso de 10 Fold Cross Validation **[6]**. En este proceso, el conjunto de entrenamiento se dividió en 10 subconjuntos de tal forma que la red es entrenada con 9 subconjuntos y validada con el conjunto restante. Este proceso se repite para cada una de las 10 posibles combinaciones. Al final, los errores del proceso de validación son promediados escogiendo el mejor.

3.2.3.1. 10–Fold Cross Validation para la red Backpropagation

Con el objetivo de buscar el orden óptimo de la capa oculta de la red de propagación inversa se decidió entrenar redes de orden 2 a 20. Para cada una de éstas se utilizaron los 10 subconjuntos de validación teniendo como parámetros una tasa de aprendizaje constante de 0.01 para todas las épocas y una condición de terminación con un error máximo de 0.001 en la componente más grande del gradiente o un número máximo de 10,000 iteraciones. Los valores de estos parámetros fueron resultado de pruebas realizadas previamente con el fin de optimizar el proceso de convergencia. Para cada subconjunto se escogió el mejor, de tal forma que para cada orden se escogieron 10 redes.

3.2.3.2. 10–Fold Cross Validation para la red de funciones de base radial

Se hicieron pruebas colocando desde 2 hasta 20 neuronas para la capa de neuronas cuya función de activación es del tipo radial. Como se mencionó en el numeral 3.2.2., los clusters dependen del la inicialización de los ω_i que se realice al principio del algoritmo. Por esta razón, y con el objetivo de encontrar la mejor red posible se decidió realizar un total de 250 iteraciones para cada uno de los 10 subconjuntos, guardando el resultado de aquella para la cual se obtenga un error de clasificación inferior.

3.2.4. Mejoras en el desempeño de los clasificadores

El conocimiento previo es un factor fundamental en cualquier proceso de clasificación. Este concepto se emplea con el fin de obtener desempeños superiores a aquellos que obtienen los clasificadores mediante un proceso de entrenamiento.

En el caso particular de este proyecto se extrajeron muestras de las canciones cuya duración fue mencionada anteriormente. Sin embargo las canciones de los géneros estudiados presentan irregularidades que se deben a arreglos musicales propios de cada intérprete. Estas irregularidades se traducen en espacios de silencio, improvisaciones, solos, cambios de velocidad y muchos otros. Por ejemplo, la inmensa mayoría de las canciones presentan introducciones que alteran el desempeño de un clasificador, por cuanto lo hacen concentrarse en patrones no relevantes para la clasificación.

Con el fin de compensar este característica se incorporó un sistema de votación 2 de 3, en el cual se evalúa la red en 3 fragmentos diferentes de la canción, reduciendo así la probabilidad de clasificación errónea.

3.2.5. Boosting

Adaboost se emplea como algoritmo de aprendizaje con el objetivo de disminuir el error de evaluación a medida que la red se sigue entrenando, incluso después de que el error de entrenamiento ha alcanzado el valor de cero **[27]**. Se escoge como clasificador débil una red de funciones de base radial debido a que los tiempos de procesamiento para las funciones de base radial son inferiores a aquellos obtenidos para el entrenamiento de una red de propagación inversa. El pseudo – código de Adaboost es el siguiente:

Se hace uniforme $D_t(i) = \frac{1}{m}$

Para t desde 1 hasta T

Se ejecuta el clasificador débil con entradas X, Y, D

Se calcula $\alpha_t \in R$

Se actualiza

$$D_{t+1}(i) = \frac{D_t(i) e^{(-\alpha_t \beta_t(i))}}{Z_t}$$

$$\textit{donde } \beta(i) \textit{ es un factor tal que } \beta(i) = \begin{cases} -1 & si\ H_t(i) = Y(i) \\ 1 & si\ H_t(i) \neq Y(i) \end{cases}$$

Fin

Se calcula la hipótesis final

$$H(i) = p \quad tal\ que \sum_{t=1}^{T} \alpha_t g_t(i) \ \ es\ máxima$$

Puede notarse que para la primera iteración que el algoritmo tenía igual probabilidad de equivocarse con cualquier canción, por lo cual la distribución inicial asigna igual probabilidad de escogencia a todas las muestras, esta probabilidad es modificada de acuerdo al resultado obtenido por la red al momento de clasificar una muestra, si esta es clasificada correctamente su probabilidad disminuye, como fue explicado anteriormente, si no, su probabilidad aumenta, a partir de esta distribución se obtiene un conjunto de muestras bootstrap, lo que se desea con este tipo de conjunto, es que cada clasificador débil se entrene con un conjunto de muestras aleatorio, pero haciendo mayor énfasis en las canciones que no han sido clasificadas correctamente con el fin de llevar a cero el error de entrenamiento.

Un ejemplo de la forma en la cual se modifican las probabilidades de selección de una canción durante la fase de entrenamiento dependiendo de si su clasificación fue correcta o no en la época anterior, se muestra en la *Figura 11.*

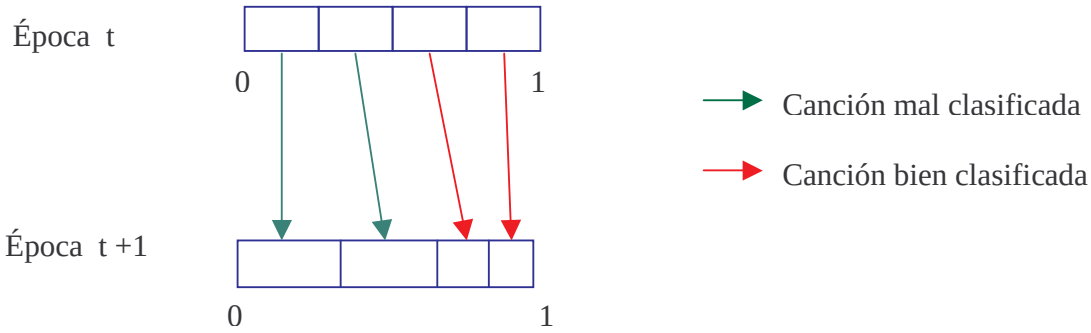


Figura 11. Probabilidad de selección en la muestra bootstrap.

En este caso se utilizó como clasificador débil una red de funciones de base radial de orden 11 puesto que presentaba una buena relación entre el tiempo de entrenamiento y el porcentaje correcto de clasificación. Una vez se ha entrenado un clasificador débil es necesario calcular su error para poder realizar los respectivos ajustes para la siguiente iteración. Este error es calculado utilizando como la distancia euclidiana entre todos los vectores de salida resultantes y la base de los vectores de salida (como se describe en el algoritmo de Radial Basis).

Con el valor del error se procede a calcular el coeficiente a que se define como:

$$\alpha_i = \frac{1}{2}\left(\log\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)\right) \qquad \textbf{(3.9)}$$

Donde a_i es el coeficiente para la iteración i y ε_i es el error de dicha iteración, este coeficiente entrega un grado de credibilidad para cada uno de los clasificadores débiles y además se utiliza para calcular la distribución de probabilidad.

Los clasificadores débiles obtenidos se utilizan para calcular el error de la hipótesis final.

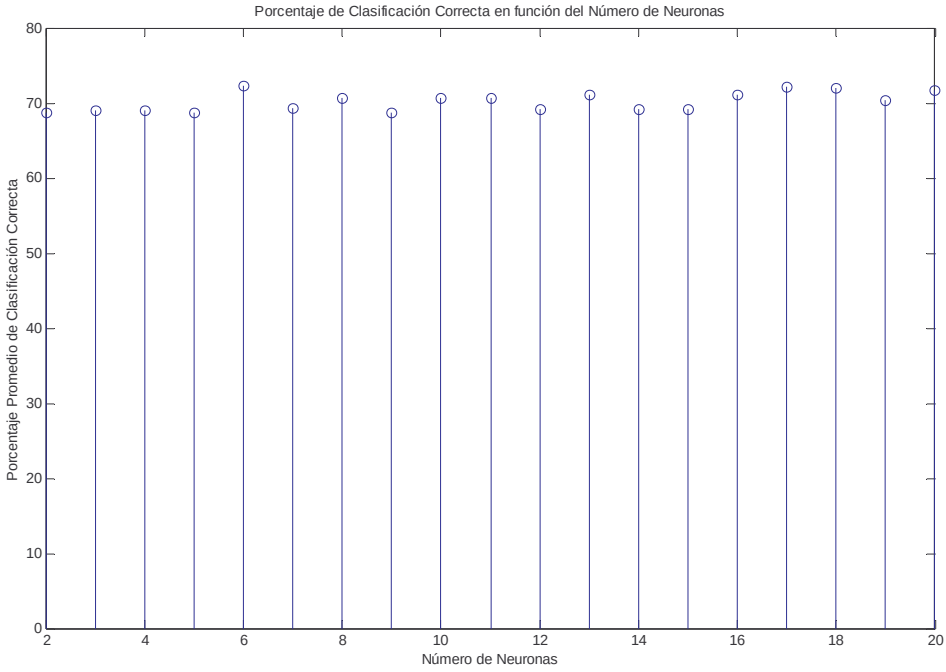
A partir de la suma de hipótesis parciales se genera la anterior hipótesis final, con la cual se calcula el error de entrenamiento y la distribución de márgenes para cada una de las clases *j*, en diferentes épocas de entrenamiento.

$$m_{ij} = y_{ij} H_{ij}(x_i) \qquad \textbf{(3.10)}$$

4. ANÁLISIS DE RESULTADOS

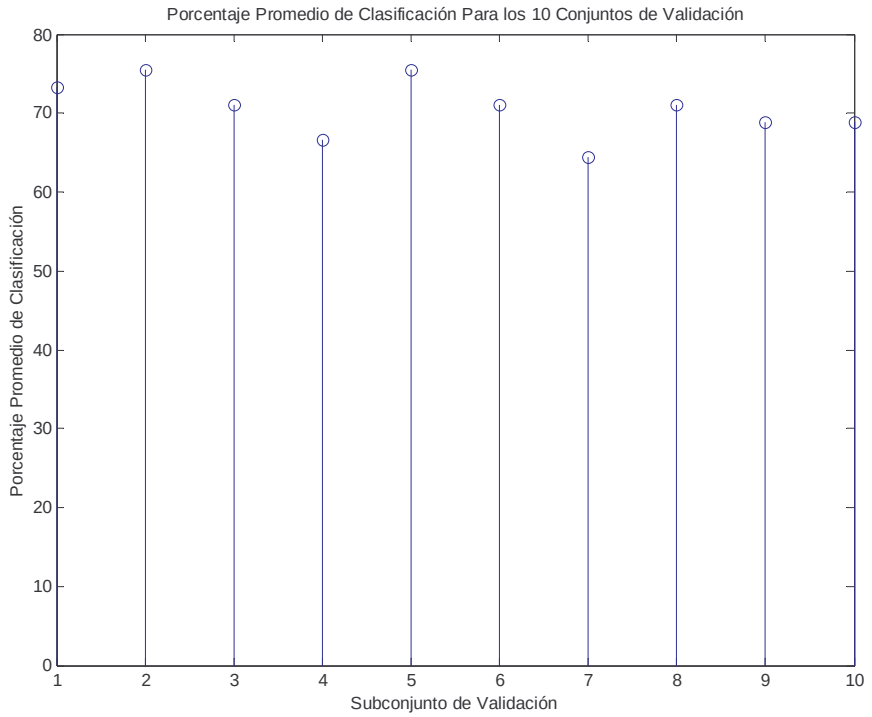
4.1. CLASIFICADOR RED DE PROPAGACIÓN INVERSA (BACKPROPAGATION)

Para realizar la 10–Fold Cross Validation, el conjunto de entrenamiento se dividió en 10 subconjuntos de forma tal que para cada iteración se contaba con 405 canciones para realizar el entrenamiento, y 45 para la validación. Los promedios de clasificación correcta que se obtuvieron para cada número de neuronas se muestran en la *gráfica 21*.



Gráfica 21. Porcentaje Promedio de Clasificación Correcta obtenido empleando 10 – Fold Cross Validation.

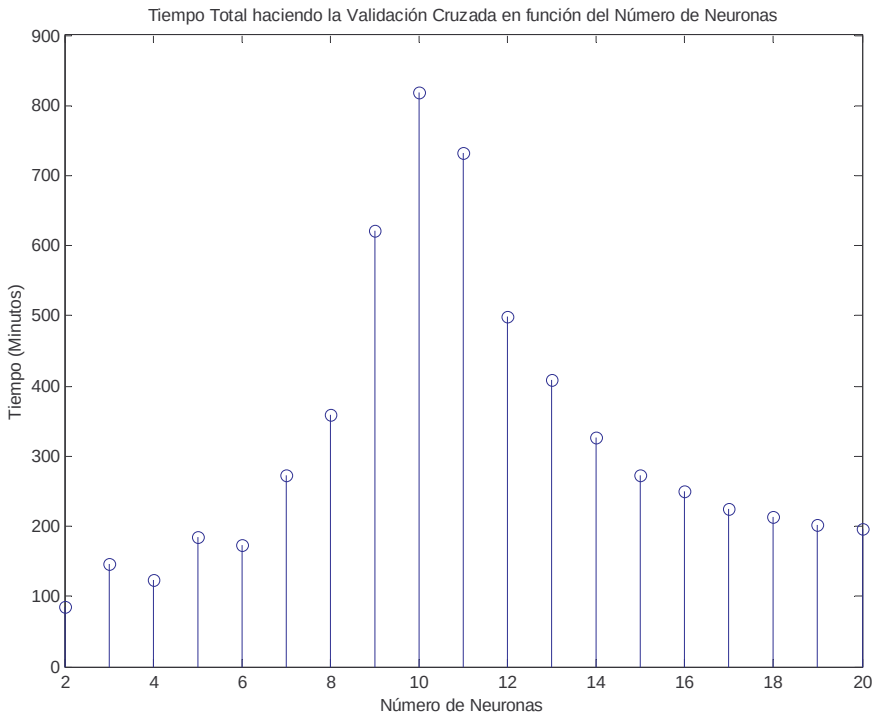
Allí se puede observar que el error promedio de clasificación para el conjunto de validación es bastante similar para los diferentes ordenes de la red, y no se presentan mejoras significativas al aumentar el número de neuronas de la capa oculta. El porcentaje promedio de clasificación correcta tiene su máximo para la red con 6 neuronas en la capa escondida, con un porcentaje de acierto del 72.25 %. Como ya se mencionó, esto es un promedio del resultado que se obtiene al realizar 10–Fold Cross Validation. En la *Gráfica 22* se muestran los promedios de clasificación correcta para cada uno de los 10 subconjuntos al realizar 10 – Fold Cross Validation para la red de 6 neuronas.



Gráfica 22. Porcentaje promedio de Clasificación correcta para los 10 subconjuntos de Validación con una red de 6 neuronas.

La observación de esta gráfica es interesante porque muestra que para la red de 6 neuronas se obtienen porcentajes de acierto superiores e inferiores al promedio mencionado anteriormente. Lo deseable sería que este porcentaje variara muy poco (idealmente no debería variar), pero como la validación depende de las canciones que sean evaluadas en cada instante, ese número cambia.

Otro factor importante en un estudio como el que se realizó es el relacionado con los tiempos de procesamiento necesarios para llevar a cabo el entrenamiento. En la *gráfica 23* se muestran los tiempos que se necesitaron para realizar 20 iteraciones con cada una de los 10 subconjuntos del proceso de validación, es decir que se muestra el tiempo empleado para realizar las 200 iteraciones correspondientes al proceso de Validación Cruzada con cada una de las redes consideradas. La gráfica muestra una tendencia creciente para redes que tengan entre 2 y 10 neuronas en la capa escondida. Se observa que el tiempo disminuye para redes que tienen entre 11 y 20 neuronas para la misma capa. La red cuyo entrenamiento tarda más es la que posee 10 neuronas con un tiempo total de 817.85 minutos.



Gráfica 23. Tiempo empleado en hacer 10 – Fold Cross Validation en función del número de neuronas de la capa oculta.

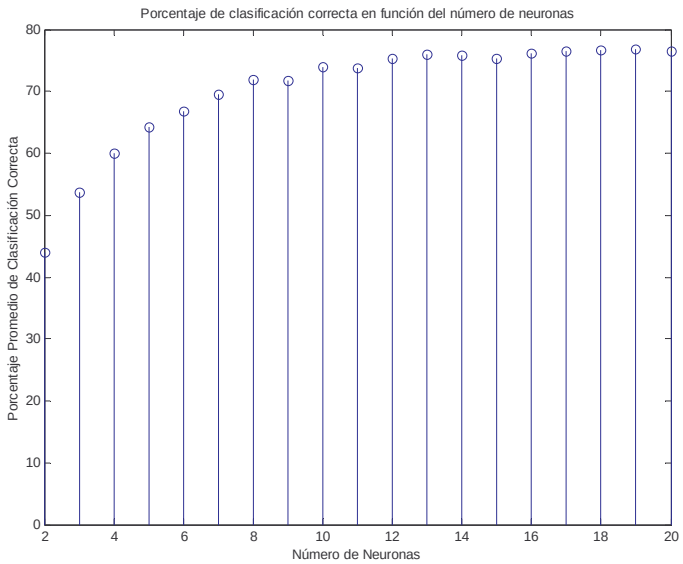
De acuerdo a la red que tuvo mayor porcentaje promedio de clasificación correcta para el conjunto de validación, se escoge la red de 6 neuronas En la *gráfica 22* se muestran los porcentajes de validación para cada uno de los subconjuntos y se aprecia que la mejor red tiene un porcentaje de validación del 75.56 %.

Haciendo uso del conjunto de evaluación es posible determinar el desempeño de la red. El porcentaje de acierto de la red escogida es del 76 % tomando el fragmento de la canción en 1 minuto.

Tal y como se había mencionado anteriormente es posible mejorar el desempeño de la red empleando un sistema de votación tomando 3 muestras para diferentes tiempos. Al realizar esto para fragmentos ubicados en 1, 1.25 y 1.5 minutos se obtiene un porcentaje de acierto de la red del 78% que es superior al obtenido previamente.

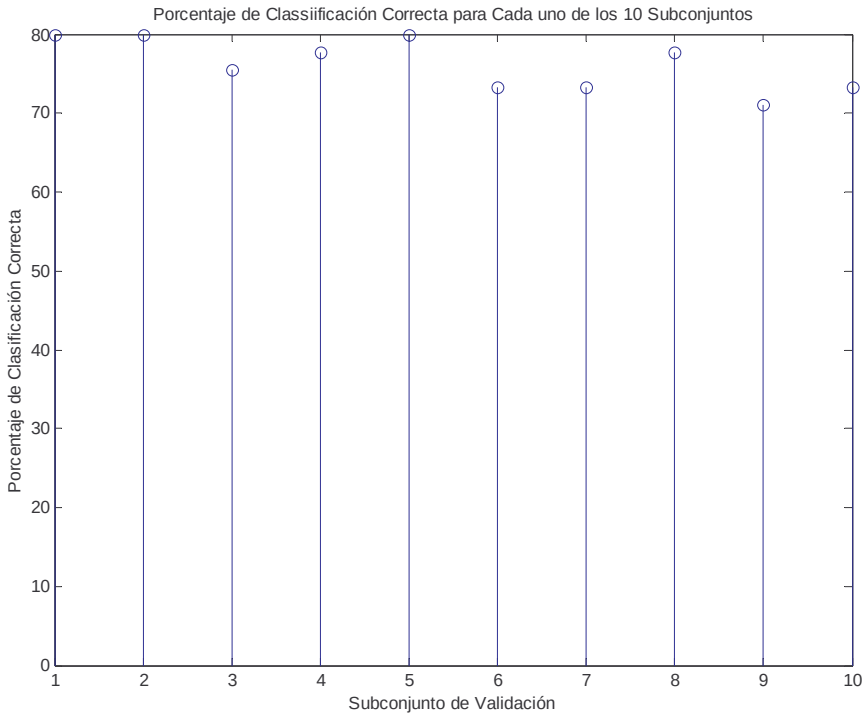
4.2 CLASIFICADOR RED DE FUNCIONES DE BASE RADIAL

Al igual que para la red de propagación inversa, se utilizó 10 – Fold Cross Validation dividiendo el conjunto de entrenamiento en el mismo número de subconjuntos, de forma tal que para cada iteración se contaba con 405 canciones para realizar el entrenamiento, y 45 para la validación. Los promedios de clasificación correcta que se obtuvieron para cada número de neuronas se muestran en la *gráfica 24*.



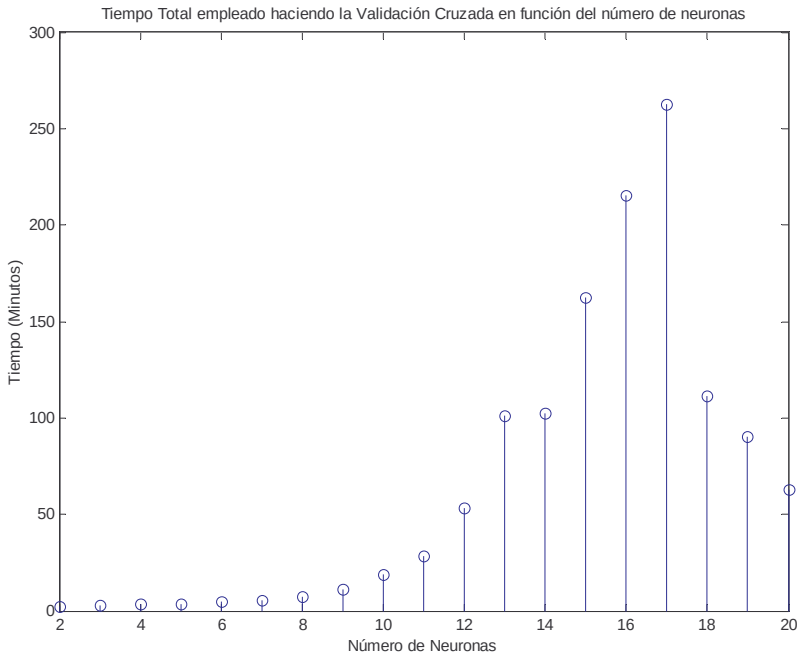
Gráfica 24. Porcentaje Promedio de Clasificación Correcta obtenido empleando Validación Cruzada.

Se ve que el porcentaje promedio de clasificación aumenta a medida que se agregan neuronas a la capa escondida para esta topología. El porcentaje promedio de clasificación correcta tiene su máximo para una red con 19 neuronas cuya función de activación esta dada por la función de base radial, con un porcentaje promedio de acierto del 76.89 %. Sin embargo, se observa que a partir de 16 neuronas, dicho porcentaje no presenta incrementos significativos. En la *Gráfica 25* se muestran los porcentajes máximos de clasificación correcta para cada uno de los 10 subconjuntos al realizar 10 – Fold Cross Validation, para la red de 16 neuronas.



Gráfica 25. Máximo Porcentaje de Clasificación correcta para los 10 subconjuntos de Validación con una red de 16 neuronas.

Otro factor importante en un estudio como el que se realizó es el relacionado con los tiempos de procesamiento necesarios para correr el algoritmo de entrenamiento. En la *gráfica 26* se muestran los tiempos que se necesitaron para realizar 250 iteraciones con cada una de los 10 subconjuntos del proceso de validación, es decir que se muestra el tiempo empleado para realizar las 2500 iteraciones correspondientes al proceso de 10 – Fold Cross Validation con cada una de las redes consideradas. La gráfica muestra una tendencia al alza a medida que el número de clusters aumenta, teniendo su máximo para una red con 17 neuronas, con un tiempo total de 162.74 minutos. La razón por la cual el tiempo disminuye para redes con 18, 19 y 20 neuronas es la limitación que se impuso con respecto al número mínimo de elementos por cluster a medida que el número de neuronas aumenta, tal y como se expresó anteriormente en el numeral 3.2.2.



Gráfica 26. Tiempo empleado en hacer 10 – Fold Cross Validation en función del número de neuronas de la capa oculta.

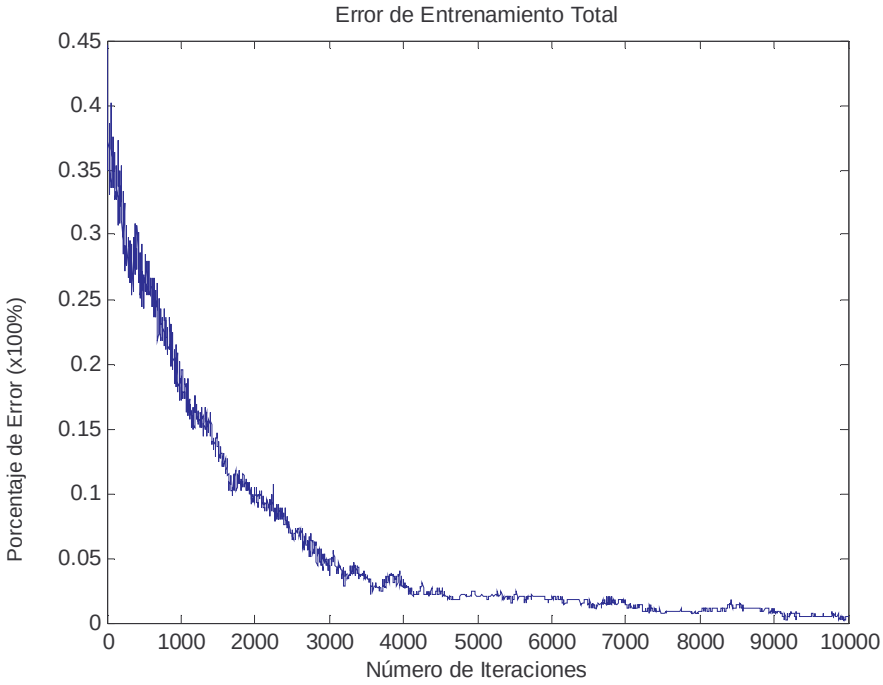
Basándose en los parámetros mencionados en el numeral 4.1. se escoge una red de 16 neuronas. En la *gráfica 25* se muestran los porcentajes de validación para cada uno de los subconjuntos y se aprecia que la mejor red tiene un porcentaje de validación del 80 %, la cual se escoge.

Haciendo uso del conjunto de evaluación es posible determinar el desempeño de la red. El porcentaje de acierto de la red escogida es del 80 % tomando el fragmento de la canción en 1 minuto.

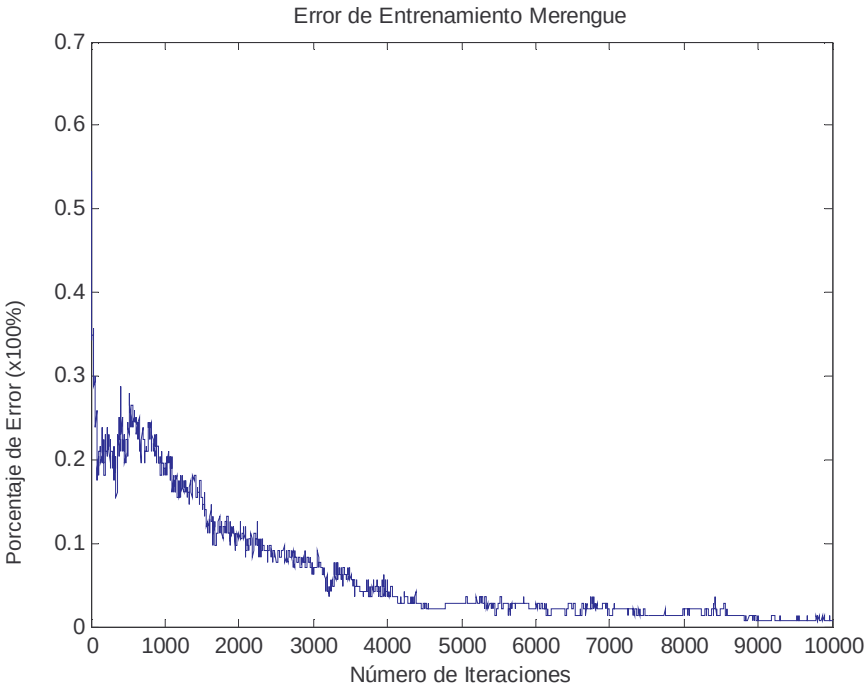
Tal y como se había mencionado anteriormente es posible mejorar el desempeño de la red empleando un sistema de votación tomando 3 muestras para diferentes tiempos. Al realizar esto para fragmentos ubicados en 0.75, 1 y 1.25 minutos se obtiene un porcentaje de acierto de la red del 84% que es superior al obtenido previamente.

4.3. CLASIFICADOR ADABOOST

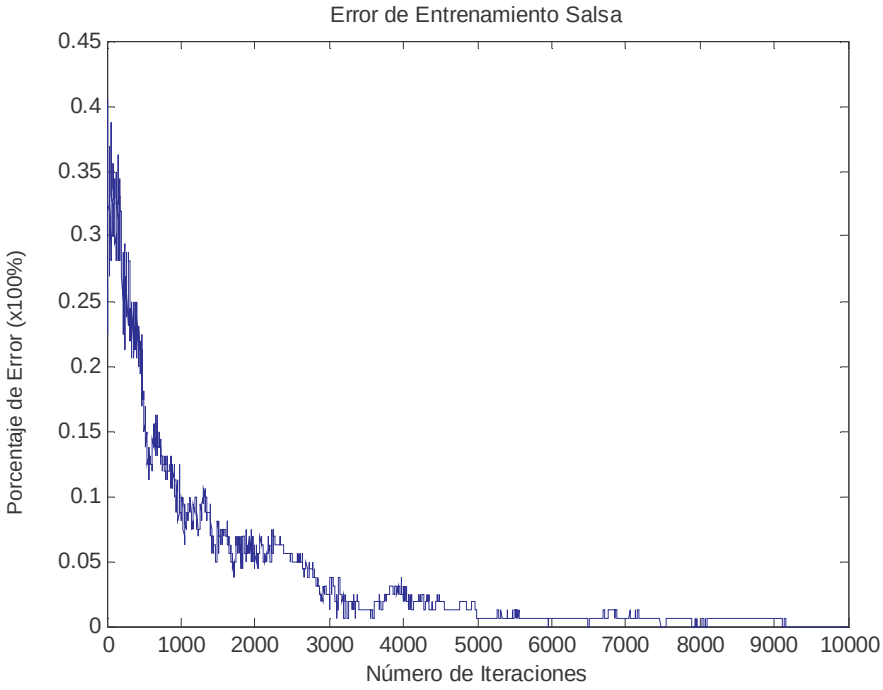
Se muestra a continuación el error de entrenamiento total en la *gráfica 27* y los errores de entrenamiento parcial para cada género en las *gráficas 28 ,29* y *30*.



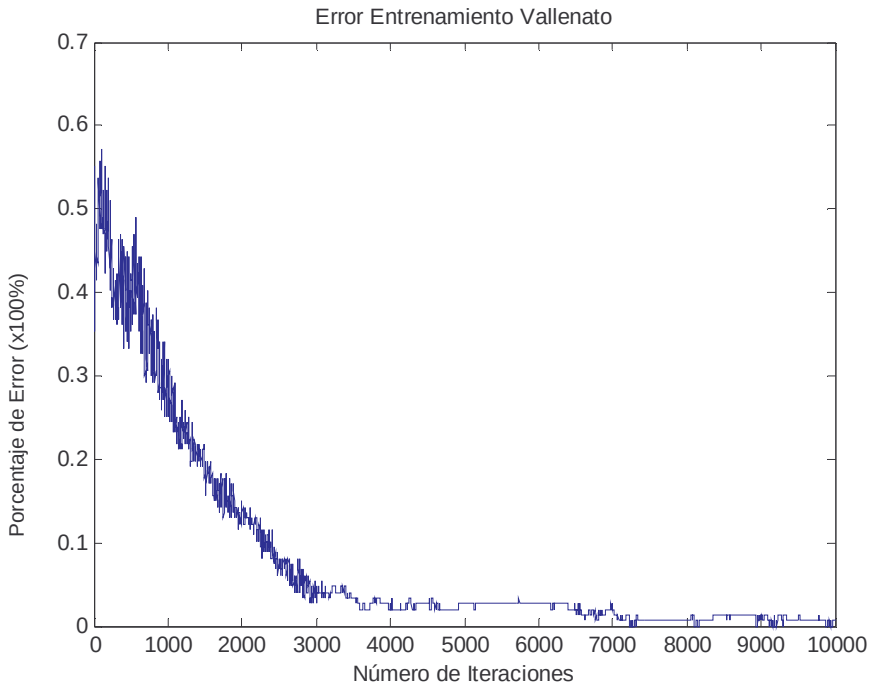
Gráfica 27. Error de entrenamiento total en función de las iteraciones.



Gráfica 28. Error de entrenamiento para merengue en función de las iteraciones.

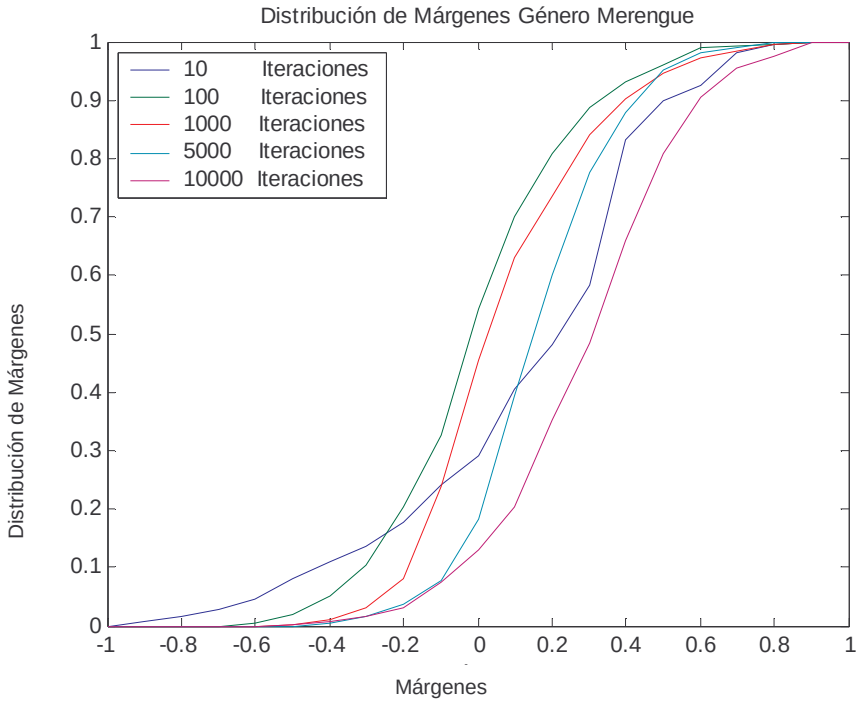


Gráfica 29. Error de entrenamiento para salsa en función de las iteraciones.

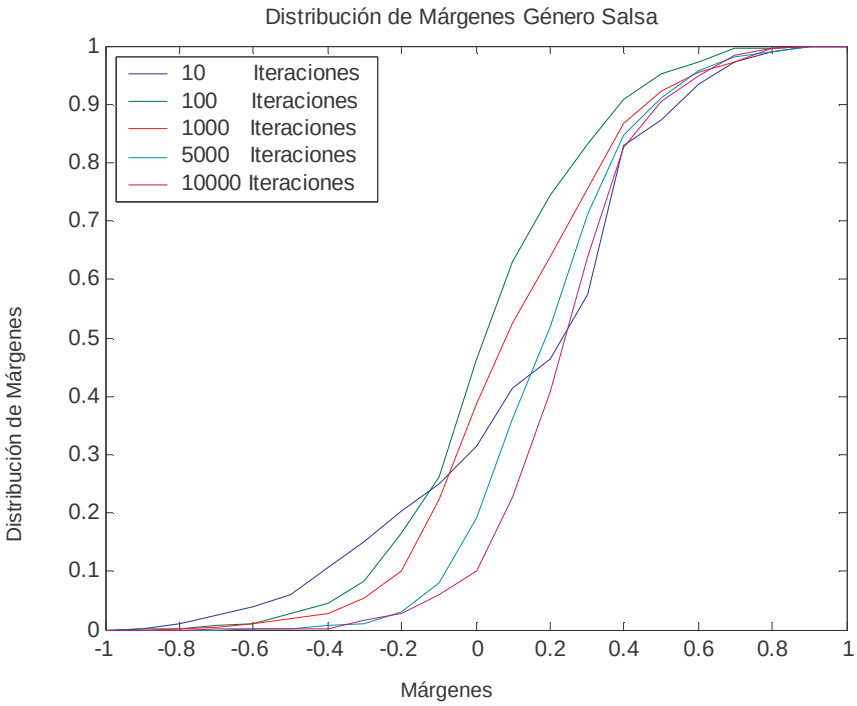


Gráfica 30. Error de entrenamiento para vallenato en función de las iteraciones.

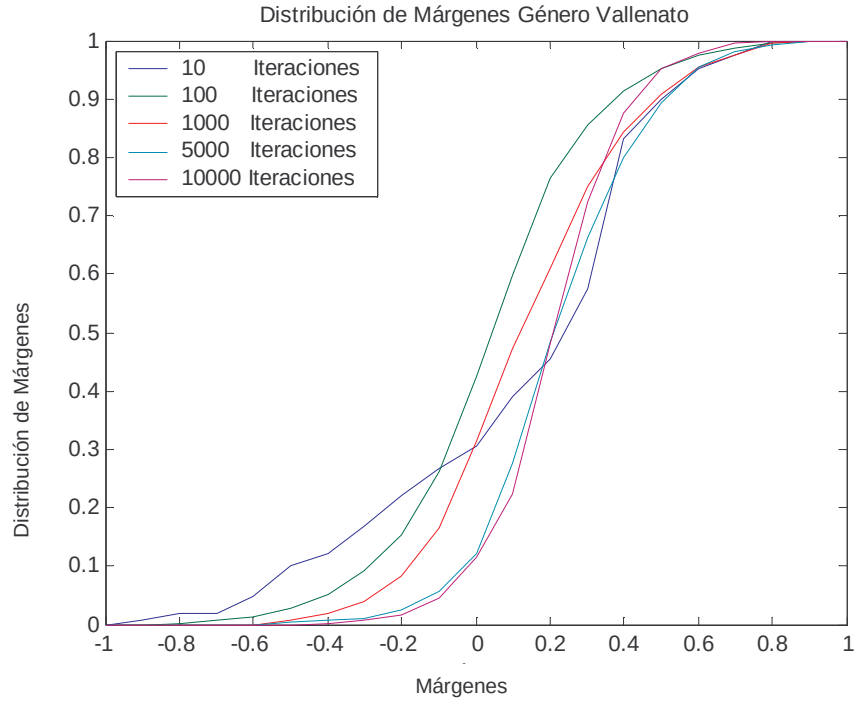
Adicionalmente para comprobar que se están aumentando los márgenes de confidencia en las *gráficas 31, 32* y *33* se muestran las respectivas distribuciones de márgenes para cada género.



Gráfica 31. Distribución de márgenes de entrenamiento para merengue.



Gráfica 32. Distribución de márgenes de entrenamiento para salsa.



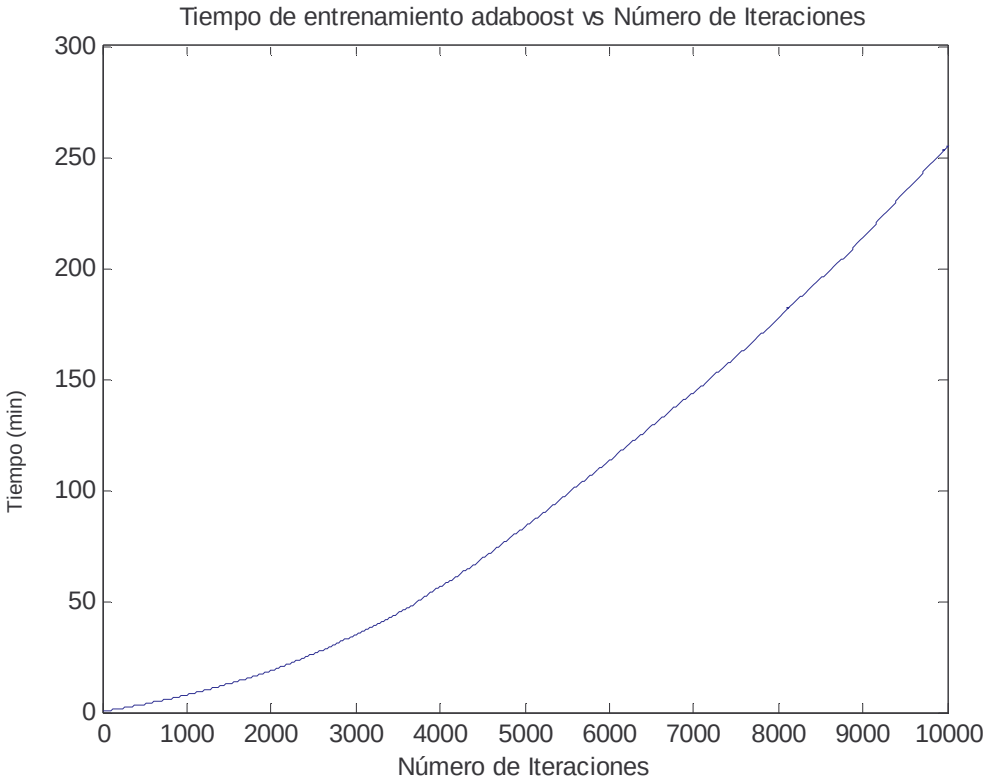
Gráfica 33. Distribución de márgenes de entrenamiento para vallenato.

De las *gráficas 27* a *30* se aprecia como el error de entrenamiento baja considerablemente hasta la iteración 3000 y continúa bajando lentamente hasta la iteración 10000 a un punto cercano a 0. El error total de entrenamiento alcanzó un valor mínimo de 0.0022, que equivale a una canción del conjunto de entrenamiento mal clasificada.

Se destaca que el error del merengue es el que decrece con mayor rapidez, pero presenta un pequeño incremento en la iteración 1000.

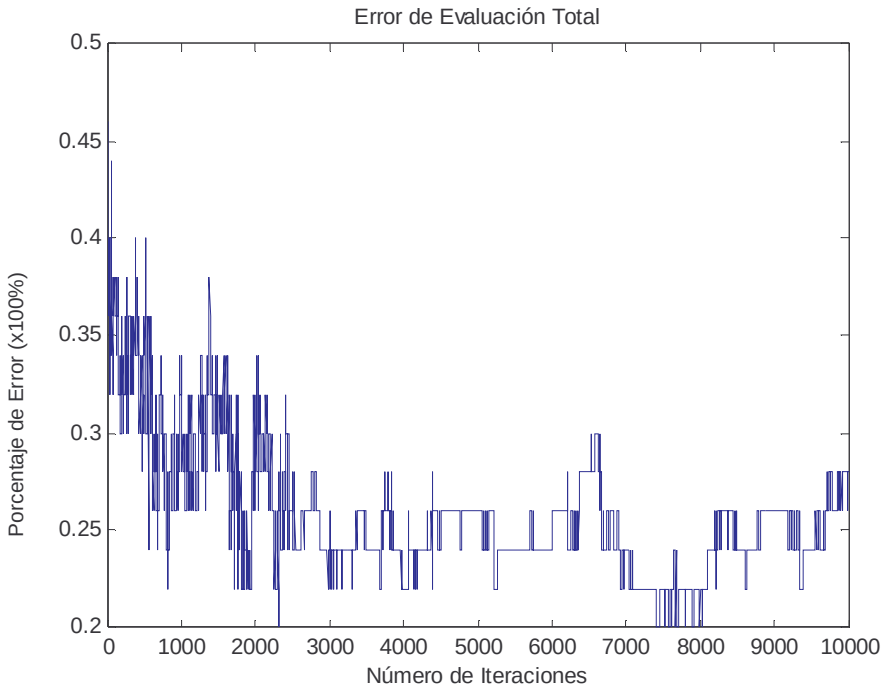
Por otra parte de las *gráficas 31* a *33* se confirma que el margen de confidencia que se incrementa más es el correspondiente al género merengue mientras que para el género vallenato este incremento es menor.

Se iteró hasta 10000 puesto que como se aprecia en la *gráfica 34* a medida que aumenta el número de iteraciones, el tiempo presenta un incremento exponencial. Adicionalmente se suspende el entrenamiento ya que dicho error ha alcanzado un valor cercano a cero a partir de la iteración 7000.

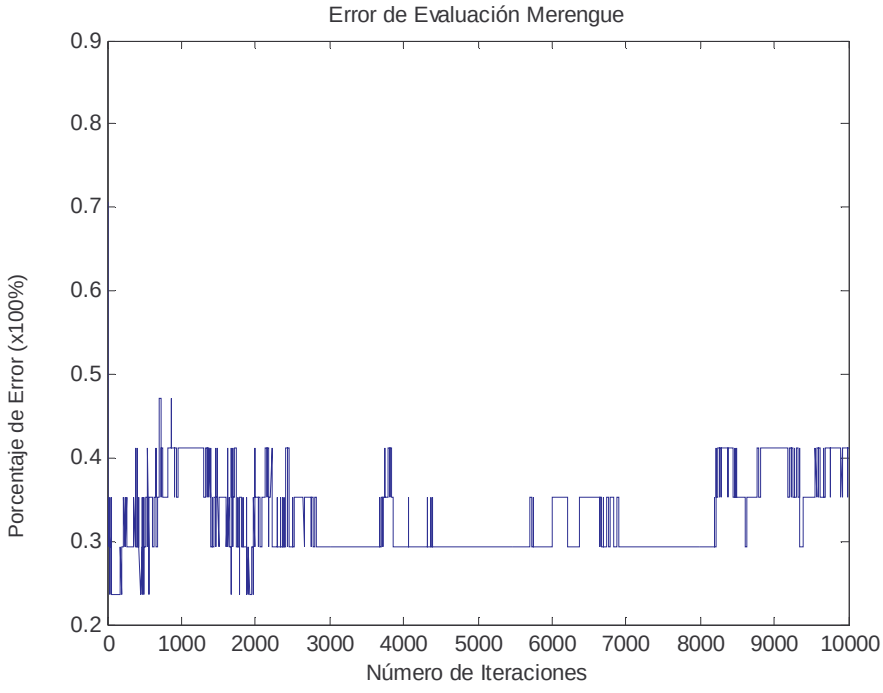


Gráfica 34. Tiempo de entrenamiento en función del número de iteraciones.

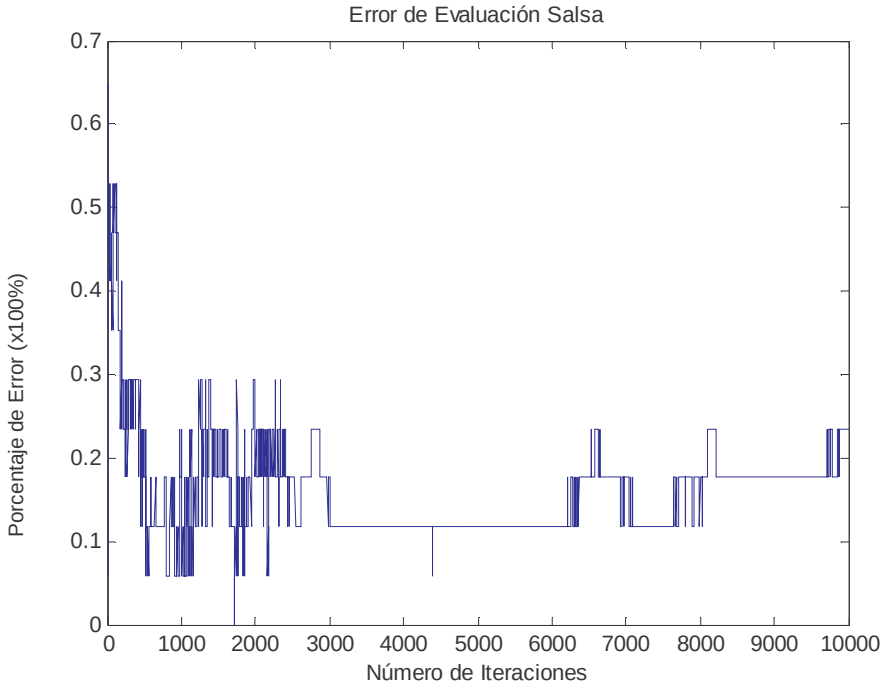
A continuación se presenta el error de evaluación total y para cada género en las *gráficas 35* a *38* para la red entrenada anteriormente. Así mismo se presentan las respectivas distribuciones de márgenes en las *gráficas 39* a *41*.



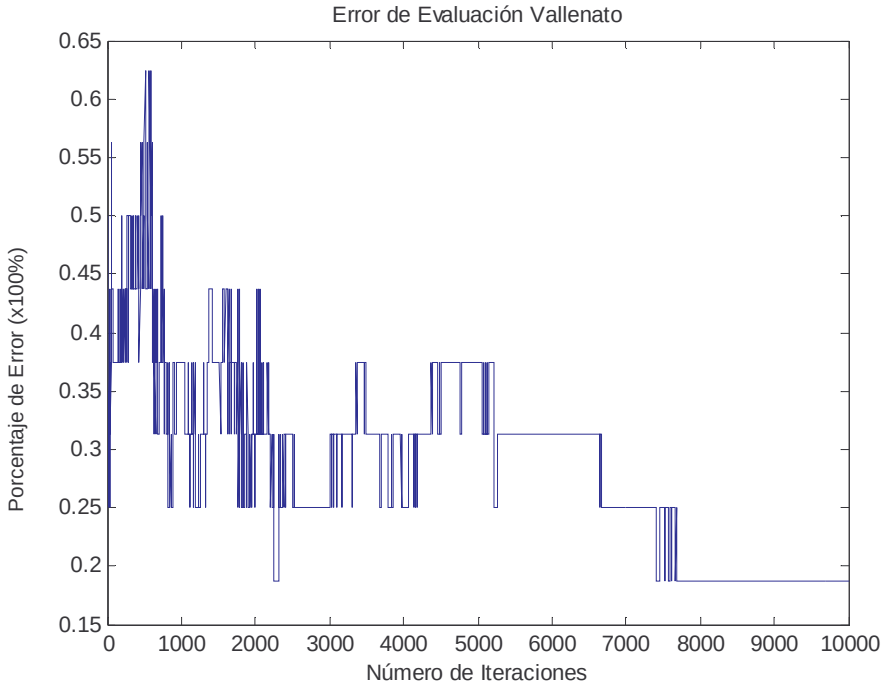
Gráfica 35. Error de Evaluación Total en función del número de iteraciones.



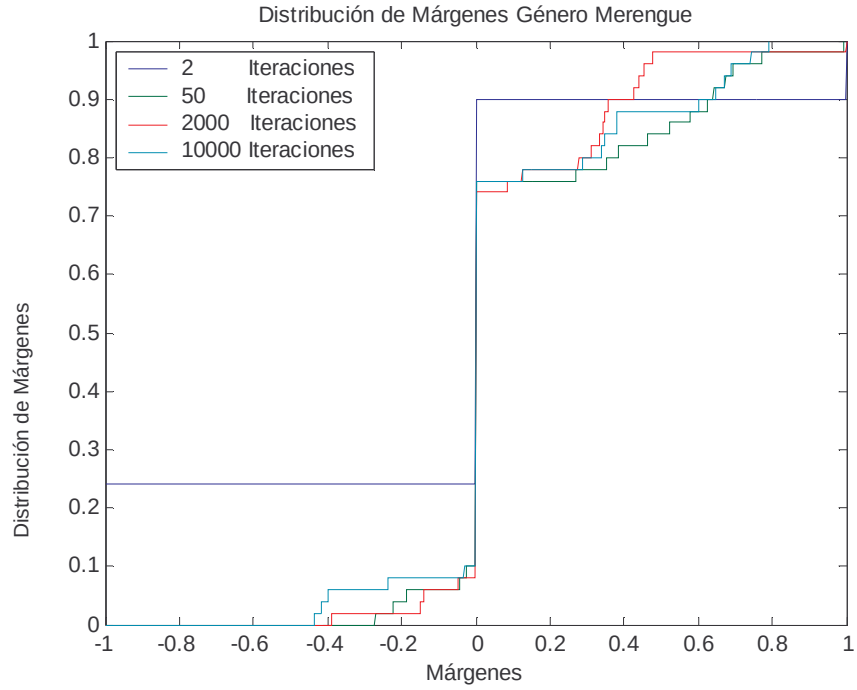
Gráfica 36. Error de Evaluación para merengue en función del número de iteraciones.



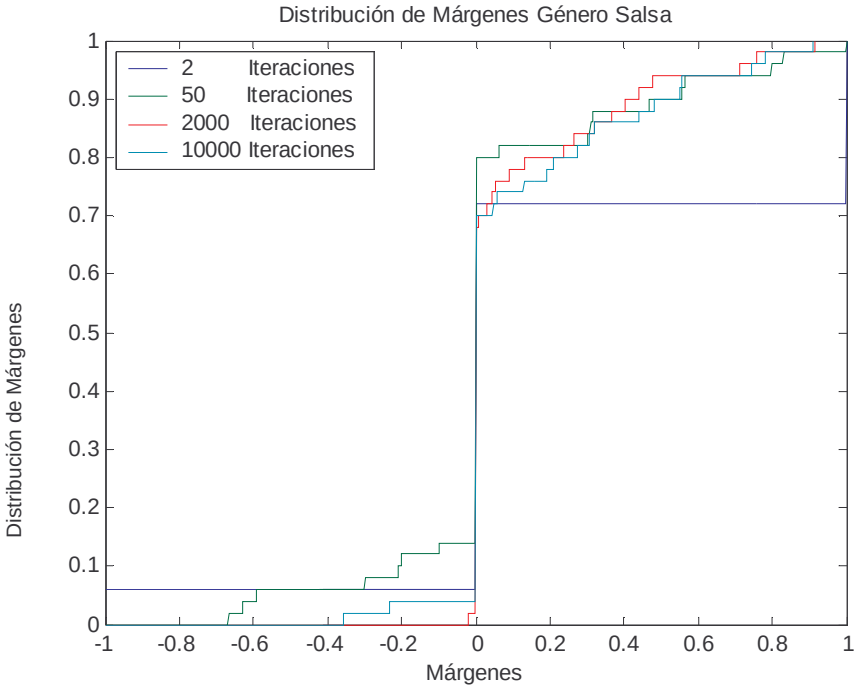
Gráfica 37. Error de Evaluación para salsa en función del número de iteraciones.



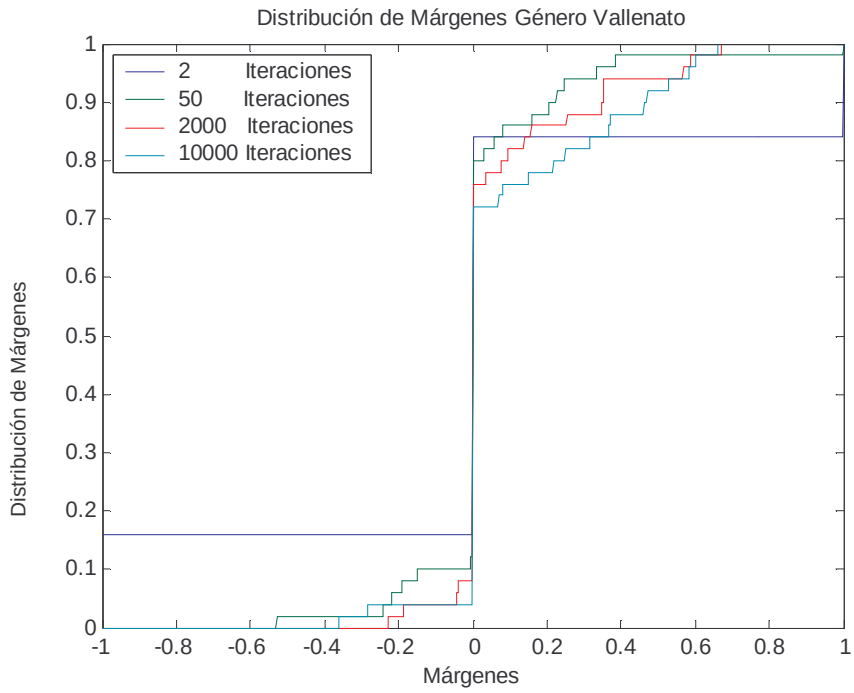
Gráfica 38. Error de Evaluación para vallenato en función del número de iteraciones.



Gráfica 39. Distribución de márgenes de evaluación para merengue.



Gráfica 40. Distribución de márgenes de evaluación para salsa.



Gráfica 41. Distribución de márgenes de evaluación para vallenato.

En el error de evaluación total mostrado de la *gráfica 35* se aprecia como éste también disminuye conforme aumenta el número de iteraciones. El tiempo empleado en realizar dicha evaluación fue de 11.6375 min.

Analizando el error de cada género por separado de acuerdo a las *gráficas 36* a *38* es importante notar, cómo para el merengue éste decrece rápidamente y se estabiliza, ocurriendo algo similar aunque con decrecimiento más lento para la salsa; mientras para el vallenato la disminución del error sólo se presenta alrededor de las 5000 iteraciones, razón por la cual, los márgenes de confidencia del vallenato se incrementan en las iteraciones finales mientras para el merengue y la salsa lo hacen a partir de las iteraciones iniciales.

El error mínimo total alcanzado para el conjunto de evaluación es de 20%

En la *tabla 2* se resumen los porcentajes de clasificación correcta para el clasificador Adaboost implementado.

	Total	**Merengue**	**Salsa**	**Vallenato**
Porcentaje de clasificación correcto (%)	80	70.59	88.24	81.25

Tabla 2. Porcentajes de clasificación correcto utilizando Adaboost.

4.4. COMPARACIÓN ENTRE LOS DIFERENTES CLASIFICADORES IMPLEMENTADOS

La Tabla 3 presenta un resumen detallado de todas las características obtenidas para las redes de Propagación Inversa y de Funciones de Base Radial.

	RED DE PROPAGACIÓN INVERSA	RED DE FUNCIONES DE BASE RADIAL
Tiempo total empleado en la búsqueda de la Red Óptima.	101.8357 horas	20.84 horas
Número de Neuronas Óptimo para la Capa Oculta.	6	16
Tiempo total empleado en el entrenamiento de la Red Óptima	173.8836 minutos	215.56 minutos
Número de Iteraciones realizadas en la Validación Cruzada.	200	2500
Porcentaje Promedio de Clasificación en la Validación Cruzada	70.67	76.22
Máximo Porcentaje de Clasificación en la Validación Cruzada	75.56	80
Porcentaje Correcto de Evaluación	76	80
Tiempo escogido para realizar la Evaluación	60 segundos	60 segundos
Porcentaje Correcto de Evaluación empleando Votación 2 de 3	78	84
Tiempos escogidos para realizar la evaluación con Votación	60, 75 y 90 segundos	45, 60 y 75 segundos

Tabla 3. Comparación de los resultados obtenidos para las topologías Red de Propagación Inversa y Red de Funciones de Base Radial.

La razón por la cual se utilizó como clasificador débil una red de Funciones de Base Radial es claramente visible si se observa el campo relacionado al tiempo. Allí es notoria la diferencia existente entre el tiempo necesario para el entrenamiento de una red de propagación inversa y una red de funciones de base radial. Específicamente hablando, este tiempo es casi 5 veces inferior.

Para la implementación de Adaboost, se utilizó una red de funciones de base radial con 11 neuronas en la capa escondida debido a que el tiempo requerido para su entrenamiento es muy bajo comparado con los demás tamaños de red .

Fue posible cerciorarse de la forma en la cual se mejora el desempeño de un clasificador débil mediante la observación del porcentaje obtenido al realizar las 10000 iteraciones. Adicionalmente al hecho que los errores de entrenamiento y evaluación disminuyen conforme aumenta el número de iteraciones, se observa que el porcentaje de acierto en la evaluación obtenido utilizando Adaboost es superior a éste mismo porcentaje para la mejor red que se obtiene usando Validación Cruzada, como se muestra en la *tabla 4*.

	Red de Funciones de Base Radial de 11 neuronas en la Capa Escondida	Clasificador obtenido utilizando Adaboost para la red anterior con 10000 iteraciones
Porcentaje de acierto en la Evaluación	70	80

Tabla 4. Resultado de la aplicación de Adaboost a un clasificador débil.

5. CONCLUSIONES

El estudio inicial realizado acerca de la obtención y extracción de parámetros relevantes a partir de un archivo de música en formato digital muestra que las características utilizadas son apropiadas y permiten diferenciar entre los tres tipos de géneros que forman parte de este estudio. Los resultados arrojados por los clasificadores implementados corroboran la afirmación anterior, ya que de no ser así, la categorización de canciones no hubiese mostrado los resultados satisfactorios que fueron obtenidos.

Los parámetros extraídos a partir de la Transformada Wavelet tienen una estrecha relación con la forma en la cual el ser humano realiza el proceso de distinción de música por género. Esto es, se centran en la identificación de un patrón rítmico básico presente en la música

Es de destacar que los parámetros que son extraídos buscando un patrón rítmico son producto de la experimentación y la realización de numerosas pruebas en las cuales se buscó representar esta característica . Esto constituye un aporte importante al campo de la Clasificación de Señales de Audio.

El desarrollo del proyecto obtuvo como resultado dos tipos básicos de clasificadores que fueron producto de sucesivos estudios en los cuales se optimizaron las respectivas arquitecturas de red, buscando aquellas para las cuales se presentaba el mejor desempeño

medido en términos de la clasificación correcta de canciones que no fueron usadas durante la fase de entrenamiento.

Las mejoras introducidas durante la fase de desarrollo fueron las siguientes:

- Extracción de Características Relevantes a partir de la transformación del algoritmo de Karhunen – Loéve.
- 10 – Fold Cross Validation con el fin de determinar el orden apropiado del modelo.
- Incorporación de Conocimiento previo para inferir el intervalo de tiempo adecuado para tomar la muestra de la canción a clasificar.
- Votación 2 de 3 con el fin de disminuir la probabilidad de error debida a la escogencia de un intervalo de tiempo no representativo de la estructura musical.

La última y más relevante mejora implementada fue el uso de Adaboost aplicado al problema específico propio de este trabajo. Se demostró que su utilización eleva el porcentaje de acierto en la clasificación para un clasificador débil con 3 salidas.

Los resultados producidos en este proyecto son bastante buenos en comparación con los que han sido obtenidos por otros investigadores que han trabajado en el mismo campo. Por ejemplo Seth Golub **[2]** obtuvo un porcentaje de clasificación correcto del 77% utilizando redes de propagación inversa para dos géneros muy similares, mientras que Paul Scott **[3]** logró un porcentaje de clasificación de 94.8% utilizando como entradas, géneros muy diferentes como Rock, Clásica, Soul y Country. Otros trabajos como el de Soltau **[5]** obtuvieron 81.9% para Rock, Pop, Tecno y Clásica, mientras Tzanetakis **[25]** utilizó seis

géneros y obtuvo un porcentaje de clasificación de 60%. Estos porcentajes son inferiores para aquellos estudios que utilizaron ritmos similares y comparables con los estudios que usaron ritmos diferentes. A pesar de la similitud existente en los géneros utilizados en este proyecto, se realizó un proceso de clasificación óptima que contribuye al desarrollo de la redes neuronales artificiales aplicadas no solo a la clasificación de música sino a la clasificación de señales de audio.

Futuros trabajos que tomen como base el presente proyecto, y cuyo tema sea la clasificación de música por género pueden orientarse hacia los siguientes aspectos:

- Implementar sistemas clasificadores de música que no utilicen Redes Neuronales Artificiales con el fin de comparar los resultados con los obtenidos en este estudio.
- Anexar nuevas topologías de Redes Neuronales con el fin de comparar su desempeño general en cuanto a tiempo de entrenamiento, porcentaje de clasificación correcta y tamaño de la red.
- Realizar el mismo proceso de éste estudio con un número superior de géneros, para determinar si los parámetros que aquí fueron extraídos siguen siendo suficientes para realizar la clasificación.
- Efectuar la extracción de características a partir de un archivo en formato MP3, sin tener la necesidad de convertirlo previamente a formato WAV.

BIBLIOGRAFÍA

[1] Gerhard, David. Ph.D. Depth Paper: Audio Signal Classification. School of Computing Science. Simon Fraser University. 2000.

[2] Golub, Seth. Classifying Recorded Music. MSc in Artificial Intelligence. Division of Informatics. University of Edinburgh. 2000.

[3] Scott, Paul. Music Classification using Neural Networks. EE373B Proyect. Stanford University. 2001.

[4] Foote, J.. A similarity measure for automatic audio classification. Technical report, Institute of Systems Science, National University of Singapure. 1997

[5] Soltau, H., Schultz, T., Westphal, M., y Waibel, A.. Recognition of music types. Interactive System Laboratory. 1998.

[6] Bishop, Christopher. Neural Networks for Pattern Recognition. New York. Oxford University Press Inc. 1995.

[7] Desain, P. Autocorrelation and the study of musical expression. Centre of knowledge technology , Utrecht School of arts. 1990

[8] Proakis, John y Manolakis,Dimitris. Tratamiento digital de señales. Principios, Algoritmos y Aplicaciones. 3ra Edición Prentice Hall. 1998

[9] Oppenheim, Alan y Willsky, Alan. Signals and Systems. Prentice Hall Signal Processing Series. 1983.

[10] Matlab Help. The MathWorks, Inc. Versión 6.1.0.450. Release 12.1.

[11] Burrus, Gopinath. Introduction to Wavelets and Wavelet Transforms A Primer. Prentice Hall. 1998

[12] Fukunaga, K. Introdutcion to Statistical Pattern Recognition. 2da Edición. SanDiego Academic Press. 1990

[13] Paplinski, A. Neural Networks introduction. School of Computer Science and Software Engineering.

[14] Rumelhart, D.E., G.E. Hinton, y R.J. Williams. Learning Internal represantations by error propagation. Parallel Distributed Processing: Exploration in the Microstructure of Cognition. Cambridge, MIT Press. Reimpreso en 1988.

[15] D.S. Bromead and D. Lowe. Multivariate Functional Interpolation and Adaptive Networks. Complex Systems, 2:321-355, 1988.

[16] Orr, Mark. Introduction to radial basis function networks. Centre of Cognitive Science University of Edinburgh. 1996

[17] Moody, J y Darken, C. Fast learning networks of locally tuned processing units. Neural computation 1, 281-294. 1989

[18] Stone. M. Cross – validatory choice an assessment of statistical predictions. Journal of the Royal Statistical Society. 111-147. 1974

[19] R.E. Schapire. Theoretical views of boosting. In Computational Learning Theory: Fourth European Conference, EuroCOLT'99, 1999.

[20] R, Meir y G, Rätsch. An introduction to boosting and leveraging. In S. Mendelson and A. Smola, editors, páginas 119-184. Springer, 2003

[21] Auterlitz, Paul. Local and International Trends in Dominican merengue.

[22] Manuel, Peter. Caribbean Currents: Caribbean Music from Rumba to Reggae. Philadelphia: Temple UP, 1995.

[23] Samper, Daniel. 100 años de Música Vallenata. Antología. 1999

[24] www.lame.org Grupo de investigación dedicado a desarrollar algoritmos para descomprimir MP3.

[25] Tzanetakis, George; Essl, Georg; Cook, Perry. Automatic Musical Genre Classification of Audio Signals. Computer Science Department. Princeton University. 2001

[26] Wold, E., Blum, T., Keislar, D., and Wheaton, J. (1996). Content-based classification, search, and retrieval of audio. *IEEE Multimedia*, 3(3):27–36.

[27] Fonseca, Oscar y Fonseca Otto. Categorización automática de texto. Trabajo de Grado. Facultad de Ingeniería Carrera de Ingeniería Electrónica. Pontificia Universidad Javeriana. 2002.

ANEXO A - INTERFAZ GRÁFICA CON EL USUARIO

Este programa permite realizar clasificación de archivos de audio digitales para los géneros merengue, salsa y vallenato. El software emplea internamente las diversas topologías de redes neuronales artificiales implementadas en este estudio, cinco en total. La pantalla inicial del programa de clasificación de música se presenta a continuación.

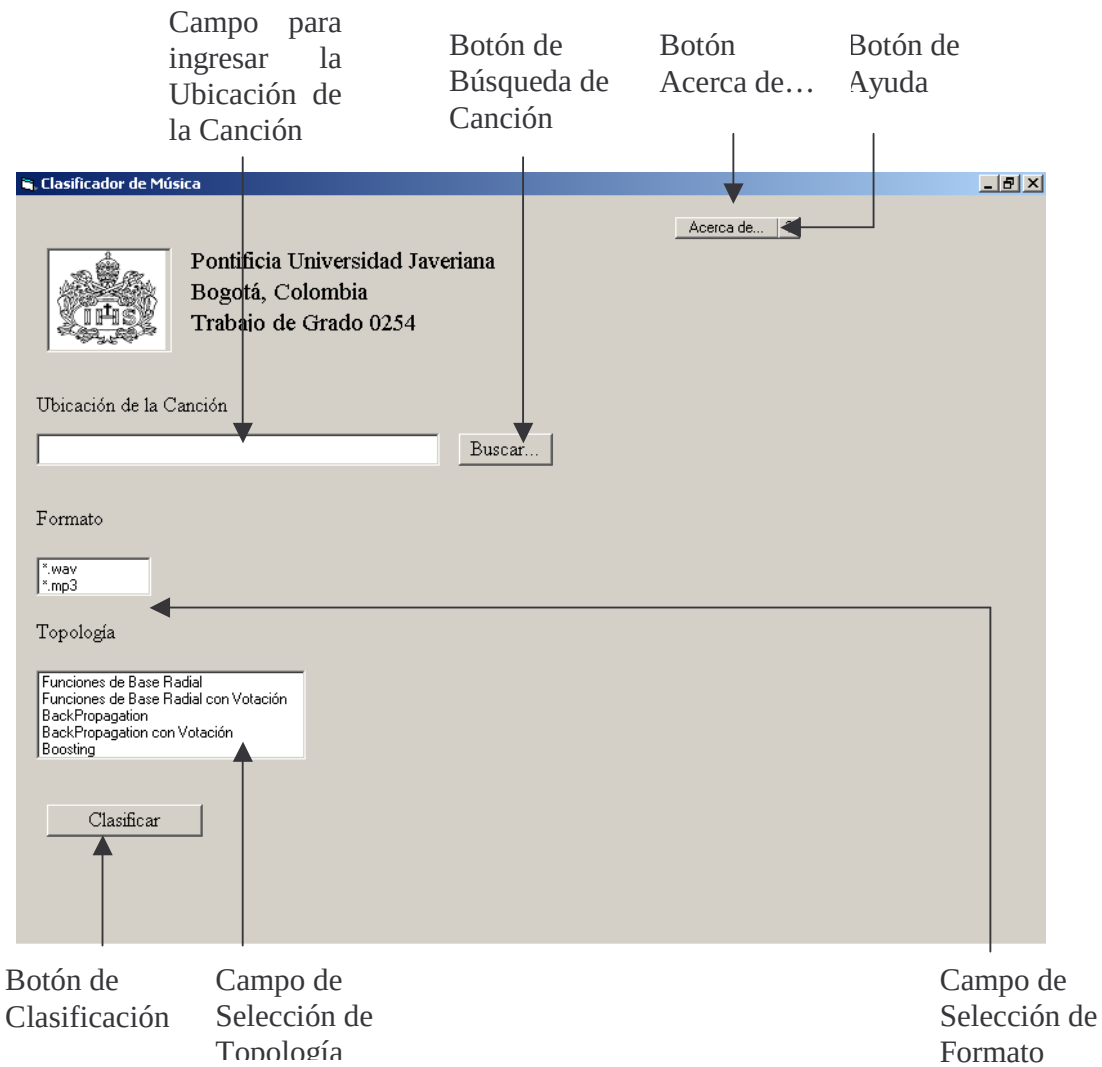


Figura 12. Pantalla inicial Programa Clasificador De Música.

Si conoce la ubicación del archivo de música, se escribe en el campo destinado para tal fin. De lo contrario, se oprime el botón “Buscar” y se selecciona la ruta. Al utilizar esta opción se observa la siguiente pantalla.

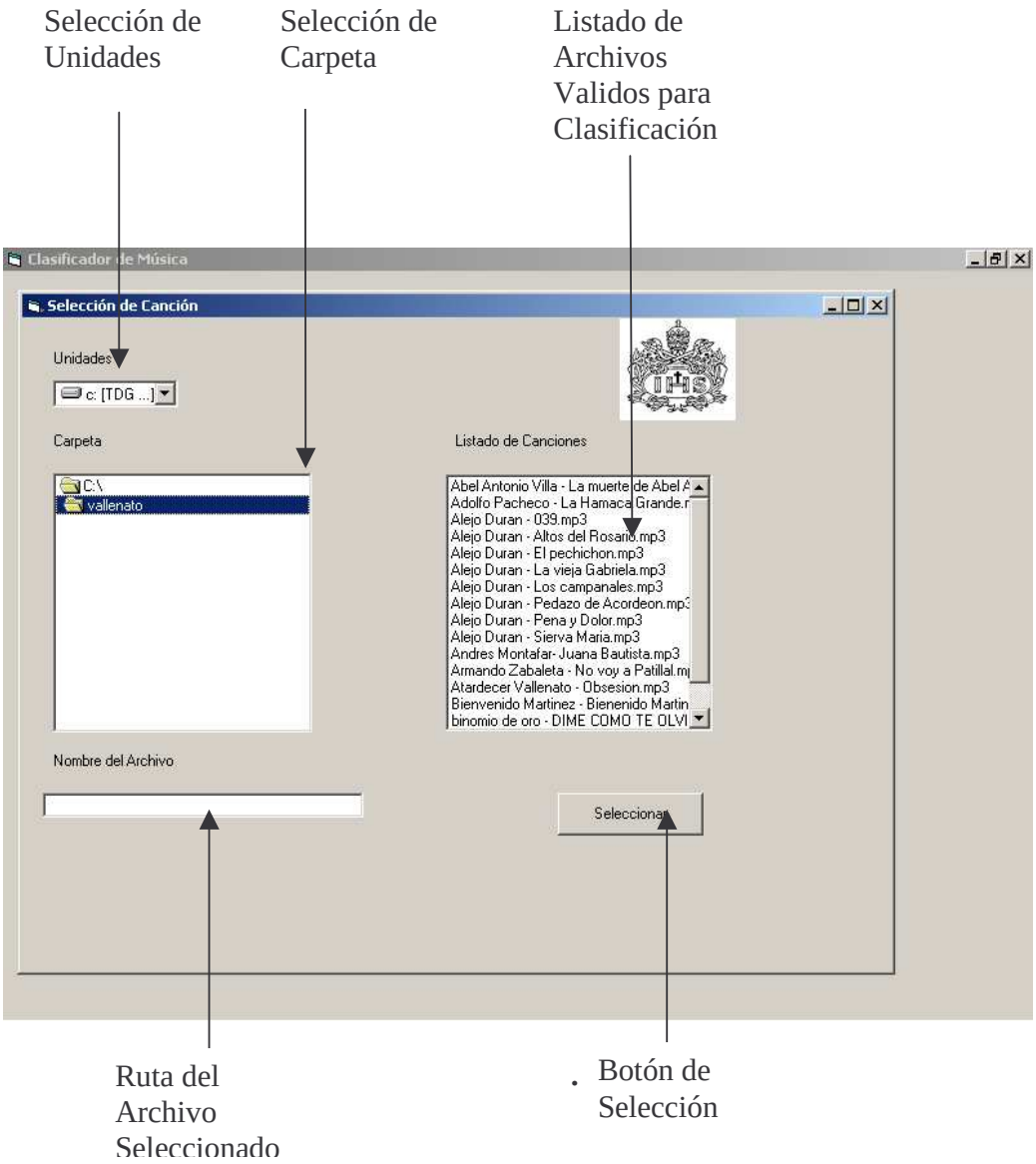


Figura 13. Pantalla de Búsqueda de Canción.

Es importante recordar que el listado de canciones que se puede seleccionar depende del formato se que haya escogido, es decir, si selecciona formato .WAV solo verá el listado de canciones en dicho formato, lo mismo ocurre para el formato .MP3.

Una vez se haya escogido la canción que desea clasificar se debe escoger la Topología de la Red Neuronal Artificial con la cual desea realizar la clasificación. Las opciones incorporadas son Backpropagation, Redes de Funciones de Base Radial, Backpropagation con Votación, Redes de Funciones de Base Radial con Votación y AdaBoost. Por último se oprime el botón "Clasificar" y se espera mientras el programa procesa la canción requerida y determina el género al cual pertenece. Cuando se finaliza tal proceso, aparece un anuncio con dicho género como se ve a continuación.

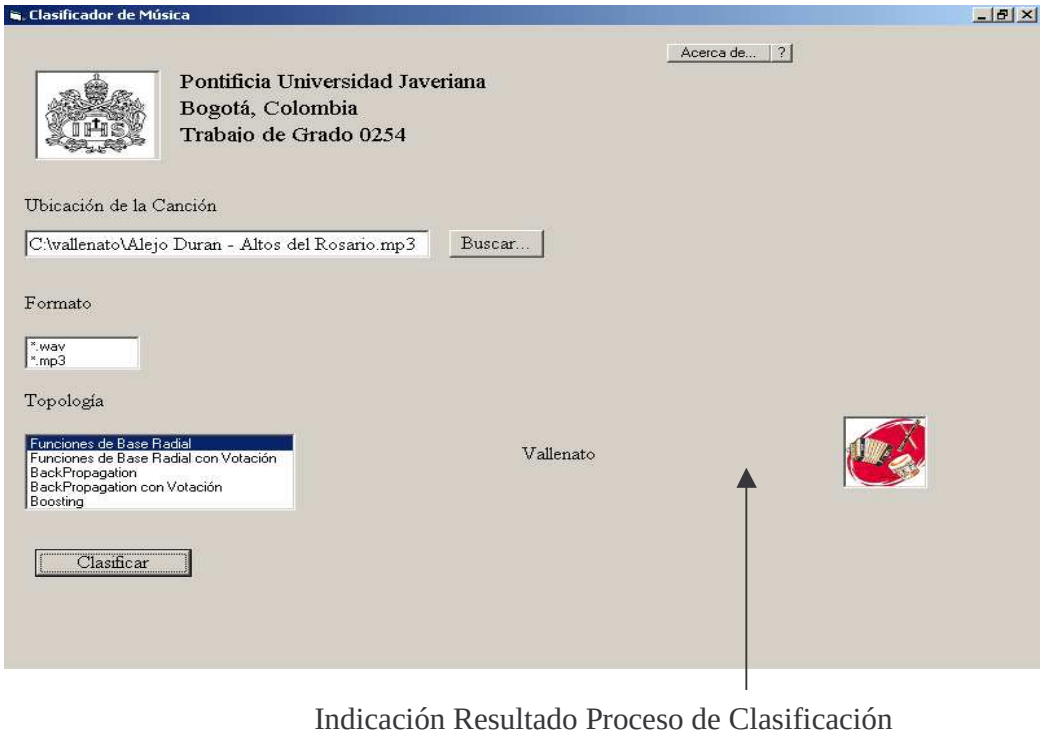


Figura 14. Pantalla de Resultado del Proceso de Clasificación.

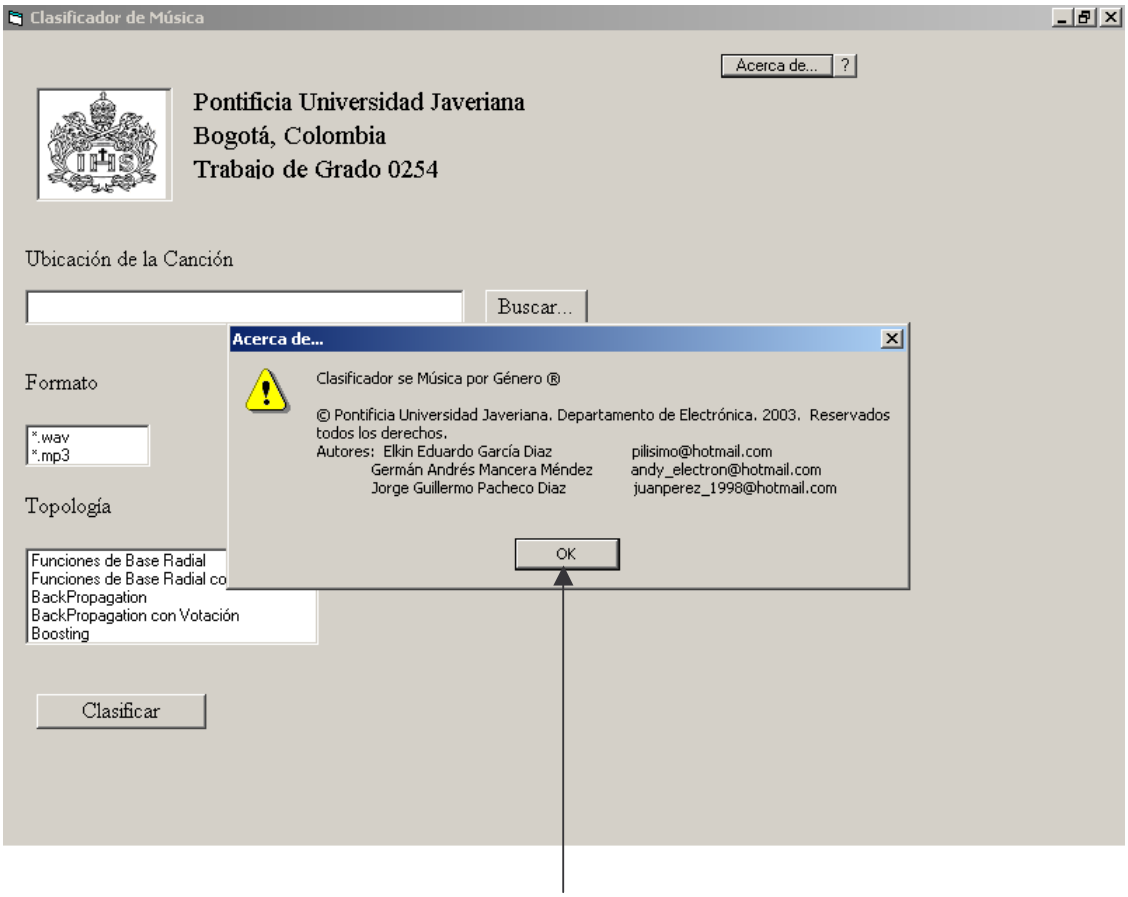


Figura 15. Pantalla Acerca de…

Además se presenta una pantalla con la información de los autores del programa al hacer clic en el botón "Acerca de…" con el fin de contactarlos si existe cualquier tipo de duda o de comentario sobre la utilización del programa. Por último se presenta el esquema típico para realizar la clasificación de una canción al hacer clic sobre el botón de "Ayuda", como se ve en la *figura 16.*

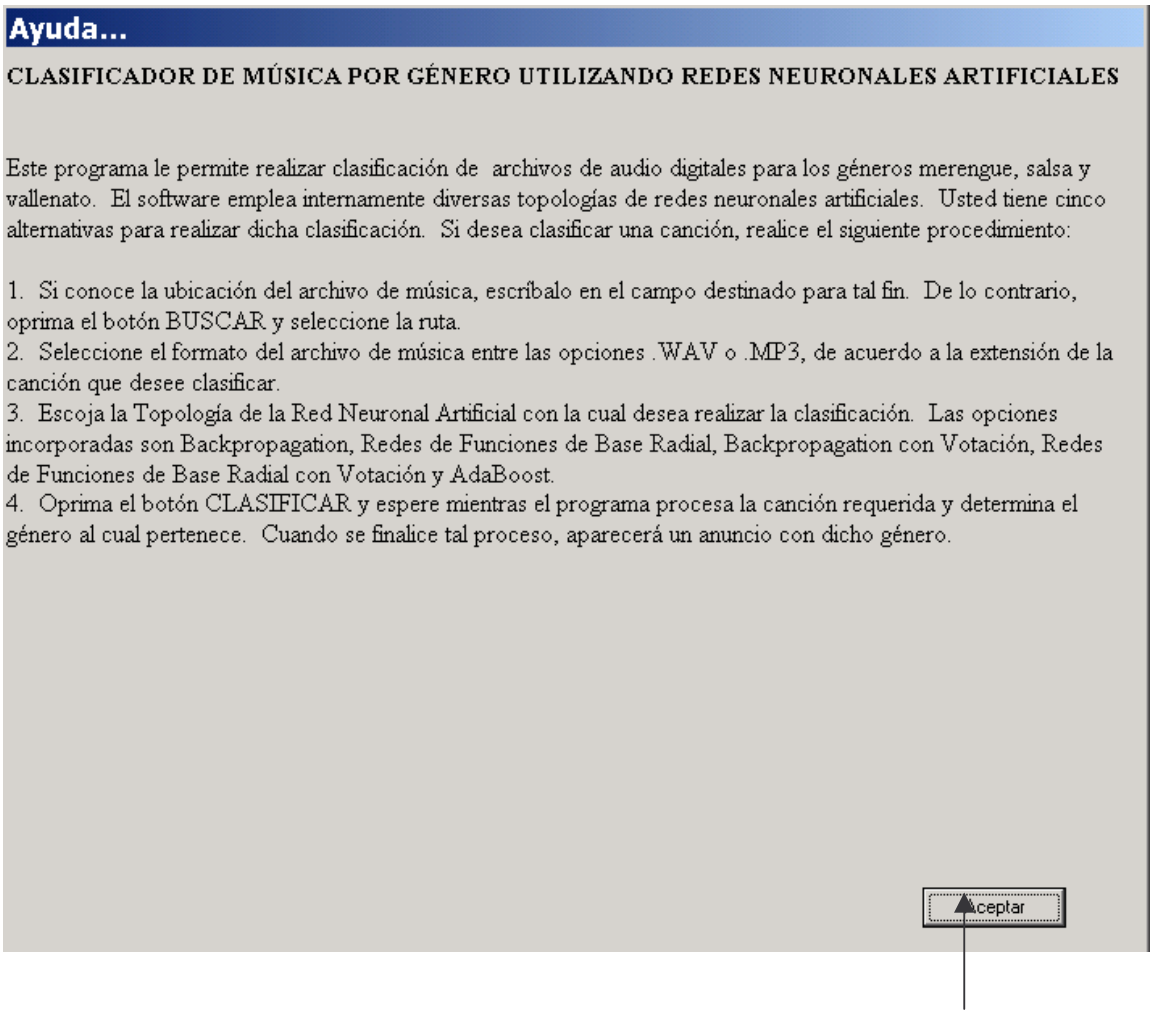


Figura 16. Pantalla de Búsqueda de Canción.