**UNIVERSIDAD AUTÓNOMA METROPOLITANA – IZTAPALAPA**
**POSGRADO EN CIENCIAS Y TECNOLOGÍAS DE LA INFORMACIÓN**

# Identificación de Versiones Musicales (Covers) utilizando Aprendizaje Maquinal

Tesis que presenta

**Andree Silva Reyes**

Para obtener el grado de

**Maestro en Ciencias y Tecnologías de la Información**

Asesora: M.I. Fabiola Margarita Martínez Licona

Jurado calificador:

Presidente: Dr. José Francisco Martínez Trinidad

Secretario: Dr. John Goddard Close

Vocal: M.I. Fabiola Margarita Martínez Licona

División de Ciencias Básicas e Ingeniería

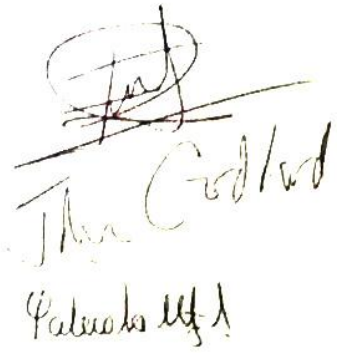
México, D.F. 2016

**UNIVERSIDAD AUTÓNOMA METROPOLITANA – IZTAPALAPA**
**POSGRADO EN CIENCIAS Y TECNOLOGÍAS DE LA INFORMACIÓN**

# Identificación de Versiones Musicales (Covers) utilizando Aprendizaje Maquinal

Tesis que presenta
**Andree Silva Reyes**
Para obtener el grado de
**Maestro en Ciencias y Tecnologías de la Información**

| | |
|---|---|
| Asesora | M.I. Fabiola Margarita Martínez Licona |
| Jurado calificador | |
| Presidente | Dr. José Francisco Martínez Trinidad |
| Secretario | Dr. John Goddard Close |
| Vocal. | M.I. Fabiola Margarita Martínez Licona |



División de Ciencias Básicas e Ingeniería
México, D.F. 2016

# Resumen

La tarea de reconocer cuándo una canción es una versión o cover de otra es una tarea relativamente fácil para el ser humano cuando se conoce la canción. Sin embargo, hacer que una máquina realice este trabajo resulta complejo debido al número de variables que se involucran en la elaboración del cover, mismas que incluyen variaciones en el ritmo, tempo, instrumentación, género y duración con respecto a la versión original. En este proyecto se desarrolló una metodología para identificar covers a partir de la aplicación y análisis de técnicas de aprendizaje maquinal, procesamiento de señales y estadística de segundo orden con la finalidad de obtener aquella configuración que diera los mejores resultados. Para esto se trabajó con la base de datos Million Songs Dataset que nos otorgó los metadatos de las canciones, a partir de los cuales se utilizaron los datos pertenecientes a las características acústicas de la canción, tales como, pitches y timbres. A lo largo del proyecto se experimentó con diferentes técnicas de tratamiento de los metadatos que proporcionó la base de datos y se pudo apreciar su utilidad para la tarea a desarrollar. De acuerdo a los resultados obtenidos, se obtuvo un sistema que integra un procesamiento en frecuencia sobre los pitches alineados con el beat, la aplicación de una codificación rala y un sistema de agrupamiento de datos que arrojó un 63% de identificación correcta de covers. También se obtuvo información sobre el posible uso de técnicas combinadas de aprendizaje supervisado con diferentes tipos de métricas dando pie a futuras experimentaciones para mejorar los resultados.

**Palabras clave: identificación de covers, codificación rala, recuperación de información musical.**

# Abstract

The task of recognizing when a song is a version or cover of another is a relatively easy task to do for humans when the song is known. However, to cause that a machine perform this work is complex due to the number of variables involved in preparing the cover, including variations in rhythm, tempo, instrumentation, genre and duration compared to the original version. In this project a methodology to identify covers from the application and analysis of machine-learning techniques, statistical signal processing and second order statistics was developed, in order to get that configuration to give the best results. For this we worked with the database Dataset Million Songs that gave us the metadata of the songs, from which data belonging to the acoustic characteristics of the song, such as pitches and timbres were used. Throughout the project we experimented with different data treatment techniques applied to the metadata provided by the database and we could see its usefulness to the task at hand. According to the results, a system that integrates processing frequency on pitches aligned with the beat, the implementation of a sparse coding and a data clustering system that showed a 63% correct identification of covers was obtained. Information on the possible use of supervised learning techniques combined with different types of metrics giving rise to future experiments to improve the results was also obtained.

**Keywords: cover identification, sparse autoencoder, music information retrieval.**

# Agradecimientos

A mis padres Lucia Reyes y Felipe Silva por el apoyo incondicional que siempre me han otorgado, por la educación y consejos que me han dado, mismos que han servido para afrontar nuevos retos en la vida.

A mi asesora la maestra Fabiola Martínez por la confianza y tiempo que deposito en mi para la realización de este proyecto.

Al Consejo Nacional de Ciencia y Tecnología (CONACyT) por brindarme el financiamiento económico, que ha sido el sustento para desarrollar este grado académico.

A la Universidad Autónoma Metropolitana por permitirme hacer uso de sus instalaciones. A sus profesores por compartir sus conocimientos durante estos estudios.

# Contenido

Pág.

# Lista de figuras

**Pág.**

# Lista de tablas

**Pág.**

CAPÍTULO 1

# INTRODUCCIÓN

La intersección entre la música, el aprendizaje maquinal y el procesamiento de señales ha permitido abordar un amplio rango de tareas entre las que se pueden mencionar la identificación automática de melodías, acordes e instrumentos, la identificación y caracterización de tiempos y estructuras a largo plazo o el reconocimiento de géneros y versiones musicales [1]. Organizaciones como la ISMIR (Sociedad Internacional de Recuperación de Información Musical, por sus siglas en inglés)[1] y MIREX (Music Information Retrieval Evaluation eXchange)[2] la han aprovechado para promover el acceso, la organización y comprensión de la información musical, centrándose en la investigación y el desarrollo de sistemas computacionales que solventen esta serie de tareas.

En lo que respecta a la tarea de identificar si una canción es una nueva versión o un cover de una ya existente, ésta resulta ser bastante compleja dadas las características que un cover puede presentar. En este sentido, debemos tomar en cuenta que una versión musical o cover es una nueva interpretación, en directo o en estudio, de una canción grabada previamente por otro artista[3]. Lo anterior implica que un cover puede variar en ritmo, tempo, instrumentación, género o duración, con

[1] http://www.ismir.net/society.html
[2] http://www.music-ir.org/mirex/wiki/MIREX_HOME
[3] http://es.wikipedia.org/wiki/Versión_(música)

respecto a su versión original. Por ejemplo, la canción Summertime interpretada originalmente por Abbie Mitchell en 1935, cuenta con 1200 covers realizados hasta la fecha de acuerdo al proyecto Second Hand Songs (SHS)[4]; de estas versiones podemos encontrar algunas parecidas a la canción original mientras que otras suenan bastante diferente. En la Tabla 1.1 se muestran los géneros musicales en los que se han grabado los covers de Summertime, disponibles en la base de datos Million Song Dataset.

Un sistema de identificación de covers es aquel que determina, idealmente de manera automática, si una canción es un cover de alguna pieza musical que se encuentra dentro de una base de datos. Para abordar este problema los métodos utilizados en las distintas investigaciones se centran principalmente en dos etapas. En la primera etapa se procede a la extracción y análisis de las características principales de la canción, tales como su representación melódica, progresión armónica o pitch. En la segunda etapa se mide el grado de similitud que existe entre las características extraídas de cada pieza musical utilizando métodos como DTW (Dynamic Time Warping) o LSH (Locality Sensitive Hashing), entre otros. En este proyecto se aplicarán métodos de aprendizaje maquinal para la identificación de versiones o covers de piezas musicales.

[4] http://www.secondhandsongs.com/statistics

**Tabla 1.1:** Géneros musicales grabados para la canción “Summertime” disponibles en la base de datos Million Songs Datasets.

| Género | # de canciones |
|---|---|
| blues | 3 |
| country | 1 |
| jazz | 25 |
| pop | 1 |
| r&b | 5 |
| rock | 1 |
| rock pop | 11 |

La identificación de versiones musicales (covers), así como otros temas relacionados con el acceso, análisis y organización de todo tipo de contenido musical, han sido tareas a resolver que han generado gran interés entre los investigadores de disciplinas como la musicología, la ciencia cognitiva, las ciencias de la información y las ciencias de la computación. De ahí que el ISMIR organice, desde el año 2000, de manera anual su foro interdisciplinario para la investigación sobre el modelado, creación, búsqueda, procesamiento y uso de datos musicales. Motivados más que nada por el deseo de proporcionar a los amantes y profesionales de la música y a la industria musical, métodos y herramientas sólidas, eficaces y útiles para ayudar a localizar, recuperar y experimentar la música a la cual desean tener acceso.

Este creciente interés en investigar los aspectos relacionados con el contenido musical ha incrementado el número de tareas a atender. Una de las que ha captado el interés de algunos grupos de investigadores es la identificación de covers, principalmente por la implicación que tiene el hecho de comercializar música de manera ilegal, más estrictamente hablando en el sentido de realizar interpretaciones (modificadas o no) de música existente y obtener beneficios económicos a partir de éstas, sin otorgar las regalías y el reconocimiento pertinente al autor de la obra.

## 1.1 Justificación

Aunque se podría creer que un cover mantiene mucha relación con su versión original debido a que son en esencia una misma canción interpretada por un artista distinto o interpretada con diferentes arreglos, lo cierto es que esto último puede hacer de la identificación de covers una tarea difícil. Aun así, las diferentes versiones siguen manteniendo en lo general la melodía o armonía de la canción original. Estas dos últimas propiedades permitieron a los investigadores desarrollar métodos basados en la extracción de características de los chromas y algoritmos que ayudan a medir distancias o similitudes entre dos secuencias como DTW. Los chromas son vectores de 12 elementos donde cada uno de ellos representa la energía espectral correspondiente a un tono, utilizados para el análisis de octavas.

Aunque estos métodos funcionan bastante bien al comparar dos elementos, con bases de datos que contienen grandes cantidades de piezas musicales representadas como metadatos, los métodos que aplican DTW o alineación de cadenas resultaron ser ineficientes. Esto es debido a que requieren una cantidad significativa de recursos computacionales al estar comparando la canción original con cada una de las candidatas a cover de la misma. De este modo, para la identificación de versiones musicales se necesita un algoritmo que permita determinar si una canción es un cover de otra canción almacenada en una base de datos, y que a su vez este algoritmo realice los cálculos y la búsqueda en el menor tiempo posible. Este es un problema que aún se encuentra en espera de ser resuelto.

Son diversos son los motivos por los cuales un grupo musical o un cantante solista puede generar sus propias versiones a partir de canciones ya existentes. Entre estas razones se pueden destacar el realizar un homenaje al intérprete o compositor original, el ganar la audiencia a la que les gusten canciones similares o el incrementar las oportunidades de éxito mediante el uso de canciones que fueron exitosas. Es por eso que varios covers son creados día a día, esto sin la autorización previa del individuo que tenga protegida la obra o sin otorgar crédito al creador original, violando de este modo los derechos de autor. Para este caso se define derechos de autor como: “la

facultad exclusiva que tiene el creador intelectual para explotar temporalmente (por sí o por terceros) las obras de su autoría (facultades de orden patrimonial), y ser reconocido siempre como autor (facultades de orden moral) con todas las prerrogativas inherentes a dicho reconocimiento" de acuerdo con la Ley Federal del Derecho de Autor (LFDA), misma que a su vez otorga protección a los titulares de los derechos, las cuales faculta al Instituto Mexicano de Propiedad Intelectual (IMPI) de imponer multas de quinientos a diez mil días de salario mínimo a quienes hagan uso de la obra sin la debida autorización del titular de la misma, tal como se describe en [2].

Un ejemplo de esta situación se puede encontrar directamente en YouTube donde a diario se suben videos de covers sin que haya restricción alguna. En este caso si se hace una búsqueda de una canción, por ejemplo "Imagine" de John Lennon, los resultados nos dará una lista de covers realizados a dicha canción, tal como se muestra en la Figura 1.1. Dado que la popularidad de un video en YouTube se mide por el número de vistas, los videos mostrados en la figura le han dado visibilidad en internet a los intérpretes de un cover.

En este trabajo se muestra un sistema de identificación de versiones musicales (SIVM) cuya principal función es reconocer aquellas canciones que son una nueva versión o interpretación de una canción ya existente, siempre y cuando la versión original se encuentre almacenada en la base de datos que se consulte.

## 1.2 Objetivos de la investigación

El proyecto se desarrolló a partir de los siguientes objetivos:

- Objetivo General.

El objetivo principal de esta investigación es desarrollar un sistema de identificación de versiones musicales aplicando técnicas de aprendizaje maquinal.

- Objetivos específicos.
    - Seleccionar los componentes que permitan la separación de la melodía principal y acompañamiento para el análisis de piezas musicales.
    - Definir la arquitectura de aprendizaje maquinal para el reconocimiento e implementarla.
    - Realizar una primera etapa de pruebas del sistema de reconocimiento para ajustar los parámetros que así lo requieran.
    - Realizar los ajustes del sistema de reconocimiento, realizar las pruebas finales y evaluar su rendimiento.

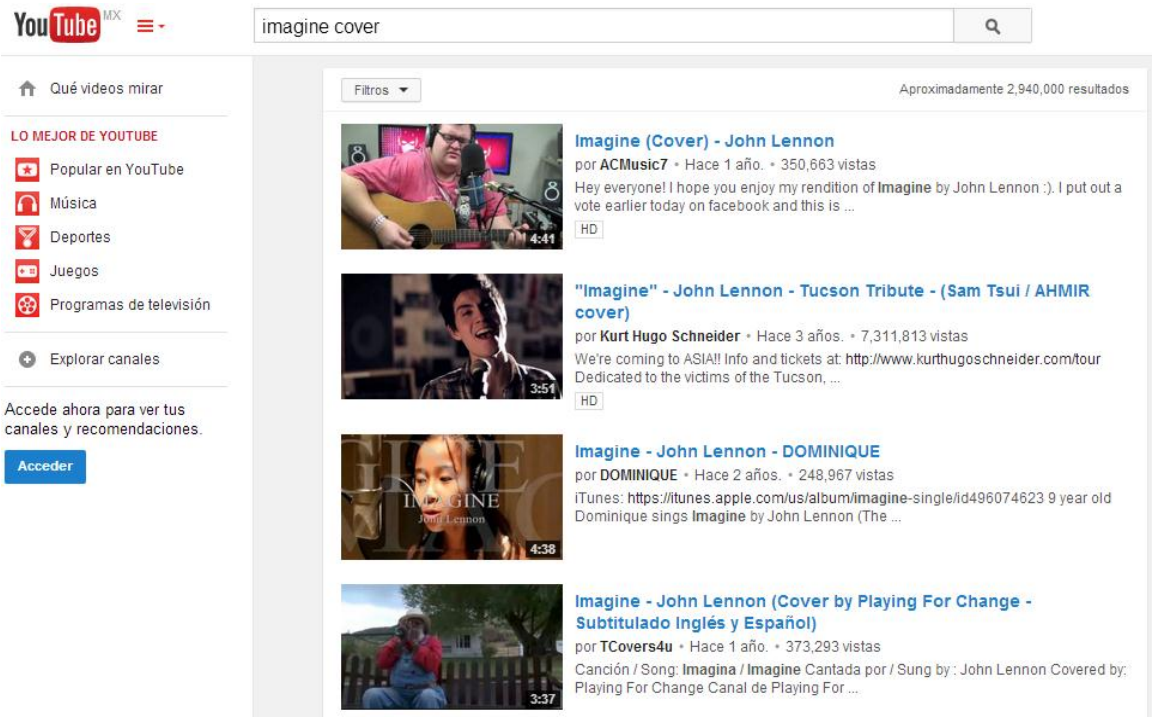



**Figura 1.1:** Ejemplo de búsqueda en YouTube de la canción “Imagine” de John Lennon.

CAPÍTULO 2

# ANTECEDENTES

Para el ser humano el proceso de determinar cuándo una canción es un cover, con base en una serie de canciones de las cuales tiene conocimiento, puede llegar a ser una tarea relativamente sencilla en la mayoría de las ocasiones. Sin embargo, encontrar un método que permita automatizar de manera óptima la identificación de covers resulta un gran desafío. Esto es principalmente porque una misma pieza musical interpretada en distintas versiones puede variar en el ritmo, tempo, instrumentación, género o duración de la canción, así como en las características intrínsecas de la voz del intérprete. Por tal motivo, la identificación de versiones musicales (covers) se ha convertido en la última década, en un reto a vencer para los estudiosos en el tema. De igual manera, se han desarrollado distintos trabajos en los cuales los investigadores han propuesto diversos métodos que ayudan a solventar este problema.

Dentro de los trabajos realizados previamente encontramos la investigación elaborada en [3] por Lee donde propone un método para extraer una secuencia de acordes para cada canción aplicando el algoritmo de Modelos Ocultos de Markov HMM, (Hidden Markov Model) y posteriormente midiendo el grado de similitud entre las secuencias de acordes mediante DTW. Por otro lado, en [4] se calculó un chromagrama[5] (matriz de características) que no es sensible a los cambios de

[5] Los chromagramas son secuencias de vectores de chromas.

instrumentación y tiempo para posteriormente obtener la distancia mínima, de las diferentes matrices, utilizando la norma de Frobenius[6]. De igual forma, en [5] proponen obtener un chromagrama para posteriormente calcular tres características por canción y clasificar mediante máquinas de soporte vectorial (SVM) o perceptrón multicapa (MLP). En [6] se propone calcular un chromagrama que conserva los armónicos parciales[7] de la melodía y mantiene la invariancia del volumen, y sobre esto se planea un marco de trabajo (framework) de medición de la similitud, con el cual se puede aplicar cualquier clasificador binario, para este proyecto se utilizó el método de Bayes.

En [7] se propone un método inspirado en la creación de huellas digitales para minimizar los tiempos de ejecución en la búsqueda de covers. Esta investigación hace uso por primera vez de la base de datos MSD (Million Song Dataset) la cual consta de características y metadatos para un millón de canciones bajo la licencia Creative Commons (CC). En [8] se da una introducción a MSD en donde se explica el proceso de creación, cómo está constituida y posibles usos para ésta. Más información sobre esta base de datos lo podemos encontrar en [9].

Dividir una canción en pequeños fragmentos (usados como hashes) es una técnica utilizada para acelerar el proceso de búsqueda en una base de datos, en [10] se experimenta con fragmentos más largos para cada canción y así minimizar el número de búsquedas.

En [11] se utiliza la transformada de Fourier bidimensional para obtener una representación de los parches chroma con el fin de obtener un algoritmo eficiente del vecino más cercano. Bajo este esquema los vecinos más cercanos tienen alta probabilidad de estar relacionados a una misma canción. Por otro lado, en [12] los

---

[6] La norma de Frobenius es la norma matricial de una matriz A (*m x n*) definida como la raíz cuadrada de la sumatoria del cuadrado absoluto de sus elementos.

[7] Un armónico es el resultado de una serie de variaciones adecuadamente acomodadas en un rango o frecuencia de emisión. Mientras que los armónicos parciales son los armónicos cuyas frecuencias son múltiplos enteros.

autores realizaron dos principales modificaciones al trabajo elaborado en [11] para mejorar la identificación de covers: primeramente hacer una dispersión para mejorar la separabilidad y después supervisar la reducción de dimensionalidad.

Dos métodos son propuestos en [13] y [14] que ayudan a realizar las consultas de una manera más rápida en la identificación de covers en una base de datos grande. En [13] se utiliza una técnica de indexación de bio-secuencia conocida como BLAST (Basic Local Alignment Search Tool) que permite aumentar significativamente la eficiencia en identificación de un cover. A su vez, en [14] se desarrolló un método para podar la base de datos, el cual reduce el conjunto donde se efectúa la búsqueda.

En [15] se utilizó el método LSH (Locality-sensitive hashing) para obtener canciones con acordes[8] similares y posteriormente aplicar una etapa de progresión para refinar el ranking de la búsqueda. Por otra parte en [16] se enfocan a extraer las representaciones tonales (la melodía[9], línea de bajo[10] y la progresión armónica[11]) mediante el uso algoritmos propuestos en el estado del arte, y utilizan un algoritmo de programación dinámica para medir el grado de similitud. Se concluye que la representación armónica es más fiable para la identificación del cover, sin embargo, el uso de las tres representaciones tonales que exponen mejoran considerablemente el grado de precisión en el reconocimiento.

Por otro lado, en [17] proponen utilizar tres nuevos descriptores para la recuperación musical basada en contenido. Dichos descriptores son: pitch

[8] Un acorde consiste en un conjunto de tres o más notas diferentes que suenan simultáneamente y que constituyen una unidad armónica.
[9] Una melodía es una sucesión de sonidos que es percibida como una sola entidad. Se desenvuelve en una secuencia lineal, es decir a lo largo del tiempo, y tiene una identidad y significado propio dentro de un entorno sonoro particular.
[10] Una línea de bajo es la sucesión de las notas más graves de un pasaje o composición. Cumple una función de soporte al resto de los elementos y establece la base de la progresión armónica.
[11] Una progresión armónica es una sucesión de acordes, explícitos o implícitos.

bihistogram[12], coeficiente de correlación de chroma[13] y características de armonización[14].

J. Serrà presenta un análisis en [18] donde menciona que los SIVM en su mayoría están constituidos principalmente por cinco etapas: extracción de características, invariancia de key, invariancia del tempo, invariancia de la estructura y cálculo de la similitud. Mientras que en [17] agrupan estos pasos en una arquitectura de dos etapas, en la primer fase se calcula la características armónica o del picth para cada canción y en la segunda fase se hace una comparación entre estas características para medir su grado de similitud.

De igual manera se debe tener en consideración que el correcto desarrollo de estos estudios implicó poseer una adecuada base de datos, la cual facilitara a los investigadores el proceso de obtención de toda la información necesaria, tanto para el desarrollo de los algoritmos como para la correcta evaluación de éstos. En este sentido, se han creado y puesto a disposición diferentes bases de datos para tratar de resolver el problema. En la Tabla 2.1 se muestra la comparación entre varias bases de datos con contenido musical disponibles en la actualidad, algunas de ellas otorgan solamente los metadatos que describen cada canción, mientras que otras cuentan con el audio disponible también para su descarga. Como se puede apreciar MSD cuenta con 1 millón de canciones de las cuales se extrajeron sus características acústicas mediante el uso de la Echo Nest API, para proporcionar de esta manera solamente los metadatos de las piezas musicales (ver anexo B para mayor información).

Como se había mencionada anteriormente el desarrollo de sistemas para la identificación de covers implica la realización de dos grandes etapas: una etapa de análisis de la información y otra de comparación de la información; para el

[12] Un pitch bihistogram es una representación bigrama que se puede calcular a partir del tono melódico continuo.
[13] Los coeficientes de correlación chroma es una matriz de 12x12 obtenida a partir de coeficiente de correlación calculados entre cada par de vectores chroma.
[14] Las características de armonización son un conjunto de histogramas de los tonos armónicos, ya que acompañan a cada tono melódico.

cumplimiento de los objetivos de esta investigación nos centramos en la ejecución de estas dos etapas. Partiendo de la base de datos MSD, tomaremos el subconjunto de canciones denominado Second Hand Songs (SHS), el cual se conforma por todas las canciones que tienen por lo menos un cover dentro del MSD. Con este conjunto de datos realizamos un análisis desde la perspectiva estadística y de procesamiento de señales y desarrollamos comparaciones de similitud basadas en diversas métricas.

**Tabla 2.1:** Tabla comparativa de distintas bases de datos con contenido musical, utilizadas en las distintas tareas concernientes a la recuperación musical. Referencia [8].

| dataset | # songs/samples | audio |
|---|---|---|
| RWC | 465 | Yes |
| CAL500 | 502 | No |
| GZTAN genre | 1,000 | Yes |
| USPOP | 8,752 | No |
| Swat10K | 10,870 | No |
| Magnatagatune | 25,863 | Yes |
| OMRAS2 | 50,000? | No |
| MusiCLEF | 200,000 | Yes |
| MSD | 1,000,000 | No |

Por último se muestra en la Tabla 2.2 resultados obtenidos en trabajos que utilizaron MSD.

**Tabla 2.2:** Tabla con resultados obtenidos en investigaciones que emplearon MSD.

| Referencia | Method | Mean Average Precision (%) |
|---|---|---|
| [7] | jcodes 2 | 0.21 |
| [11] | 2DFTM (200 PC) | 2.95 |
| [11] | 2DFTM (50 PC) | 2.00 |
| [11] | 2DFTM (10 PC) | 0.29 |
| [12] | k-LDA (50) | 13.41 |
| [12] | k-LDA (200) | 0.83 |
| [12] | k-pca(200) + LDA (200) | 12.76 |
| [15] | Chord Profiles | 3.71 |

CAPÍTULO 3

# METODOLOGÍA

Para el desarrollo del proyecto se tuvo en consideración la investigación realizada por Thierry Bertin-Mahieux, and Daniel P. W. Ellis [11]. Si bien hay trabajos desarrollados para clasificar los covers de una canción, los más notables son citados en el capítulo anterior, en este trabajo se utiliza la base de datos MSD mediante una metodología muy precisa que permite obtener un marco comparativo para el proyecto. Los autores enumeran un conjunto de procedimientos a seguir para la tarea de identificación de covers, los cuales se muestran a continuación:

1. Obtener las características chroma de MSD.
2. Alinear las características chroma con el beat.
3. Aplicar la ley de potencia sobre la matriz resultante.
4. Generar parches y calcular su transformada 2D de Fourier.
5. Calcular la mediana.
6. Aplicar PCA (Análisis de componente principales, por sus siglas en inglés).

Tomando como base el grupo de etapas mencionadas, la metodología seguida se muestra en las Figuras 3.1 y 3.2. De manera general se conformó una base de datos a partir del conjunto de canciones dado por SHS y después se procedió a generar las características del chroma que permitan encontrar un patrón que se asocie a cada una de las canciones que forman parte de los datos. El procedimiento se aplicó sobre las canciones de manera individual (Figura 3.1) y para el ensamble de canciones que pertenecen a la misma clase, es decir, las que tengan el mismo título (Figura 3.2).

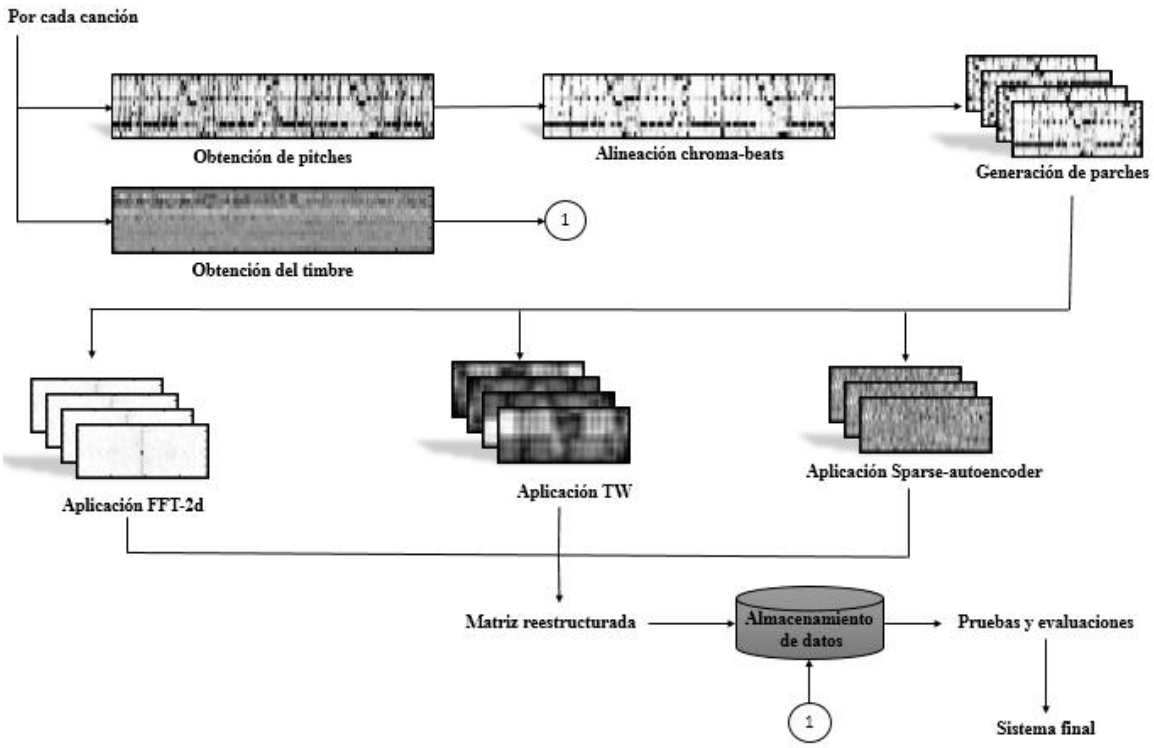



**Figura 3.1:** Metodología aplicada a las canciones de manera individual.

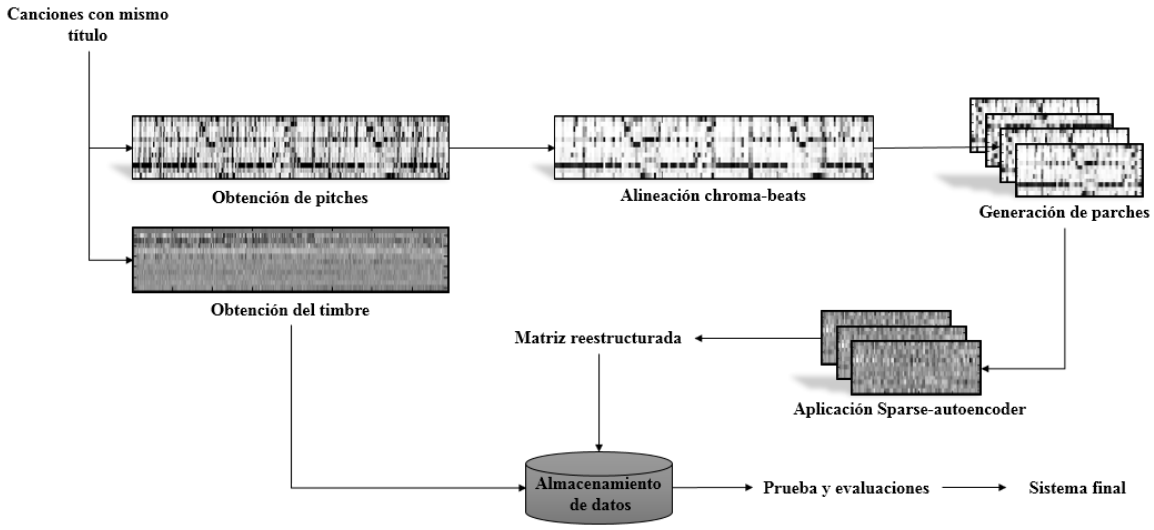



**Figura 3.2:** Metodología utilizada para el ensamble de canciones del mismo título.

Las características de las canciones se obtuvieron a partir de una adecuación de los datos del croma y de los datos del timbre, estos últimos tomados directamente de la base de datos. Posteriormente a las matrices resultantes se les aplicó un análisis espectral para obtener la transformada de Fourier bidimensional, la transformada

wavelet y los patrones generados a partir de la aplicación del algoritmo de denominado Sparse Autoencoder. El reconocimiento de covers se realizó a partir de la aplicación de métricas que evaluaron el grado de agrupamiento de los ensambles correspondientes a cada canción. A continuación se describen cada una de las etapas de la metodología utilizada.

## 3.1 Base de datos

Se descargó la base de datos MSD, la cual se creó a partir de los metadatos obtenidos de un millón de canciones, en su mayoría de diferentes títulos. De este conjunto se seleccionaron aquellas piezas musicales que comparten el mismo título, es decir, se obtuvo un conjunto de datos que incluyera la versión original de una pieza musical (si es que existe) y los covers que ésta tuviese. Para realizar esta tarea se hizo uso del conjunto de datos proporcionado por SHS, el cual es un proyecto fundado en 2003 por Bastien De Zutter, Mathieu De Zutter y Denis Monsieur con el objetivo de proporcionar una base de datos que estuviese constituida por el mayor número de interpretaciones originales, así como de los covers que se hayan generado a partir de las primeras [19]. Este proyecto facilita dos conjuntos de datos elaborados a partir de las canciones contenidas en MSD, un conjunto de entrenamiento y un conjunto de prueba, la Tabla 3.1 muestra la distribución de las canciones en ambos conjuntos. Se tomaron los datos correspondientes al conjunto de entrenamiento para el análisis de la información así como para la evaluación en las etapas posteriores durante el desarrollo del SIVM.

**Tabla 3.1:** Organización de los datos proporcionados por SHS. Cada título corresponde a un grupo de canciones integrado por la canción original y sus covers. Las interpretaciones son el número de canciones que cada conjunto posee.

| | # de títulos | # de interpretaciones |
|---|---|---|
| Conjunto de entrenamiento | 4,128 | 12,960 |
| Conjunto de prueba | 1,726 | 5,236 |
| **Total** | **5,854** | **18,196** |

## 3.2 Obtención de características

Cada pieza musical incluida en la base de datos consta de un conjunto de 55 atributos entre los que se encuentran descriptores informativos de la canción como nombre del artista, título de la canción, etiquetas con artistas similares, y descriptores acústicos de la canción como pitches, timbre, volumen y beat. Como primer paso se determinaron los atributos más representativos y la forma de manipular esta información.

Como se menciona en [8], [20] y con base en una serie de experimentos realizados por nuestra parte, se determinó utilizar las características acústicas pitches y timbre como base para el desarrollo de los sistemas de evaluación de los covers. La serie de experimentos elaborados consistieron en extraer las características más relevantes de las canciones que tienen un mayor número de covers y que se enlistan en la Tabla 3.2 y clasificarlas utilizando los algoritmos de aprendizaje maquinal Naive Bayes, MLP y SVM (en el Anexo A se encuentra una descripción de estos algoritmos). Inicialmente se utilizaron todos los atributos presentes en la base de datos, y posteriormente se fueron eliminando uno a uno en cada nuevo experimento para determinar cuál o cuáles de ellos impactaban en la correcta clasificación de las canciones. Como resultado se obtuvo que los atributos correspondientes a las características acústicas fueron los de mayor relevancia con respecto a la clasificación de las canciones, éstos se describen a continuación.

**Tabla 3.2:** Lista de canciones utilizadas en la experimentación que determinó el conjunto de atributos a utilizar. La primera columna muestra el título y la segunda el número de interpretaciones por cada título.

| Titulo de las canciones | # de interpretaciones |
|---|---|
| Summertime | 47 |
| Silent Night | 43 |
| White Christmas | 37 |
| Body and Soul | 19 |

- Pitches. Es una propiedad perceptiva que permite el ordenamiento de los sonidos en una escala relacionada con la frecuencia la cual se extiende de menor a mayor [21]. Para cada canción de la base de datos MSD, la información del pitch está dada por un conjunto de vectores chroma de longitud 12, que representa las notas con sus semitonos de acuerdo a la escala cromática: C, C#, D, D#, E, F, F#, G, G#, A, A#, B; a cada una de las notas se le asigna un valor numérico normalizado en el rango del 0 al 1 el cual describe el dominio relativo de cada pitch en cada evento musical (segmento de tiempo previamente determinado en la base de datos).

  Por ejemplo, en la Tabla 3.3 se pueden visualizar la manera en la que se representan los pitches mediante los vectores chroma en MSD para el primer segundo de la canción, donde en el segmento de tiempo que comienza en el segundo 0.40345 tienen un mayor dominio de las notas B, C#, G#.

**Tabla 3.3:** Vectores chroma del primer segundo de la canción "Summertime" interpretada por Ohio Players.

| Segmento de tiempo | C | C# | D | D# | E | F | F# | G | G# | A | A# | B |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.40345 | 0.553 | 0.667 | 0.362 | 0.161 | 0.142 | 0.159 | 0.071 | 0.247 | 0.635 | 0.259 | 0.423 | 1 |
| 0.57769 | 0.956 | 0.839 | 0.83 | 0.536 | 0.543 | 0.56 | 0.76 | 1 | 0.793 | 0.546 | 0.742 | 0.41 |
| 0.81537 | 0.268 | 0.464 | 0.847 | 0.367 | 0.407 | 1 | 0.532 | 0.51 | 0.483 | 0.414 | 0.717 | 0.293 |
| 0.94912 | 0.559 | 0.283 | 0.277 | 0.177 | 0.26 | 0.199 | 0.452 | 1 | 0.324 | 0.197 | 0.131 | 0.235 |
| | | | | | | ... | | | | | | |

- Timbre. Es la calidad de una nota musical o un sonido que distingue diferentes tipos de instrumentos musicales o voces [22]. En MSD es derivado de la superficie espectro-temporal de un segmento y es independiente del pitch o el volumen. Está representado por un vector de 12 valores o coeficientes sin límite centrados en cero (ver Tabla 3.4), donde los primeros cuatro coeficientes están relacionados con el promedio del volumen del segmento, el brillo, la planitud del sonido y el sonido con más fuerza [22], mientras que los ocho coeficientes restantes carecen de un significado concreto ya que la información proporcionada sobre lo que representan resulta ser imprecisa, motivo que nos

hizo prescindir de ellos y utilizar solamente los cuatro primeros coeficientes en los experimentos.

Se obtuvieron las matrices correspondientes del pitch y timbre de cada una de las canciones (original y covers) del conjunto de datos SHS y se procedió a su adecuación para el análisis.

**Tabla 3.4:** Valores de los doce coeficientes del timbre en el primer segundo de la canción "Summertime" interpretada por Ohio Players.

| Segmento de tiempo | coef1 | coef2 | coef3 | coef4 | coef5 | coef6 | coef7 | coef8 | coef9 | coef10 | coef11 | coef12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.40345 | 39.889 | 37.147 | 191.71 | -70.54 | 4.995 | 213.61 | 34.44 | 49.699 | -11.12 | 81.099 | 114.69 | -37.28 |
| 0.57769 | 46.689 | 22.626 | 144.94 | 35.009 | -43.47 | -12.82 | -28.72 | -18.58 | 8.907 | 19.546 | -28.26 | -3.863 |
| 0.81537 | 46.389 | 36.406 | 108.27 | 84.4 | -36.13 | -23.67 | 19.044 | -51.12 | 33.515 | 3.03 | -16.81 | -22.99 |
| 0.94912 | 47.046 | 44.802 | 144.96 | 16.764 | -23.41 | 12.116 | -9.351 | -10.82 | 26.372 | 3.366 | 1.599 | -6.609 |
| | | | | | | ... | | | | | | |

## 3.3 Alineación chroma-beats

En música el beat es una unidad básica empleada en la medición del tiempo, se trata de una secuencia constante de pulsaciones que se repiten y dividen el tiempo en partes iguales [23]. De esta manera al colocar la canción en una misma escala utilizando el beat, se logra superar la variabilidad en tiempo que las distintas versiones musicales pudiesen presentar, una propiedad que resulta importante en el reconocimiento de covers.

Con base en lo anterior para esta etapa de la investigación se realizó un alineamiento de los pitches con respecto a los beats proporcionados en MSD para cada canción. En la base de datos este atributo marca el tiempo en segundos en el cual inicia cada beat. Para realizar esta alineación se tomaron los vectores chroma que se incluyen en el segmento de tiempo que abarca cada beat y se les calculó su media ponderada para generar un vector chroma-beat. En la Figura 3.3 se ejemplifica gráficamente la manera en que fueron creados estos vectores.

Una vez generada la matriz alineada con los beats, ésta se ponderó elevándola a la potencia 1.96 para acentuar los principales patrones de la señal aumentando así el

contraste entre los valores chroma. Cabe mencionar que este valor fue reportado como el que mejor resultados arrojó en la experimentación de [11].

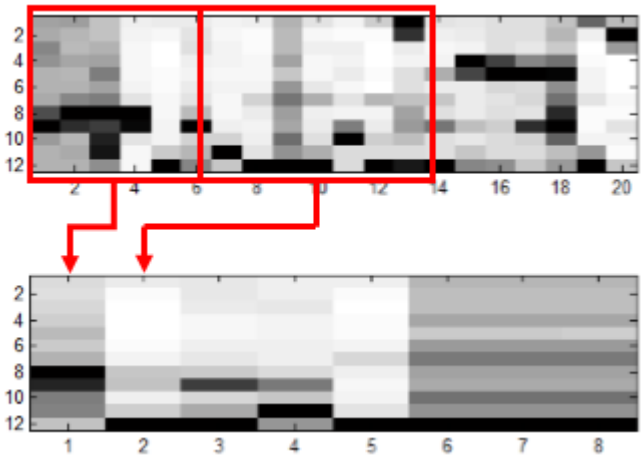



**Figura 3.3:** Matriz pitches (superior) y matriz alineada (inferior). Los cuadros en rojo enmarcan los segmentos de la matriz de pitches que abarca cada segmento de beat, y a los cuales se les calcula la media ponderada para generar el vector alineado correspondiente**.**

En la Figura 3.4 se muestran un ejemplo representativo del proceso de obtención de la matriz alineada para la canción "Sunny" de Christophe Willem, con una duración de 3.41 minutos. El chromagrama (a) muestra la matriz de pitches obtenida de MSD, donde los cuadros más oscuros representan las notas que más dominaron en cada segmento. A partir de ésta se forma la matriz alineada (b) donde se puede apreciar que otra ventaja de la alineación de los datos es la reducción de dimensiones, es decir, de estar constituida originalmente por 741 segmentos la matriz se reduce a los 441 segmentos del beats. El chromagrama (c) muestra la matriz alineada elevada a la potencia, lo cual permite acentuar los valores de las notas que más dominio tuvieron.

## 3.4 Generación de parches

El siguiente paso fue aplicar un método de segmentación que consiste en generar un conjunto de parches, dicho en otros términos, generar un conjunto de sub-matrices a partir de la matriz chroma-beats alineada.

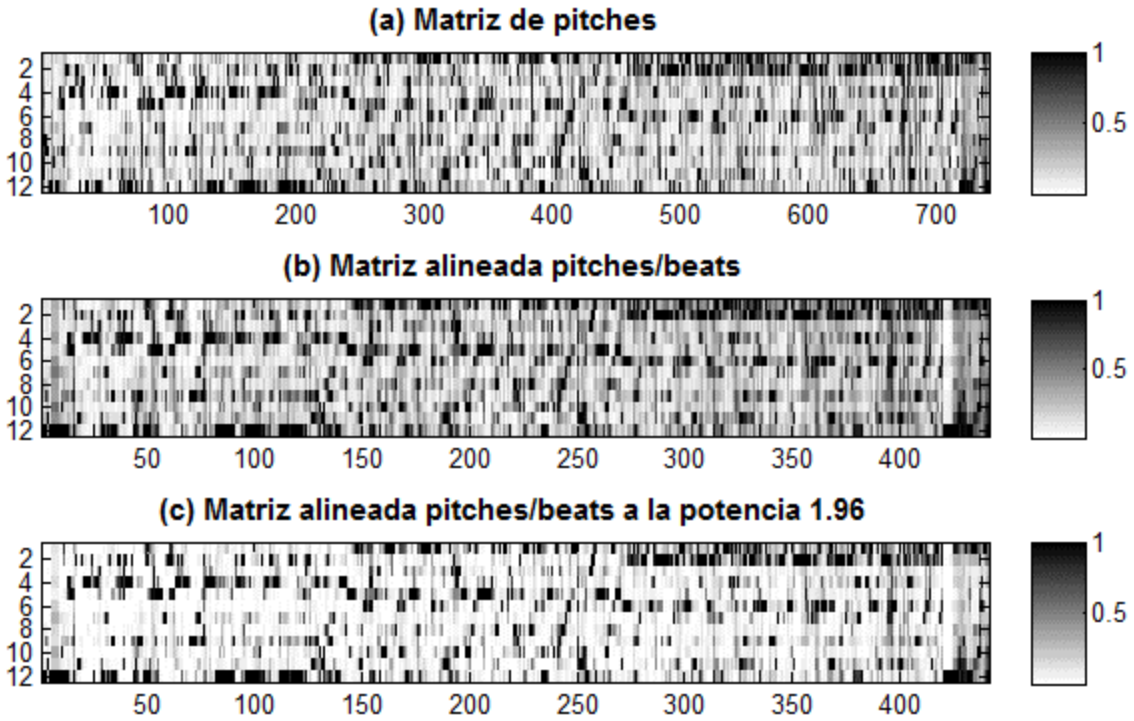



**Figura 3.4:** Proceso de alineación de pitches/beats para la canción "Sunny" interpretada por Christophe Willem. El eje Y representa los valores para cada vector chroma, mientras que el eje X presenta cada segmento que conforma la canción. (a) Matriz de pitches tal como se obtiene de la base de datos, (b) matriz alineada con los beats (c) matriz alineada elevada a la potencia 1.96.

El primer parche se tomó directamente de los primeros 75 vectores de la matriz alineada; se hizo un corrimiento de un vector a la derecha y se tomaron los 75 vectores siguientes, es decir del vector 2 al 76, para formar el segundo parche y así sucesivamente hasta terminar de recorrer la matriz, tal como se ejemplifica en la Figura 3.5. De esta manera se obtuvo un conjunto de M parches de 12x75. Cabe recalcar que se toman parches con longitud 75 por ser la medida con la que mejor resultados se obtuvieron en [11]. Asimismo, el número de parches por canción (M) varía de acuerdo a la duración de cada uno de los covers.

Este conjunto de parches generados conforman la base de los datos que se utilizaron para la aplicación de las transformaciones de Fourier bidimensional, Wavelet y el algoritmo Sparce Autoencoder, los cuales generaron los descriptores de

las canciones que se utilizaron para realizar las comparaciones entre las piezas musicales.

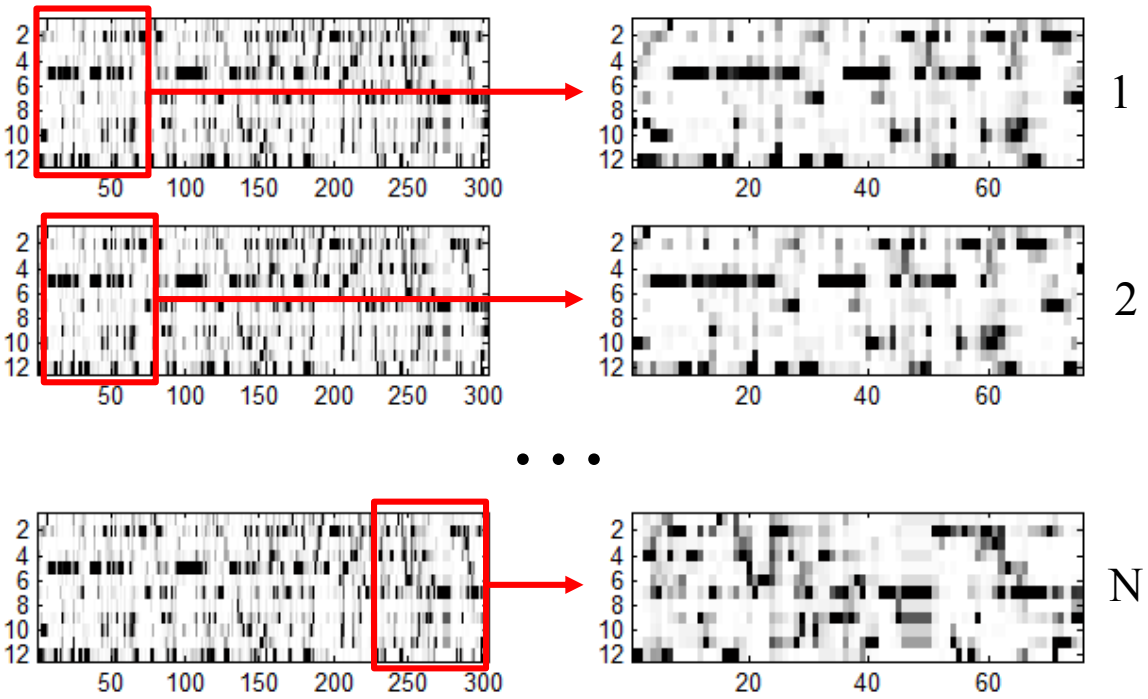



**Figura 3.5:** Proceso de obtención de los parches. La columna de la izquierda muestra la matriz alineada, el rectángulo rojo enmarca los 75 vectores del chroma-beats que se tomaron para generar cada parche. La columna derecha muestra cada parche generado.

## 3.5 Aplicación de Sparse Autoencoder

Como se describe en [32], una red neuronal autoencoder es un algoritmo de aprendizaje no supervisado que aplica la retropropagación (backpropagation) con la finalidad de establecer los valores de salida lo más parecido a los valores de entrada, intentando de esta manera reproducir sus propias entradas.

Es muy similar a un MLP, cuenta con una capa de entrada, una capa de salida y una o más capas ocultas que las conectan; la diferencia radica en que la capa de salida cuenta con el mismo número de nodos que la capa de entrada para tratar de reproducir los valores que entran a la red. Esta última característica permite que al establecer en

la capa oculta un número de nodos menor al número de nodos que posee la capa de entrada, la red se vea forzada a encontrar una representación distintiva de los datos, y a su vez de menor tamaño. Tal característica hace que el Autoencoder sea utilizado para aprender una representación comprimida del conjunto de datos, dicho de otra manera, es utilizado con propósitos de reducción de dimensionalidad; para mayor detalle ver anexo C.

Dadas las características que MSD presenta, las canciones con las que se encuentra constituida esta base de datos suelen poseer demasiada información en sus atributos pitches, lo que en ocasiones llega a ser un problema ya que impacta en los tiempos de cálculo cuando se implementa un algoritmo. Este problema nos llevó a ver como solución la implementación de Sparse Autoencoder[15] ya que su ejecución nos permitiría una reducción en la dimensionalidad de los datos y la obtención de un patrón mínimo representativo de clase a la cual éste se aplicara.

La implementación de Sparse Autoencoder se realizó sobre cada canción y sobre el conjunto de canciones con un mismo título, lo que nos llevó a obtener un patrón por canción y un patrón por título. Los pasos que se siguieron se describen a continuación:

a) *Obtención de patrones por canción*. Generados los N parches a partir de la matriz alineada de una determinada canción, se aplicó el algoritmo Sparse Autoencoder para obtener los patrones característicos de la canción. Primeramente se reestructuró el conjunto de parches a una matriz 900xN (donde 900 representa un parche de 12x75 convertido a vector y N es el número de parches). A esta nueva matriz se le aplicó el algoritmo Sparse Autoencoder, calculando así una nueva matriz de características de 900xM, donde M representa el número de patrones calculados a partir del algoritmo.

[15] La implementación del algoritmo se realizó con base en el ejercicio propuesto en: http://ufldl.stanford.edu/wiki/index.php/Exercise:Sparse_Autoencoder y en http://www.amolgmahurkar.com/learningfeatusingsparseautoencoders.html

Después de probar configuraciones que fueron de 30 a 120, se determinó utilizar una red de 30 neuronas en la capa oculta, por lo que se redujo la cantidad de N parches a M=30 patrones de 12x75 (esto reestructurando nuevamente la matriz de 900x30), que de acuerdo con la teoría del Autoencoder, representan las características de la pieza musical. En la Figura 3.6 se muestra gráficamente los procedimientos que se siguieron para esta etapa.

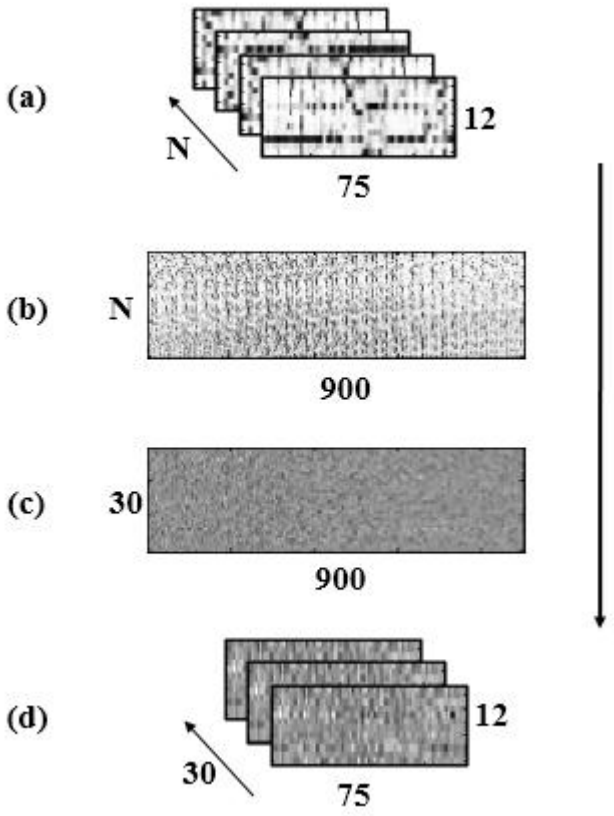



**Figura 3.6:** Representación gráfica del proceso de cálculo de patrones utilizando el algoritmo Sparse Autoencoder sobre el conjunto de parches. (a) Conjunto original de parches, (b) matriz generada a partir de los parches, (c) matriz de patrones generada al aplicar Sparse Autoencoder, (d) conjunto final de patrones generados a partir de la matriz de patrones.

b) *Obtención de patrones de las canciones con el mismo título*. Para generar el patrón general se obtuvieron todas aquellas canciones con un título igual y a ellas se les extrajo sus parches para reestructurarlos en la forma 900xN. A todo este conjunto de parches se le aplico el algoritmo Sparse Autoencoder para

generar un patrón general de 900x30. De igual manera se utilizó en la implementación una red de 30 neuronas en la capa oculta.

## 3.6 Cálculo de la trasformada de Fourier 2D

La transformada de Fourier convierte una señal representada en el dominio del tiempo al dominio de la frecuencia pero sin alterar su contenido de información, sólo es una forma diferente de representarla. En este contexto, la transformada de Fourier se suele utilizar para obtener todas aquellas las características frecuenciales con las cuales está compuesta una señal, permitiendo con esto obtener información de una manera mucho más clara. Es una herramienta de análisis muy utilizada en el campo científico (acústica, ingeniería biomédica, métodos numéricos, procesamiento de señal, radar, electromagnetismo, comunicaciones, etc.)[33]. Mayores detalles pueden verse en el anexo D.

La representación en frecuencia resulta atractiva para codificar una señal en un conjunto finito de coeficientes por lo que en esta etapa nos valimos de estas características para buscar y separar los patrones en diferentes niveles de detalle, algo relativamente útil ya que nos permite poder hacer coincidencias dentro del espacio euclidiano[16].

Sobre los N parches creados por cada canción, los cuales son matrices de 12x75 donde 12 describe los notas musicales existentes en una octava para ese segmento de tiempo y 75 constituye el número de beats tomados en cuenta para generar el parche, se tomaron uno por uno y se les realizó el cálculo de su transformada de Fourier bidimensional (FFT-2D), la cual es misma que la transformada de Fourier sólo que para un arreglo en dos dimensiones de datos, conocidos también como matrices.

[16] Un espacio euclidiano un espacio de dos o tres dimensiones en el cual rigen los axiomas y postulados de la geometría euclidiana; también, un espacio en cualquier número finito de dimensiones, en el cual los puntos están definidos por coordenadas (una por cada dimensión) y la distancia entre dos puntos está dada por una fórmula de distancia particular.

Calculada la FFT-2D para cada parche, se procedió a integrar los datos en una matriz de Nx900, donde cada fila representa un parche de 12x75 transformado a un arreglo de 900x1, y N continúa siendo el número de parches de la canción.

En la Figura 3.7 se representa este proceso de manera gráfica, aquí podemos ver el conjunto de parches y cómo quedan los datos contenidos en éstos una vez que se le ha aplicado la FFT-2D; posteriormente dentro de la misma imagen se muestra los parches integrados en una sola matriz, la cual sirvió de base para los experimentos.

A continuación se realizó el mismo procedimiento, sólo que esta vez sobre los 30 patrones de 12x75 obtenidos de la aplicación del Sparse Autoencoder sobre los datos; a cada patrón se le calculo su FFT2-D y se reestructuraron los datos a una matriz de 30x900, de la misma manera que se realizó con los parches.

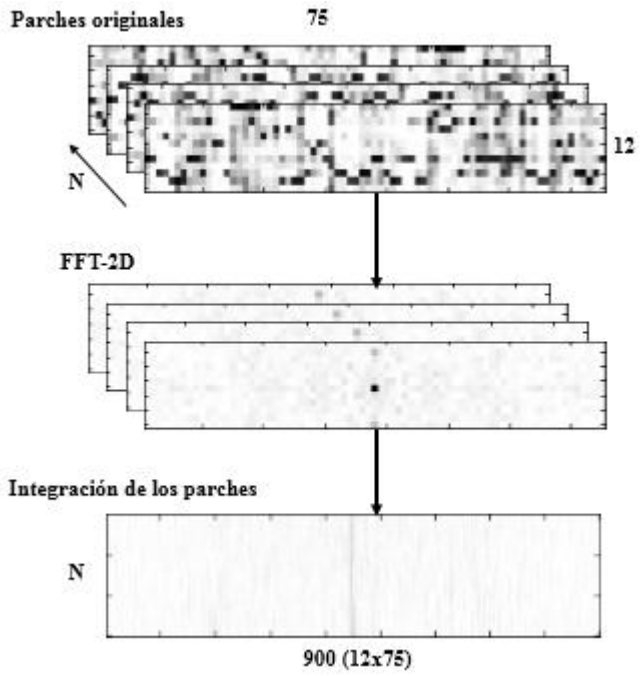



**Figura 3.7:** Obtención de la FFT-2D sobre los parches de una canción. En la parte superior se muestra el conjunto de parches de la canción, en la parte media se presenta la FFT-2D aplicada sobre cada parche y en la parte inferior las FFT-2D de los distintos parches integradas en una sola matriz.

## 3.7 Calculo de la transformada Wavelet

Dentro del análisis de señales otro tipo de transformada que podemos encontrar es la transformada Wavelet (TW), también llamada transformada onditas, las cuales son funciones que satisfacen ciertos requerimientos matemáticos y son utilizadas para la representación de datos o de otras funciones. Permiten generar datos a partir de una señal fraccionándolos en pequeños componentes de escala tiempo-frecuencia. Las wavelets son muy adecuadas para aproximación de datos con variaciones o con discontinuidades abruptas [34]. Para mayores detalles sobre la transformada wavelet véase el anexo D.

A fin de aprovechar las ventajas que la TW otorga se decidió aplicar esta transformada sobre los parches y los patrones generados en las secciones 3.4 y 3.5 respectivamente, de manera similar a como fue aplicada la FFT-2D. Para cada canción se tomaron los parches, los patrones y se les aplicó la TW para generar nuevos conjuntos de datos, en la Figura 3.8 se puede apreciar el procedimiento seguido para el caso de los parches. Las wavelets empleadas fueron: Morlet, Sinc, Gauss, Meyer y Daubechies.

## 3.8 Realización del sistema de reconocimiento de covers

Hasta este punto se realizó una adecuación a los datos y posteriormente un procesamiento para obtener características adecuadas para desarrollar un reconocimiento de covers basado en técnicas de aprendizaje maquinal.

Resumiendo lo realizado en las etapas anteriores, los datos generados son:

- Matriz de parches (900xN).
- Matriz alineada.
- Conjunto de patrones (por canción y por título) en forma de una matriz de 900x30, generados mediante el uso de Sparse Autoencoder.
- La FFT-2D de los parches en la forma de matriz de 900xN.
- La FFT-2D de los patrones en la forma de matriz de 900x30.

- La TW de los parches en la forma de matriz de 900xN.
- La TW de los patrones en la forma de matriz de 900x30.
- Matriz del timbre.

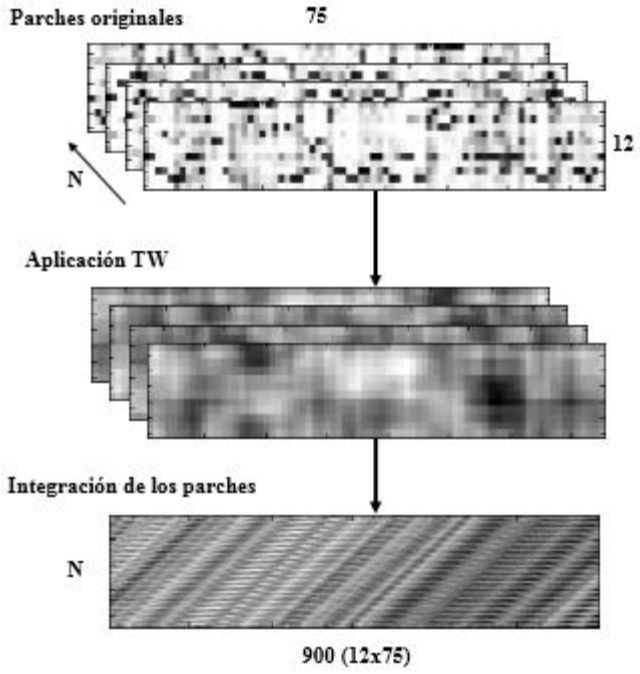



**Figura 3.8:** Obtención de la TW sobre los parches de una canción. En la parte superior se muestra el conjunto de parches de la canción, en la parte media se presenta la TW aplicada sobre cada parche y en la parte inferior las TW de los distintos parches integradas en una sola matriz.

Con estos datos se desarrollaron los siguientes procedimientos para definir el sistema de reconocimiento de covers:

a. Identificación de covers mediante la comparación con un ensamble general y la delimitación por umbral.
b. Identificación de covers mediante la comparación con un patrón general y la delimitación por umbral.
c. Identificación de covers con técnicas de aprendizaje supervisado y no supervisado.
d. Identificación mediante mapas auto-organizados (SOM) utilizando el timbre.

En los primeros procedimientos (puntos a y b) se trabajó en la obtención de características descriptoras y definición de un umbral de las interpretaciones con mismo título para ser utilizados en las comparaciones, es decir, en medición de similitud entre estas características y la canción a comparar. Mientras que en los procedimientos c y d las comparaciones de similitud se realizaron canción contra canción.

**a. Identificación de covers mediante la comparación con un ensamble general y la delimitación por umbral.**

Esta experimentación se dividió en dos etapas, en la primera etapa se obtuvo un ensamble general descriptivo de un grupo de canciones con un mismo título, aunado a esto se calculó un valor máximo para utilizarlo como umbral en la comparación de canciones. La segunda etapa consistió en realizar comparaciones para determinar si una pieza musical era o no un cover. Todo esto utilizando la matriz alineada y la matriz de parches, trabajando con ellas por separado.

En la Figura 3.9 se muestra el procedimiento seguido para la generación del ensamble general, donde se trabajó con canciones con un mismo título para extraerles la matriz alienada o de parches; con cada una de las matrices se calcularon medidas estadísticas (media, mediana o desviación estándar según el tipo de medida con el cual se pretendiera trabajar). Por último el ensamble general se obtuvo del promedio de las medidas estadísticas.

Obtenido el ensamble general se procedió a obtener los valores máximos. En la Figura 3.10 se muestra la descripción gráfica del procedimiento en el cual se calcularon nuevamente las medidas estadísticas de la matriz alineada o de parches para obtener su distancia con respecto al ensamble general; se utilizaron tres tipos de cálculo de distancia: resta, distancia Mahalanobis y distancia euclidiana, con la finalidad de identificar cuál se ajustaba de mejor manera a los objetivos. Calculadas todas las medidas se obtuvo el valor máximo para cada tipo de distancia, el cual definimos como umbral.

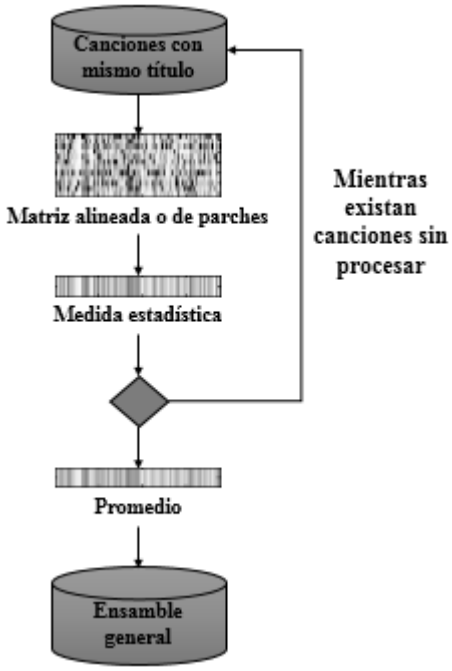



**Figura 3.9:** Procedimiento realizado para la generación del ensamble general, con piezas musicales de un mismo título.

Para la segunda etapa se realizaron pruebas implementando el procedimiento mostrado en la Figura 3.11, utilizando grupos de comparación constituidos por canciones con covers y canciones con títulos ajenos a éstas. En cada prueba se seleccionaba un grupo de comparación el cual indicaba el título del cover y las canciones a comparar. Del primero se toma el ensamble general y el umbral y con las canciones se mide la similitud calculando su distancia con respecto al ensamble general, utilizando resta, distancia Mahalanobis y distancia euclidiana, donde aquella canción con menor distancia es considerada como un cover de la canción comparada siempre y cuando se encuentre por debajo del umbral establecido.

Estas pruebas permitieron determinar el grado de similitud entre el ensamble general y otros covers con el mismo título que no fueron incluidas en la creación del ensamble. También dejaron ver qué tan factible era el uso de un umbral en la identificación de covers.

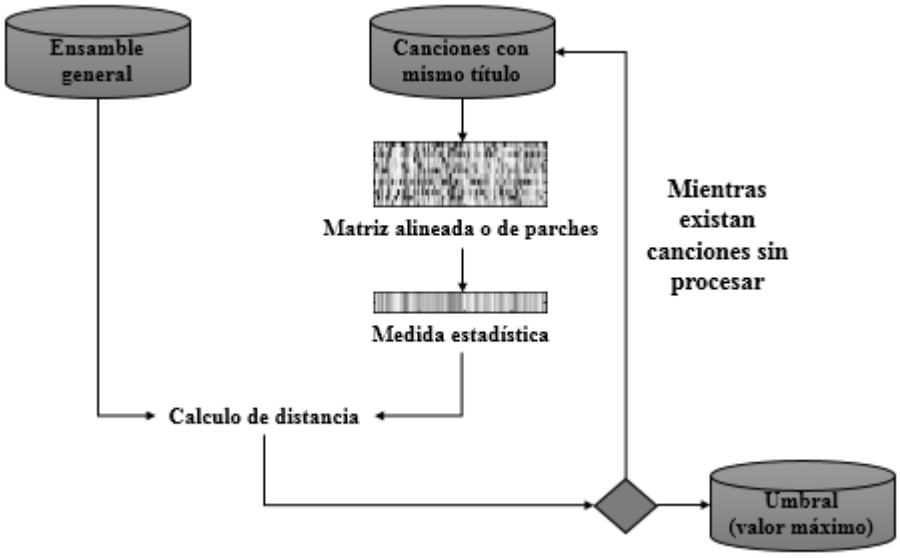



**Figura 3.10:** Procedimiento realizado para la obtención del valor máximo, entre piezas musicales del mismo título.

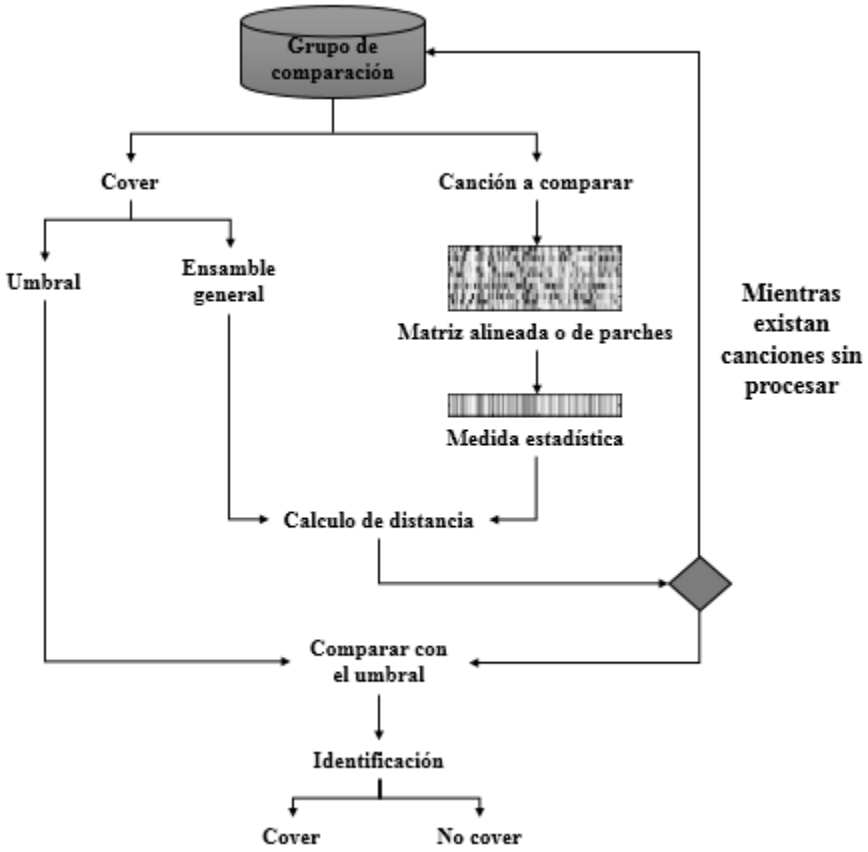



**Figura 3.11:** Método utilizado en la comparación de canciones, para la segunda etapa de la experimentación.

**b. Identificación de covers mediante la comparación con un patrón general y la delimitación por umbral.**

Esta experimentación se realizó en dos etapas de manera semejante a la descrita en la sección anterior. En la primera etapa se obtuvieron los patrones generales (de un grupo de canciones con un mismo título) y el umbral a utilizar. La segunda etapa consistió en realizar comparaciones para determinar si una pieza musical era o no un cover, utilizando para esto el patrón general y el patrón de cada canción comparada, generados con el algoritmo Sparse Autoencoder como se fue descrito en la sección 3.5.

Para generar los umbrales se calculó una medida estadística de los patrones y con ella se obtuvo la distancia de cada canción con respecto a su patrón general, aquella canción con mayor distancia es la que definió el umbral a utilizar, como se muestra en la Figura 3.12. Las medidas estadísticas empleadas fueron la media, mediana y desviación estándar, por lo cual se generaron tres tipos de umbrales para aplicar en diferentes pruebas; uno utilizando la media, otro utilizando la mediana y uno ultimo aplicando la desviación estándar.

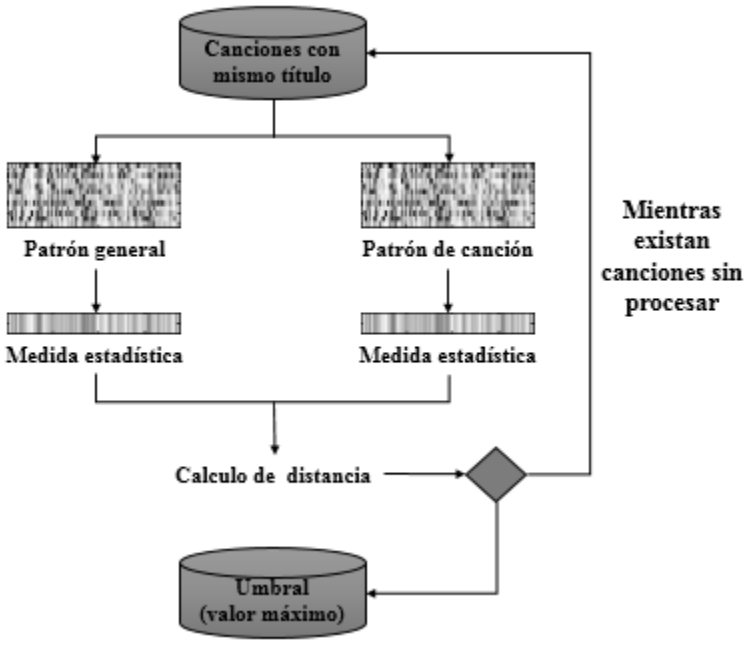



**Figura 3.12:** Pasos utilizados para obtener el umbral a partir de los patrones de las canciones del mismo título.

En la etapa de comparaciones para la identificación de covers se utilizó el método descrito en la Figura 3.13. Para cada prueba se obtuvo el patrón general y el patrón de cada una de las canciones a comparar para calcularles una medida estadística y utilizarlas en la medición de distancias, empleando la distancia coseno, cityblock y euclidiana. Si la distancia más corta con respecto al patrón general se encontraba por debajo del umbral, la canción era considerada como cover.

Concluida esta experimentación se volvió a realizar el mismo procedimiento, pero aplicando la entropía local sobre los patrones antes de calcular la medida estadística. En música la entropía es utilizada para medir el grado de incertidumbre y el contenido de la información de la fuente [35]. En la aplicación de nuestra investigación nuestra hipótesis fue que los covers poseerían una entropía similar a diferencia de aquellas que no son covers.

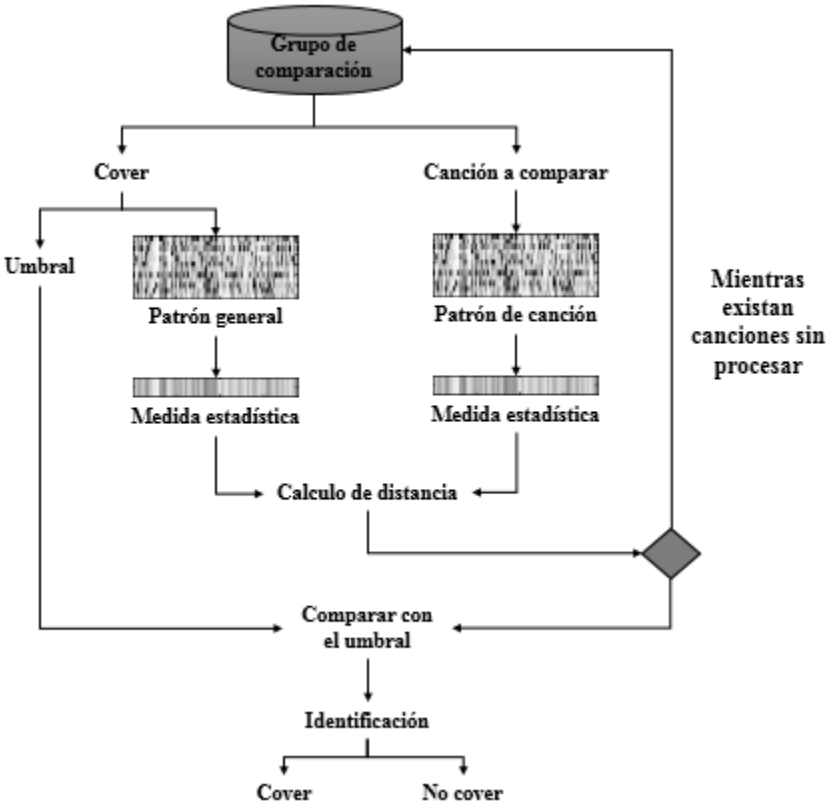



**Figura 3.13:** Procedimiento seguido en las pruebas de comparación.

**c. Identificación de covers con técnicas de aprendizaje supervisado y no supervisado.**

En esta sección se clasificaron las canciones con los datos generados con la FFT-2D y la TW, utilizando los algoritmos de aprendizaje no supervisado "K-vecinos más cercanos (k-nn)" y el supervisado "perceptrón multicapa (MLP)" como se describe a continuación.

*Método k-nn (k- vecinos más cercanos).* En reconocimiento de patrones k-nn es un método no paramétrico utilizado para la regresión y la clasificación, basado en medidas de similitud, por ejemplo funciones de distancia, tal como se define en [42]. Para nuestra investigación se aplicó de la siguiente manera:

1. Tomando la FFT-2D de los parches en la forma 900xN (de cada pieza musical) se calculó la mediana de éstos, resultando en un vector de 900x1 por canción.
2. Mediante un k-nn se identificaron los elementos más cercanos al elemento puesto a prueba utilizando las distancias: Euclidiana, SEuclidiana, City-block, Minkowski, Chebyshev, Coseno, Correlación, Spearman. Ver el anexo F con la descripción de las distancias.
3. Se repitieron los pasos 1 y 2, sobre la FFT-2D de los patrones, la TW de los parches y la TW de los patrones.
4. Se hicieron comparaciones para determinar cuáles características permitían un mejor agrupamiento de los datos, es decir, con qué configuración se obtenía un agrupamiento de covers.

En la Figura 3.14 se muestra gráficamente el procedimiento descrito anteriormente.

*Red neuronal Perceptrón multicapa (MLP).* Un MLP es un modelo de red neuronal artificial que mapea conjuntos de datos de entrada en un conjunto de salidas apropiadas. En la investigación se aplicó de la siguiente manera (Figura 3.15):

1. Se tomó la FFT-2D de los parches en la forma 900xN (de cada pieza musical a comparar).

2. Se aplicó la red MLP, entrenándola con un cover y varios no covers.
3. Se probó el sistema con un cover para ver si la red la clasificaba como tal.
4. Se repitieron los pasos 1, 2 y 3, ahora sobre la FFT-2D de los patrones, la TW de los parches y la TW de los patrones, respectivamente.
5. Se hicieron comparaciones para determinar cuáles características permitían una mejor clasificación de los datos, es decir, con qué configuración se obtenía una asignación correcta de covers.

Esto ayudó a obtener resultados que permitieron ver que método de clasificación resultaba más conveniente para nuestros datos.

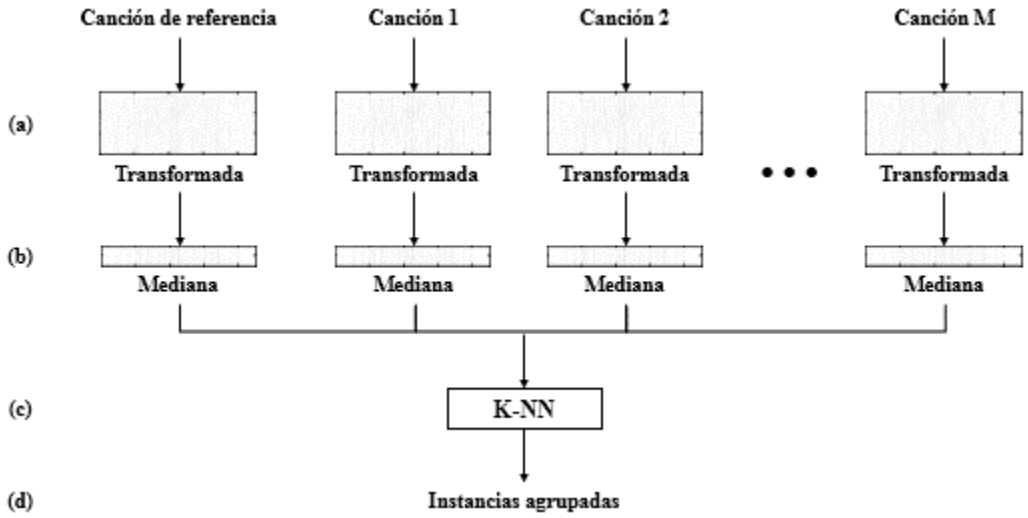



**Figura 3.14:** Método empleado para la clasificación con K-nn. (a) es la transformada FFT-2D o TW de los parches o patrones, de acuerdo con la matriz de características con la que se esté trabajando. (b) es un vector con de las medianas calculadas a partir las transformadas. (c) representa la aplicación del algoritmo k-nn sobre el conjunto de medianas. (d) es la lista de vecinos más cercanos a la canción de referencia**.**

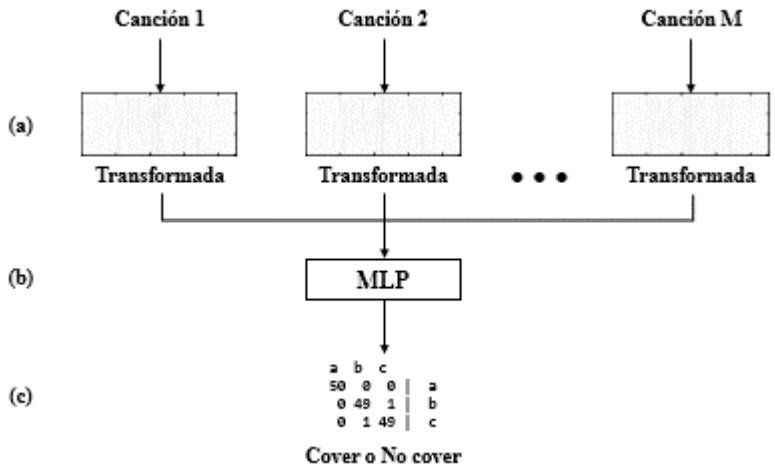



**Figura 3.15:** Método empleado para la clasificación utilizando un MLP. (a) es la transformada FFT-2D o TW de los parches o patrones, de acuerdo con la matriz de características con la que se esté trabajando. (b) representa la aplicación del algoritmo MLP sobre el conjunto de matrices. (c) muestra el resultado de la clasificación.

### d. Identificación mediante mapas auto-organizados (SOM) utilizando el timbre.

Los mapas auto-organizados (SOM), conocidos comúnmente como red de Kohonen son un método computacional para la visualización y análisis de datos de alta dimensión [43]. Son un medio de organización automático de los datos de modo que las entradas similares sean mapeadas al estar cerca entre sí (Ver anexo G para una mayor comprensión).

Para nuestra investigación los SOM se aplicaron sobre los cuatro primeros coeficientes de la matriz de timbres obtenida directamente de la base de datos, sin realizarles algún tipo de procesamiento. En la Figura 3.16 se presenta el procedimiento aplicado en esta etapa. En este caso el objetivo fue observar la manera en que las características del timbre se agrupaban cuando se tenían ejemplos de canciones del mismo título y si había diferencia entre covers y no covers.

Todos estos experimentos ayudaron a identificar las características que describían los covers de una mejor manera, con lo que se pudo apreciar que la TTF-2D aplicada

sobre los parches o sobre los patrones describe mejor una canción. Por otro lado el uso de un k-nn mejora los resultados en comparación con otras técnicas.

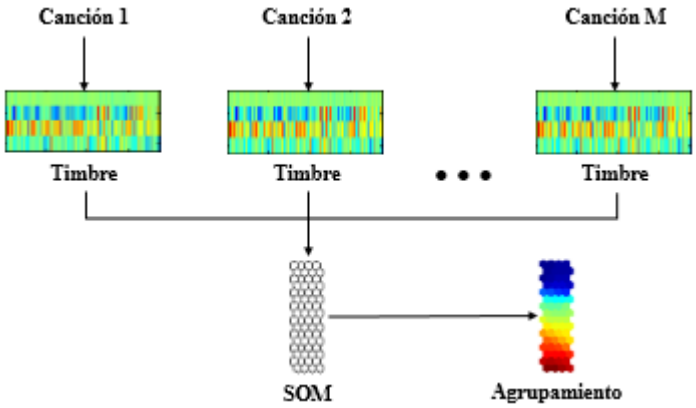



**Figura 3.16:** Procedimiento utilizado en la aplicación de los SOM.

CAPÍTULO 4

# RESULTADOS

En este capítulo se exponen los resultados generados al aplicar los procedimientos planteados en la sección 3.8 de la metodología para definir el sistema de reconocimiento de covers. Se muestran únicamente los resultados considerados como más relevantes de las series de pruebas elaboradas.

Para abordar esta sección se comienza dando una explicación de la estructura de los datos utilizados en las pruebas y posteriormente se muestran los resultados obtenidos para cada prueba de los procedimientos empleados.

## 4.1 Estructura de los datos

Para evaluar la efectividad de los procedimientos descritos en el capítulo anterior se creó un subconjunto de datos para análisis y un conjunto de grupos de comparación para pruebas, los cuales fueron utilizados en la ejecución de cada experimento.

En la creación del subconjunto de datos se utilizaron aquellas canciones que poseían más de 15 covers dentro de la base de datos MSD, auxiliándonos para esto con el subconjunto SHS. Este subconjunto de datos se puede apreciar en la Tabla 4.1, donde en total tenemos 417 covers que fueron realizados para 19 títulos diferentes.

**Tabla 4.1:** Subconjunto de canciones que más covers poseen en MSD. En la tabla se puede apreciar el título de cada grupo de canciones, así como el número de covers que incluye cada título.

| # | Titulo de las canciones | # de versiones |
|---|---|---|
| 1 | Summertime | 47 |
| 2 | Silent Night | 42 |
| 3 | White Christmas (LP Version) | 37 |
| 4 | Unchained Melody | 26 |
| 5 | I'll Be Home For Christmas (Album Version) | 22 |
| 6 | The Christmas Song (Chestnuts Roasting On An Open Fire) | 21 |
| 7 | St. Louis Blues (Album Version) | 20 |
| 8 | Over The Rainbow | 19 |
| 9 | Body And Soul | 19 |
| 10 | The Way You Look Tonight | 18 |
| 11 | Louie Louie | 18 |
| 12 | Stand By Me (Album Version) | 18 |
| 13 | Will You Love Me Tomorrow (LP Version) | 18 |
| 14 | Santa Claus Is Coming To Town | 16 |
| 15 | Sunny | 16 |
| 16 | People Get Ready (Album Version) | 15 |
| 17 | Smile | 15 |
| 18 | My Way | 15 |
| 19 | Save the Last Dance for Me | 15 |
| **Total** | | **417** |

Por otro lado, dado que la idea es identificar el cover de una canción con respecto a canciones de otros títulos, se determinó que cada uno de los grupos del conjunto de comparación estén constituidos de la siguiente manera (ver Figura 4.1):

- Un identificador que indica el número del grupo.
- Dos covers de la misma canción, elegidas aleatoriamente del subconjunto de canciones mostrado en la Tabla 4.1.
- Tres canciones de diferentes títulos, seleccionadas aleatoriamente del subconjunto de entrenamiento de SHS.

Estos grupos fueron utilizados con el objetivo de generar un porcentaje de precisión para los algoritmos elaborados, por lo que cada procedimiento hizo uso de

los mismos grupos de comparación, que en total fueron 1000, sin alteración alguna en las canciones seleccionadas.


**Figura 4.1:** Ejemplo de la estructura de un grupo de comparación, utilizado en las pruebas.

## 4.2 Resultados experimentales

En esta sección se muestran los resultados de cada uno de los experimentos descritos en la sección 3.8, siguiendo el mismo orden en el que fueron mencionados.

**a. Identificación de covers mediante la comparación con un ensamble general y la delimitación por umbral.**

Esta experimentación consistió en obtener ensambles generales que utilizamos como características representativas de los covers con un mismo título; de igual manera se calculó un umbral que envolviera las piezas musicales utilizadas en la creación del ensamble, como se describen en las Figuras 3.9 y 3.10.

Para comenzar con la experimentación se crearon los ensambles generales y los umbrales a utilizar, para los títulos de la Tabla 4.1. En la Figura 4.2 se muestra cómo están distribuidos los valores obtenidos del cálculo de las distancias (resta, Mahalanobis y euclidiana) entre las medias de cada matriz alineada y el ensamble general de los covers de Summertime; donde a partir de estos resultados el valor más alto obtenido representa el umbral tomado para las comparaciones.

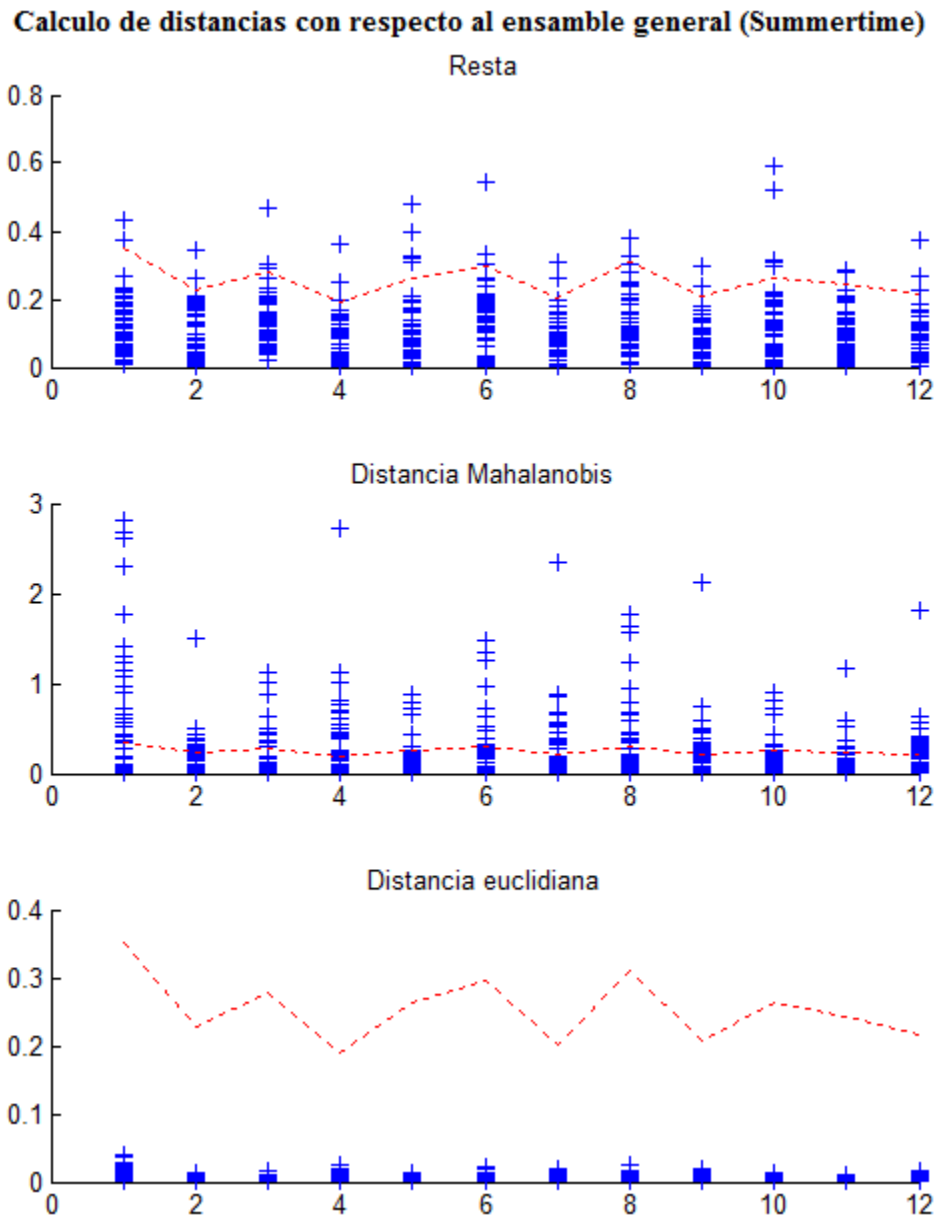



**Figura 4.2:** Valores generados (eje x) a partir del cálculo de las distancias entre la media de cada matriz alineada y el ensamble general (línea punteada color rojo) de los 47 covers de Summertime.

Para la segunda etapa de la experimentación se realizaron pruebas de comparación, como se describe en la Figura 3.11. Estas consistieron en dos comprobaciones de mil pruebas cada una.

- Identificación de los no covers. En esta comprobación se trató de identificar las canciones que no eran covers, por lo que ninguna de las canciones a comparar compartía el mismo título entre sí. En cada prueba se tomó un grupo de comparación (definidos en la sección 4.1) donde las dos primeras canciones definían el título del cual se tomaría el ensamble y las tres restantes se utilizaron para realizar las comparaciones. Si ninguna de las distancias calculadas entre el ensamble y las canciones que no eran covers estaba dentro del umbral, la prueba era calificada como correcta. El algoritmo tuvo un 6.2% de aciertos sobre las mil pruebas que se realizaron, utilizando la resta como medida de distancia, tal como se aprecia en la Tabla 4.2.

- Identificación del cover. En esta comprobación se trató de identificar el cover en cada prueba realizada. En cada grupo de comparación se tomó el primer cover para obtener el ensamble general y el umbral, y con el segundo cover y dos canciones no cover diferentes entre sí se realizaron los cálculos de medición de distancia. Si el cover era la que poseía menor distancia con respecto al ensamble general y además se encontrada dentro del umbral, la prueba era calificada como correcta. En la Tabla 4.3 se muestran los mejores resultados de las mil comparaciones, logrando identificar un 57% de los covers utilizando la matriz alineada y la resta como cálculo de distancia.

**Tabla 4.2:** Mejores resultados en la identificación de canciones no covers.

| Resta (%) | Distancia Mahalanobis (%) | Distancia euclidiana (%) | Datos utilizados | Medida Estadistica |
|---|---|---|---|---|
| 6.2 | 0.3 | 1.5 | Matriz alineada | Media |
| 6.2 | 0.2 | 1.4 | Parches | Media |
| 5.7 | 0 | 2.5 | Parches | Mediana |
| 5.2 | 0 | 3.5 | Matriz alineada | Mediana |
| 1.7 | 0.1 | 1.1 | Matriz alineada | SD |
| 1.1 | 0 | 0.9 | Parches | SD |

**Tabla 4.3:** Mejores resultados en la identificación de canciones covers.

| Resta (%) | Distancia Mahalanobis (%) | Distancia euclidiana (%) | Datos utilizados | Medida Estadistica |
|---|---|---|---|---|
| 56.8 | 29.1 | 42.9 | Matriz alineada | Media |
| 54.8 | 25.6 | 45.8 | Matriz alineada | Mediana |
| 53.4 | 27.5 | 42.4 | Parches | Media |
| 53.3 | 24.1 | 45.2 | Parches | Mediana |
| 42.6 | 36.2 | 39.6 | Matriz alineada | SD |
| 39 | 37.2 | 42.4 | Parches | SD |

**b. Identificación de covers mediante la comparación con un patrón general y la delimitación por umbral.**

En esta sección se muestran los resultados obtenidos al aplicar el procedimiento descrito en la Figura 3.13, dicho experimento consistió en la comparación de canciones con respecto a un patrón general y el uso de un umbral para identificar si eran o no covers del conjunto de canciones a partir del cual se obtuvo el patrón general. En esta experimentación se realizaron dos comprobaciones de mil pruebas cada una, los cuales consistieron en lo siguiente:

- Identificar los no covers. En esta comprobación se trató de identificar aquellas canciones que no eran covers, por lo que ninguna de las canciones a comparar compartía el mismo título entre sí. En cada prueba se tomó un grupo de comparación (definido en la sección 4.1) donde las dos primeras canciones definían el título del cual se tomaría el patrón general (o entropía del patrón general) y el umbral, y las tres canciones restantes se utilizaron para realizar las comparaciones. Si ninguna de las distancias calculadas entre el patrón general y los patrones de las no covers (o entropías) estaba dentro del umbral, la prueba era calificada como correcta. El algoritmo tuvo un 2.4% de aciertos sobre las mil pruebas que se realizaron, utilizando la entropía generada a partir de los patrones, como se muestra en la Tabla 4.4.

**Tabla 4.4:** Mejores resultados en la identificación de no covers, utilizando patrones y la entropía de los patrones.

| Distancia coseno (%) | Distancia cityblock (%) | Distancia euclidiana (%) | Datos utilizados |
|---|---|---|---|
| 0 | 2.4 | 2.4 | Desviacion estandar de entropia |
| 1 | 0.7 | 0.5 | Desviacion estandar de los patrones |
| 0.1 | 0.3 | 0.1 | Mediana de patrones |

- Identificar los covers. En esta comprobación se trató de identificar el cover para cada prueba realizada. En cada grupo de comparación se tomó el primer cover para obtener el patrón/entropía general y el umbral, con el segundo cover y dos no covers se realizaron los cálculos de medición de distancia. Si la canción que poseía el mismo título al del patrón general era el que se encontraba a menor distancia y además se estaba dentro del umbral, la prueba era calificada como correcta. En la Tabla 4.5 se muestran los mejores resultados para las mil pruebas realizadas, logrando identificar un 57.8% de los covers utilizando la desviación estándar de la entropía.

### c. Identificación de covers con técnicas de aprendizaje supervisado y no supervisado.

Este experimento consistió en clasificar como cover o no cover una canción mediante la implementación de los algoritmos k-nn y MLP de manera independiente utilizando la FFT-2d y la TW de los parches y patrones. En cada prueba de los grupos de comparación se utilizó un cover y dos canciones de diferente título para entrenar nuestros algoritmos, posteriormente el sistema clasificó el cover restante, los procedimientos seguidos corresponden a los descritos en las Figuras 3.14 y 3.15.

**Tabla 4.5:** Mejores resultados en la identificación de covers, utilizando patrones y la entropía de los patrones.

| Distancia coseno (%) | Distancia cityblock (%) | Distancia euclidiana (%) | Datos utilizados |
|---|---|---|---|
| 36.1 | 57.8 | 57.4 | Desviacion estandar de entropia |
| 31.7 | 34.8 | 34.6 | Media de entropia |
| 31.9 | 34.6 | 34.9 | Mediana de entropia |

Los resultados que se obtuvieron para las mil pruebas elaboradas al implementar el algoritmo k-nn se pueden visualizar en la Tabla 4.6; mientras que los resultados generados por el MLP se muestran en la Tabla 4.7.

**Tabla 4.6:** Tabla con los resultados obtenidos por k-nn. Los datos están expuestos en porcentajes de aciertos.

| euclidiana | minkowski | cosine | correlation | Matriz | Tipo de wavelet |
|---|---|---|---|---|---|
| 61.5 | 61.5 | 64 | 63.3 | FFT-2d patron | |
| 60.2 | 60.2 | 63 | 56.5 | FFT-2d parche | |
| 42.4 | 42.4 | 40 | 32.9 | TW patron | sinc |
| 44.1 | 44.1 | 40.8 | 31.6 | TW patron | gaus |
| 45.3 | 45.3 | 40.5 | 34.6 | TW patron | morl |
| 41.8 | 41.8 | 38.9 | 35.3 | TW parche | sinc |
| 46.3 | 46.3 | 43.4 | 33.3 | TW parche | gaus |
| 41.8 | 41.8 | 39.3 | 33.2 | TW parche | morl |

**Tabla 4.7:** Resultados obtenidos en la implementación del MLP.

| % de aciertos | Matriz |
|---|---|
| 49.5 | FFT-2d patron |
| 47.5 | FFT-2d parche |
| 36.6 | TW parche |
| 36.2 | TW patron |

**d. Identificación mediante mapas auto-organizados (SOM) utilizando el timbre.**

Para realizar cada una pruebas utilizando los SOM, se tomó de cada grupo de comparación uno de los covers y un no cover para entrenar el algoritmo, con el cover restante el sistema debía de decidir si era o no cover, esto generó un resultado de 48.4% de aciertos para las mil pruebas realizadas.

En la Figura 4.3 se puede apreciar la manera en que los datos se agruparon para una prueba que se consideró como una clasificación errónea. En la U-matriz (a) se muestra la forma en que se agrupan una canción cover y otra que no lo es. Mientras que en (b) se distingue el agrupamiento de dos covers. Por último, en (c) está la U-matriz del agrupamiento de los dos cover y la canción no cover.

Por último, la Figura 4.4 representa lo mismo que la Figura 4.3 con la diferencia que los datos pertenecen a una prueba donde sí se realizó una correcta clasificación del cover.

## 4.3 Herramientas empleadas

Para el desarrollo de los experimentos presentados en este capítulo se utilizaron las siguientes herramientas:

- Matalab R2013b: Es un entorno de programación para el desarrollo de algoritmos, el análisis y la visualización de datos y el cálculo numérico. [44]
- UFLDL Sparse Auto-encoder: Paquete de funciones para Matlab que implementan el algoritmo Sparse Autoencoder [45], basado en [32].
- SOM Toolbox: Paquete de funciones para Matlab que implementan el algoritmo de Mapas auto-organizados, así como otros métodos para el análisis de datos. [46]

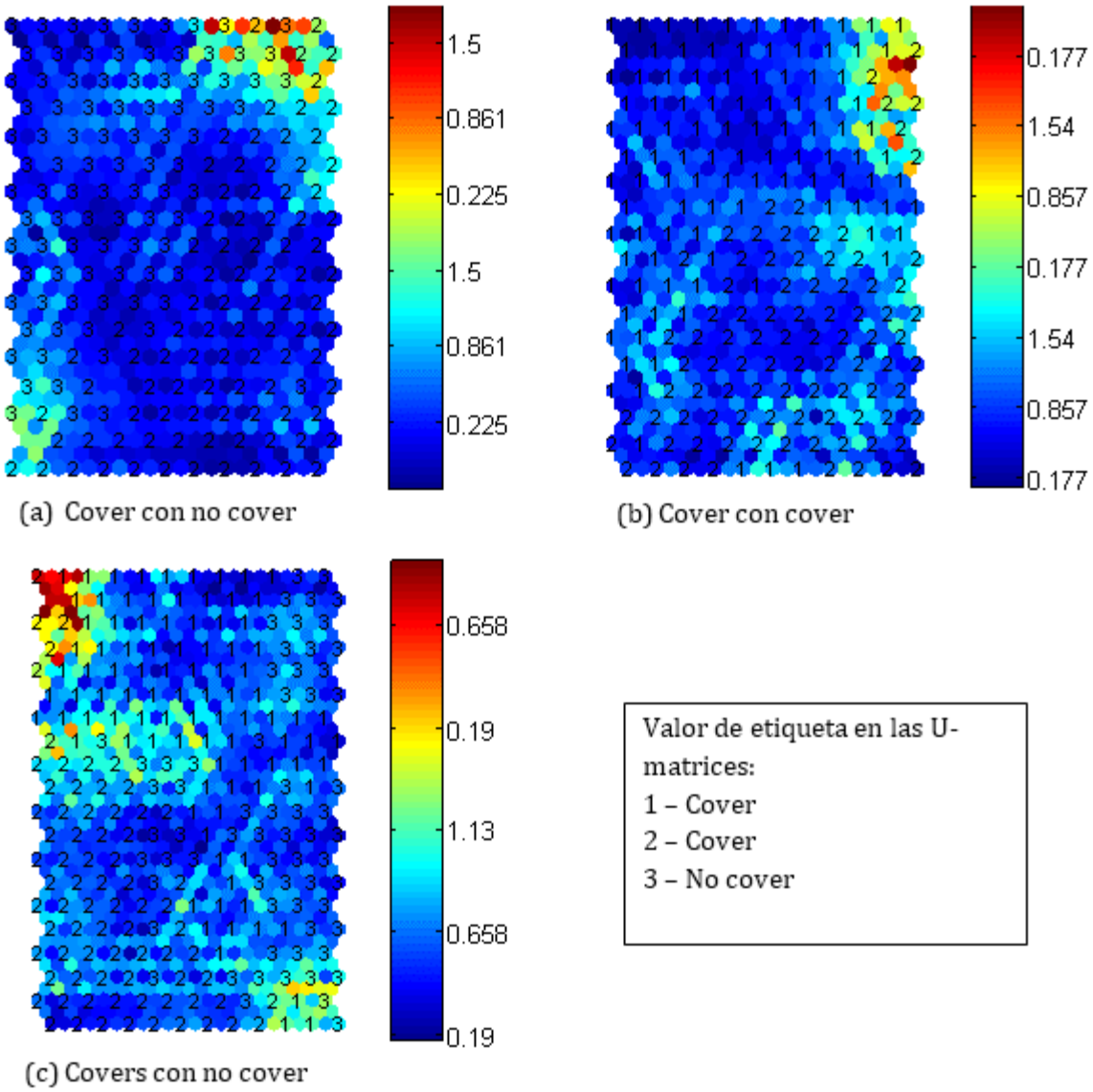



**Figura 4.3:** U-matrices de una de las pruebas que se consideró como una clasificación errónea.

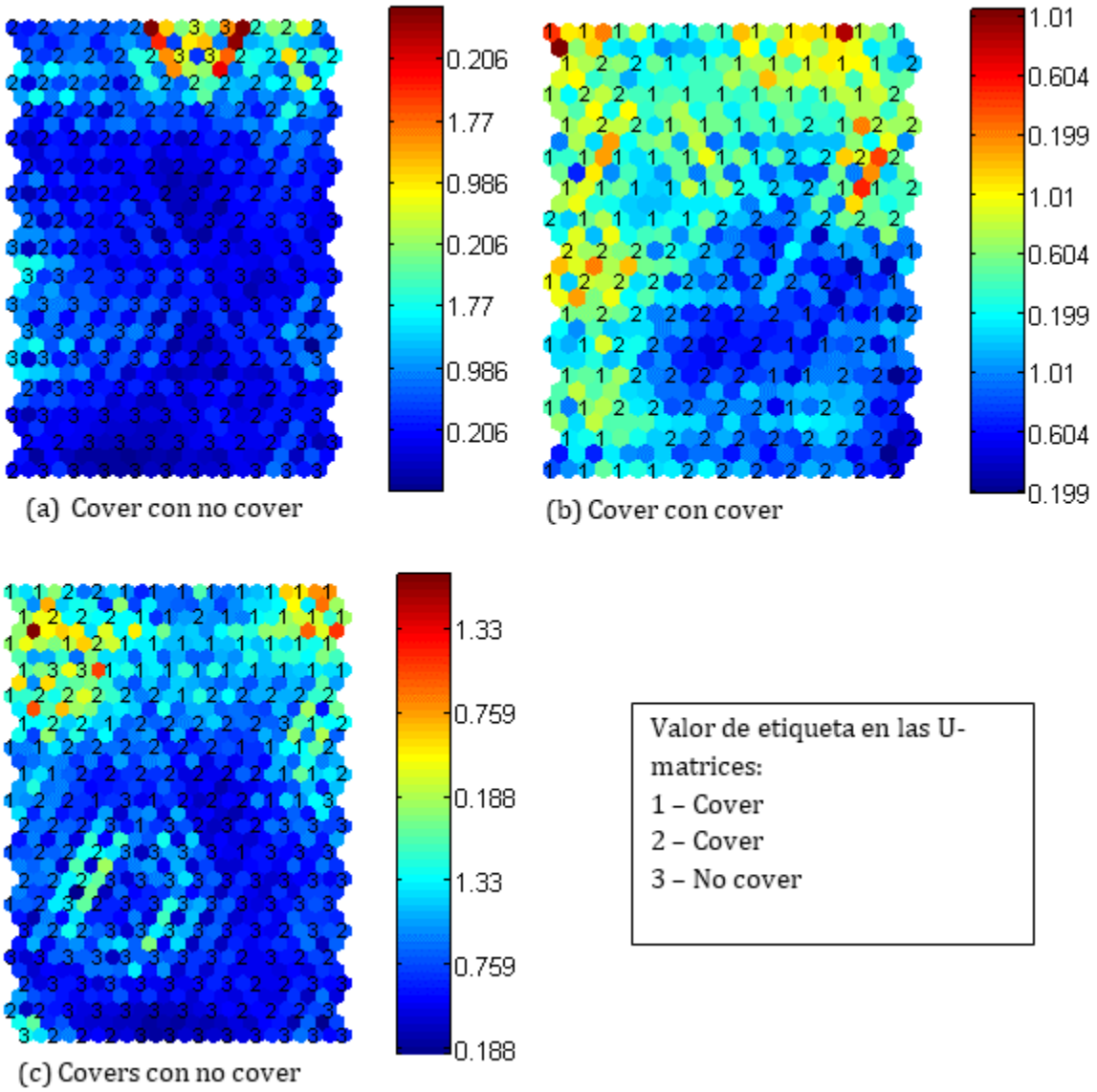



**Figura 4.4:** U-matrices de una de las pruebas que se consideró como una la clasificación correcta.

CAPÍTULO 5

# DISCUSIÓN

Esta investigación tuvo como propósito el desarrollo de la metodología de un SIVM, para alcanzar estos objetivos, primeramente comenzamos por definir algunos aspectos y requerimientos que un SIVM debe poseer, tales como:

- Base de datos. Contar con una adecuada base de datos que otorgue la información adecuada y permita realizar las pruebas necesarias para la ejecución de la metodología.
- Descriptores de la canción. Aquellas características propias de la canción que permitan distinguirla de las demás, como pudiera ser la melodía, su serie armónica o cualquier otro atributo tonal musical. También es necesario que dichos descriptores no posean un tamaño excesivamente grande, el cual interfiera de alguna manera en los costos computacionales.
- Técnica de clasificación. Aquella o aquellas que permitan una correcta separación entre los covers, como puede ser un algoritmo de aprendizaje maquinal o el uso de algún tipo de cálculo de distancia.

En esta sección se realiza un análisis de los resultados presentados en el capítulo anterior, así como también los problemas e inconvenientes surgidos en el transcurso del desarrollo de la investigación.

## 5.1 Base de datos

Para la investigación se optó por utilizar la base de datos MSD por las características que ésta posee y que consideramos como ventajas sobre otras que igualmente se encuentran disponibles en la red. Algunas de estas bases de datos se muestran en la Tabla 2.1 del capítulo de Antecedentes, donde podemos ver el número de canciones con el que cada una cuenta, y de igual manera se muestra si incluye o no el audio de la canción.

Una de estas características principales y que se distingue sobre las demás es que MSD cuenta con un millón de canciones pertenecientes a 44,745 artistas diferentes y que a su vez pertenecen a distintos géneros musicales, así mismo la base de datos contiene música tanto en inglés como en español. Al ser una base de datos rica en contenido musical da pie a que contenga distintos covers de un mismo título que sean diferentes entre sí, es decir, que varíen en cuanto ritmo, duración y arreglos.

Lo que pretendemos enfatizar con lo anterior es que esta gran cantidad y variedad de contenido musical permite probar la eficacia de los algoritmos sobre un conjunto de datos de proporciones grandes, algo muy necesario ya que hay que tomar en cuenta que las bases de datos comerciales de contenido musical que existen en la actualidad son enormes y tienen un crecimiento exponencial debido en gran parte por el avance de la tecnología. Por citar un ejemplo, actualmente Google Play dispone con más de 20 millones de canciones [36], entonces se hace totalmente imprescindible realizar las pruebas en condiciones lo más semejante posible a las condiciones reales.

Ahora, otra característica que se esperaba obtener de la base de datos era que la información que nos facilitara de las canciones fuera la necesaria para cumplir con nuestros objetivos. En este sentido, MSD proporciona únicamente metadatos extraídos de las pistas musicales, a diferencia de otras que incluyen el audio y permiten al usuario recuperar la información a partir de éste y a través de las técnicas de las cuales el usuario tuviera conocimiento.

Esto último acrecentó la duda sobre qué tan representativa podría ser la información otorgada sobre las canciones de MSD. Para solventar esta duda se recurrió

a la literatura, donde pudimos ver que se hace hincapié en que el uso de los vectores chroma es útil para la extracción de la melodía o las características propias de la canción, mismos que pudimos obtener a partir de la matriz de pitches que poseen las pistas musicales de MSD. De igual manera, se comprobó mediante experimentos qué tan representativas eran estas características acústicas, es decir la matriz de pitches y la matriz de timbres, proporcionadas por la base de datos. Estos experimentos básicamente consistieron en la clasificación de los covers utilizando distintos atributos de los datos en cada clasificación con la finalidad de tener una idea de cómo influían éstos y su nivel de importancia [ver anexo H para más detalles].

En la Figura 5.1 se muestran los resultados para uno de estos experimentos. En ella podemos ver los porcentajes de instancias clasificadas correctamente de los covers con título “Silent Night”.

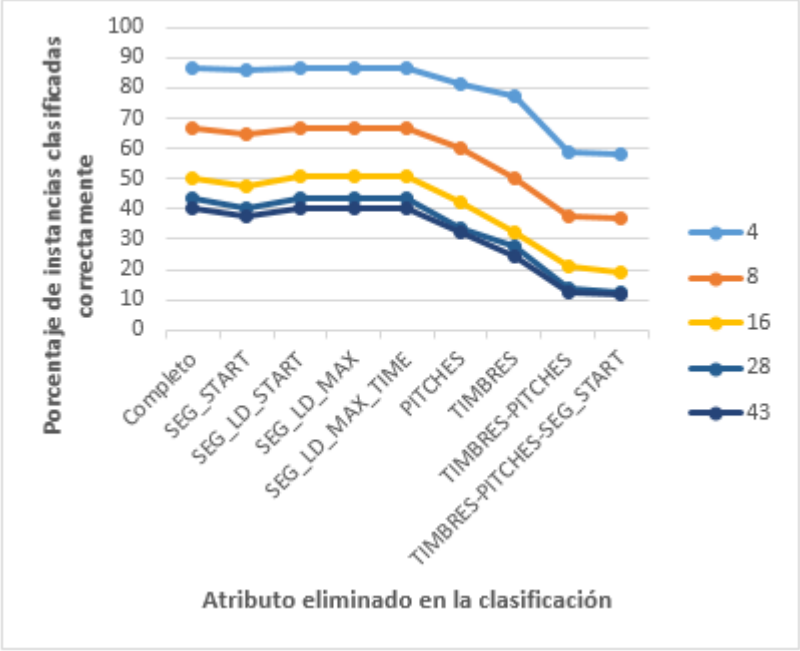



**Figura 5.1:** Porcentajes de instancias clasificadas correctamente de Silent Night de MSD utilizando Naive Bayes. Las leyendas indican el número de covers utilizados en la clasificación.

Como se distingue en la figura, mientras se utilicen los atributos pertenecientes a las matrices de pitches o de timbres dentro de la clasificación, ésta se mantiene

constante, pero al momento de eliminar alguno de estos atributos, el porcentaje de instancias clasificadas correctamente decae.

Si tomamos en cuenta los demás experimentos podemos ver que surge un comportamiento similar al mostrado en la Figura 5.1 en las clasificaciones realizadas. Esto nos ayudó a considerar la matriz de pitches y de timbres como un elemento importante a ser utilizado.

## 5.2 Análisis de Resultados

En esta sección se plantean observaciones para los resultados generados sobre los distintos experimentos de la sección 4.2, tratando de explicar lo que se pretendía obtener de éstos y las posibles causas de los resultados.

Como se recordará, primeramente se calculó la FFT y la TW para los parches de las canciones. Igualmente se utilizó el algoritmo Sparse Autoencoder sobre los parches para reducirles su tamaño y al mismo tiempo obtener un patrón característico de éstos, aprovechando que el Sparse Autoencoder puede generar una representación comprimida de los datos a los cuales se les haya aplicado el algoritmo.

Una vez generados estos datos, se decidió realizar los siguientes experimentos (ver sección 3.8):

(a) Identificación de covers mediante la comparación con un ensamble general y la delimitación por umbral.
(b) Identificación de covers mediante la comparación con un patrón general y la delimitación por umbral.
(c) Identificación de covers con técnicas de aprendizaje supervisado y no supervisado.
(d) Identificación mediante mapas auto-organizados (SOM) utilizando el timbre.

La finalidad de estos experimentos fue la siguiente: en (a) y (b) utilizar todos los covers con un mismo título para que a partir de ellos se obtuviera un ensamble/patrón característico mediante el cálculo de una medida estadística que resumiera la información de todas las canciones y con ello, cuando se analizara un nuevo cover, muy probablemente éste tuviera un alto grado de cercanía con el ensamble/patrón, lo que indicaría que en realidad es un cover. Igualmente el umbral definido nos permitiría trazar un límite para esta cercanía, es decir, hasta qué punto una canción sería considerada como cover. En el experimento (c) se pretendió realizar una comparación entre la FFT y TW de las canciones para determinar cuál transformada otorgaba una mejor representación de los datos. Para (d) se decidió utilizar los primeros cuatro coeficientes del timbre otorgados por MSD para determinar qué tanto estos permitían agrupar los covers aplicando el método SOM. Aunque la base de datos proporciona 12 coeficientes para el timbre, la descripción de ocho de estos valores es algo confusa y escasa, motivo por el cual se decidió no incluirlos en los experimentos. Mientras tanto, los cuatro coeficientes restantes representan el promedio del volumen del segmento, el brillo, la planitud del sonido y el sonido con más fuerza respectivamente de acuerdo a las notas técnicas publicadas por el proyecto Echo Nest, del cual se generó dicha información.

Para la implementación de estos experimentos se hizo uso de algunas medidas estadísticas para determinar el grado de representatividad que éstas tienen sobre la matriz de datos, la cual podía ser la matriz generada por una de las dos transformadas o por el algoritmo Sparse Autoencoder. Estas medidas estadísticas pretenden resumir la información de los datos para poder tener así un mejor conocimiento de su muestra. Las medidas empleadas fueron la media, la mediana y la desviación estándar por ser estas de rápida implementación y por consiguiente no requerir demasiado costo computacional, teniendo en cuenta que un sistema como este necesita ejecutarse en el menor tiempo posible.

Dentro de los resultados generados podemos observar lo siguiente:

### a. Identificación de covers mediante la comparación con un ensamble general y la delimitación por umbral.

En este experimento se realizaron dos tipos de pruebas. En el primero se compararon grupos de canciones elegidas al azar contra el ensamble de un grupo de covers y el sistema las tenía que identificar como cover o no cover. El segundo tipo de prueba consistió en identificar cuál era el cover dentro de las canciones comparadas.

Para las primeras pruebas se obtuvo un 6.2% de aciertos en la identificación de los no cover (ver Tabla 4.2), utilizando la resta de la media de cada matriz alineada. La Figura 5.2 nos puede mostrar lo que está sucediendo, en ella se muestra el ensamble en línea punteada roja y en azul el umbral; las líneas verdes indican donde se sitúan las canciones con respecto al ensamble y las cruces cómo están situados los covers (de los cuales se generó el ensamble) con respecto al ensamble.

Este experimento muestra que las canciones están distribuidas sobre un cierto rango de longitud no muy extenso, que hace complicado el uso de umbrales como una técnica que permita utilizarlos como límite de un grupo de canciones que comparten un mismo título. Como se percibió en la Figura 5.2, las canciones no cover comparadas con respecto al umbral se localizan por debajo de este, y se sitúan en posiciones muy cercanas a las posiciones donde se encuentran los covers a partir de los cuales se generó el ensamble.

Aunque no siempre ocurrió que todas las canciones comparadas estaban localizadas por debajo del umbral, al menos había una que si lo estaba, motivo por el cual la experimentación salió con bajos resultados en aciertos.

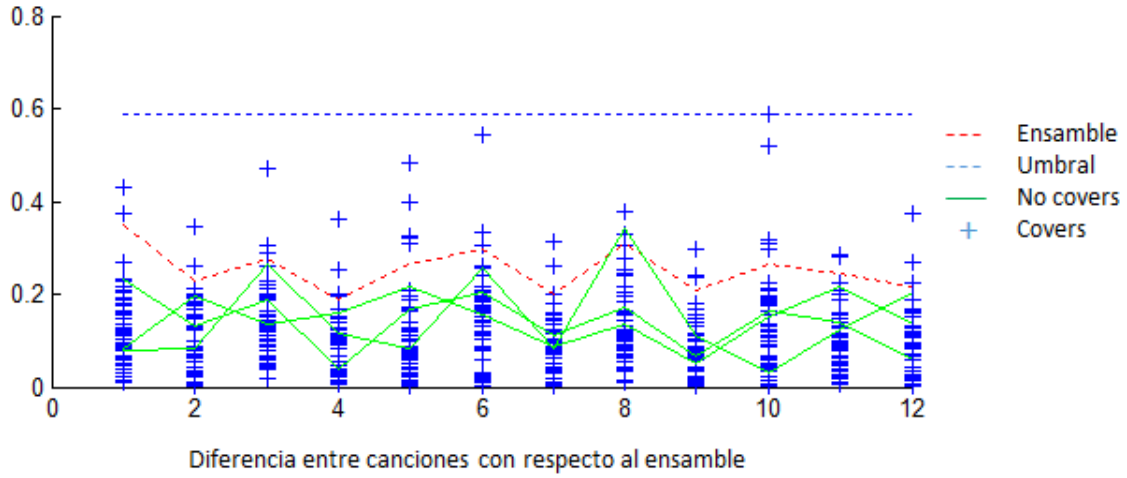



**Figura 5.2:** Comparación entre el ensamble y canciones no cover utilizando la resta. De igual manera se muestran como están distribuidos los covers a partir de los cuales se generó el ensamble.

Para las segundas pruebas el mejor resultado generó un 56.8% de aciertos en la identificación del cover (ver Tabla 4.3), utilizando la media de la matriz alineada. En la Figura 5.3 se puede apreciar lo que puede estar pasando. Al igual que en la figurara anterior, en ésta se muestra el ensamble en línea punteada roja y en azul el umbral; las líneas verdes indican donde se sitúan las canciones con respecto al ensamble y las cruces cómo están situados los covers (de los cuales se generó el ensamble) con respecto al ensamble; también se representa en línea roja el cover a identificar. En la gráfica A se ven los datos de una comparación donde se identificó erróneamente el cover; aunque el cover se encuentra por debajo del umbral existe otra canción que se acerca más al ensamble, motivo por el cual falla la identificación. La gráfica B contiene la información de una comparación donde se pudo identificar el cover; en ella se aprecia una canción no cover (en línea verde) con una distribución muy similar a la del resto de los covers y donde el cover comparado se localiza más cerca del ensamble.

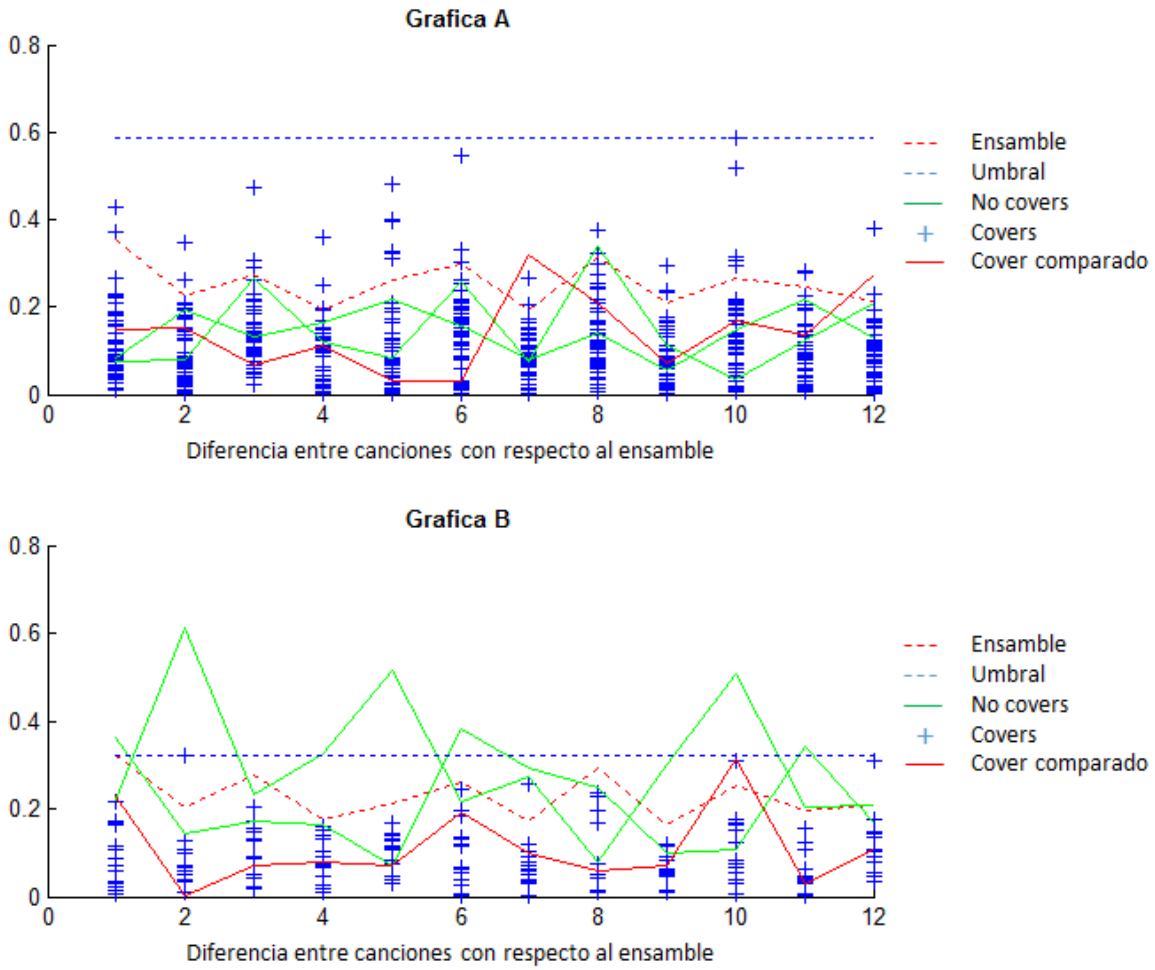



**Figura 5.3:** Comparación entre el ensamble y canciones con cover utilizando la resta. Se muestra cómo están distribuidos los covers a partir de los cuales se generó el ensamble. En la gráfica A se aprecian los datos para una identificación errónea del cover. La gráfica B muestra los datos de una identificación correcta del cover**.**

Como se puede apreciar hay una brecha significativa en los resultados entre los dos tipos de pruebas realizadas. Al no incluir un cover en la comparación la identificación falla por el hecho de que la mayoría de las canciones se sitúan dentro de un mismo rango de espacio dentro del espacio euclidiano. Pero al incluir un cover en la comparación la identificación mejoraba, esto porque el cover lograba tener más similitud con el ensamble o diferencia de las no cover. El hecho de que identificara como cover una canción que no era puede deberse a que la canción es semejante en sus arreglos a uno o varios de los covers que se utilizaron para la creación del ensamble; hay que recordar que se utilizaron datos generados a partir de la matriz de

pitches la cual contiene información sobre el dominio que tienen las notas en cada segmento de tiempo.

**b. Identificación de covers mediante la comparación con un patrón general y la delimitación por umbral.**

Básicamente esta experimentación fue similar a la del punto anterior, la diferencia radica en que el ensamble general, que en este caso le llamamos patrón general, se obtuvo a partir de los datos generados por la aplicación del Sparse Autoencoder. Este algoritmo nos permitió reducir la dimensionalidad de N parches de cada canción a un total de 30 parches por canción.

Para los resultados generados se vio una tendencia muy similar a los generados en el experimento anterior, teniendo un 2.4% de aciertos en la identificación de no covers y un 57.8% en la identificación del cover. Esto muestra también que el uso de un umbral no permite delimitar las regiones que comprenden un conjunto de covers.

Para tener una idea de lo que sucedió se muestra la Tabla 5.1, en ella se aprecian las distancias generadas (utilizando la distancia City-block) al medir el patrón general con respecto a la desviación estándar de los patrones de cada canción. Como se puede notar en la mayoría de las pruebas hubo al menos un no cover que tuviera su distancia por debajo del umbral asignado, los cual hizo que se considerara como cover. Esto igual se debe a que las distancias de los covers con los que se generó el umbral se sitúan en posiciones muy parecidas dentro del espacio euclidiano, lo cual nos dice que el uso de un umbral no ayuda en mucho en estas pruebas de identificación de los no covers.

**Tabla 5.1:** Distancias generadas de las primeras cinco pruebas para la identificación de los no covers. Utilizando un patrón general y un umbral.

| Prueba | No cover 1 | No cover 2 | No cover 3 | Umbral |
|---|---|---|---|---|
| 1 | 4.1813 | 7.4515 | 6.3101 | 7.5573 |
| 2 | 4.8746 | 5.0944 | 5.9207 | 6.6266 |
| 3 | 6.7266 | 6.0418 | 4.5278 | 6.6232 |
| 4 | 4.7624 | 5.9345 | 6.5523 | 5.2623 |
| 5 | 6.6944 | 6.8493 | 7.3138 | 7.5573 |

Por otro lado, en las pruebas de identificar el cover utilizamos la Tabla 5.2 para apreciar cómo se comportó la información, en ella se muestran las distancias calculadas para las primeras cinco pruebas, utilizando la distancia City-block. De igual manera podemos ver que en la mayoría de los casos las distancias se encuentran por debajo del umbral asignado, con la diferencia de que en gran parte de las pruebas el cover posee la menor distancia al patrón general. En este caso, sólo se muestra en la tabla una identificación correcta del cover (prueba 4), pero se decidió mostrar casos en los que no se acertó en la identificación con la finalidad de generar una idea de cómo se comportaron los datos, hay que recordar que para estas pruebas se logró un 57.8% en la identificación de los covers.

Con esta tablas podemos apreciar qué tanto las canciones cover como las no cover se encuentran en posiciones muy similares, lo cual conlleva que al no utilizar un cover dentro de la identificación no se logren resultados muy alentadores.

**Tabla 5.2:** Distancia generada en las primeras cinco pruebas para la identificación del cover. Utilizando un patrón general y un umbral.

| Prueba | No cover 1 | No cover 2 | Cover | Umbral |
|---|---|---|---|---|
| 1 | 2.9209 | 6.5266 | 3.8275 | 7.2782 |
| 2 | 3.2616 | 2.9309 | 4.1395 | 4.6327 |
| 3 | 7.3860 | 4.2950 | 5.6704 | 9.4974 |
| 4 | 5.1097 | 5.7271 | 3.3187 | 5.8260 |
| 5 | 4.6570 | 9.5571 | 6.3386 | 7.5832 |

Otra cosa interesante de este experimento fue notar que el uso del Sparse Autoencoder en combinación con la entropía de alguna manera logró obtener un patrón que conservaba la información de cada canción reduciendo la dimensionalidad de los datos. Si vemos la Tabla 5.3 podemos ver un conjunto de canciones en las cuales se muestra el identificador de la canción (track_id) y el número de parches que se generaron a partir de ella. Como se puede apreciar el tamaño de parches es una variable que depende en gran medida de la longitud de la pieza musical (el tiempo que dura la canción), al aplicar el Sparse Autoencoder se logra reducir la cantidad de estos a un tamaño de 30 patrones, los cuales logran representar al conjunto de los N parches.

Para determinar la cantidad adecuada de patrones que habríamos de generar para cada canción se realizaron diversas pruebas, en ellas se experimentó con el número de patrones que deseábamos obtener, con rangos que iban desde 20 a 120, obteniendo los mejores resultados cuando se calcularon 30 patrones.

Por otro lado a estos patrones se les calculó su entropía, que es una medida estadística de aleatoriedad que se puede utilizar para caracterizar la textura de una imagen de entrada. En este experimento se utilizó para obtener información de los patrones que indicara que tan correlacionados estaban estos datos, suponiendo que los covers con un mismo título tendrían gran similitud en la forma en que sus datos se correlacionaban.

**Tabla 5.3:** Ejemplo de la cantidad de parches que se generaron para algunas canciones.

| Track_id | # de parches |
|---|---|
| TRBAJIB128F1477E11 | 762 |
| TRGFBXM128F42BA6B4 | 247 |
| TRGYPVL128F149888E | 443 |
| TRHIRDK128F425BAD2 | 282 |
| TRUZYUD128F42929F9 | 199 |
| TRHIIZW12903CED302 | 164 |

### c. Identificación de covers con técnicas de aprendizaje supervisado y no supervisado.

En este experimento nos centramos en el uso de la transformada Wavelet y la implementación del Sparse Autoencoder. Si bien se recuerda, esta investigación está basada en el trabajo realizado por Thierry Bertin-Mahieux, donde hacen uso del cálculo de la FFT-2D para obtener los componentes frecuenciales que ayuden a describir la canción. En este trabajo utilizamos la transformada Wavelet con la finalidad de determinar si su implementación genera mejores resultados.

Como se pudo apreciar en la sección 3.8 los resultados obtenidos en la identificación del cover cuando se aplicó la TW quedaban por debajo de los resultados generados por la FFT-2D, esto pudiera deberse a la manera en que la transformada Wavelet procesa la información sobre la cual está trabajando. El hecho de calcular solo las frecuencias podría ser más ventajoso dado que se trata de encontrar la similitud entre canciones parecidas (covers). En este caso la transformada de Fourier descompone cada parche y nos dice cómo está constituido donde se resaltan aquellos datos que más dominio tuvieron en ese segmento.

Por otro lado se tenía previsto, dadas las características de la TW, que poseer información de tiempo-frecuencia resultaría ser más ventajoso; dados los resultados podemos apreciar que el hecho de poseer esta otra variable no resultó de gran ayuda. Otro punto que se puede poner a discusión, a partir de estos resultados, es que la información que se presenta en MSD de los pitches no es lo suficientemente representativa y en este caso sería mejor implementar la TW sobre las muestras de audio.

**d. Identificación mediante mapas auto-organizados (SOM) utilizando el timbre.**

En esta experimentación se decidió utilizar los SOM sobre cuatro coeficientes de la matriz de timbre, para determinar el grado de agrupamiento que estos permiten. Lo que se intentó aquí fue aprovechar las características de un SOM de mapear los datos similares en posiciones continuas, es decir posicionar los datos más parecidos en los lugares más próximos entre sí, dejando aquellos que no se parecen más separados. Esto permitió apreciar el grado de agrupamiento que las características del timbre dadas en la base de datos permitían.

Si vemos, por ejemplo, el resultado generado de una clasificación donde se identificó al cover como no cover (Figura 4.3 del capítulo de Resultados). Podemos apreciar que en cierta medida los datos de las tres canciones se alcanzan a separar entre sí para generar sus propios grupos (inciso [c] de la misma figura); lo importante aquí seria identificar hasta qué punto un grupo pertenece a otro o viceversa, hasta qué punto un grupo estaría dejando de pertenecer a otro, algo un poco complicado de fijar para estos casos. Por otro lado, si se ven los resultados generados para una clasificación donde se identificó al cover (Figura 4.4 del capítulo de Resultados), se puede ver por ejemplo, como los datos pertenecientes a las canciones que son covers (etiqueta 1 y 2 del inciso [c] sobre la misma figura) quedan "revueltos" entre si dejando al no cover en un grupo diferente; lo que permitió para este caso poder idéntica al cover de una manera más simple.

Lo que se intenta resaltar en cuanto a la implementación de los SOM es que obtener una medida de certeza en los resultados no es tan claro ni directo, por el hecho de que esta red neuronal lo que hace es agrupar los datos por la similitud de estos entre sí, dejando los más parecidos más cerca y los de menor similitud más lejos. La parte que debe atenderse es la definición de los límites donde los datos dejan de parecerse entre sí.

De igual manera podría ser un algoritmo ventajoso de implementar en conjunto con otro tipo técnica de inteligencia artificial, ya que este podría proporcionar los datos que sean más similares para posteriormente trabajar sobre estos, reduciendo así la dimensionalidad que la base de datos pudiera poseer.

## 5.3 Análisis comparativo

Con la finalidad de comprobar la eficacia de nuestra metodología con respecto a la de otros trabajos, se optó por reproducir la experimentación realizada por Thierry Bertin-Mahieux, and Daniel P. W. Ellis [11] en la cual se realizaron 500 consultas binarias que consistían cada una en: dada una canción A que servía como consulta y dos canciones B y C, determinar si B o C era el cover de A. Lo que les genero un 82.0% de covers identificados correctamente (sin aplicar PCA sobre los datos) y un 82.2% de exactitud utilizando PCA con 50 componentes principales.

Para nuestro caso, a modo de poder realizar una comparación, se utilizó la misma experimentación descrita arriba cambiando únicamente la lista de 500 consultas, por otra que nosotros generamos, obteniendo un 75.0% de covers identificados sin aplicar PCA y un 78.8% de exactitud aplicando PCA con 50 componentes. En la implementación de nuestra metodología se realizaron las mismas 500 consultas binarias pero utilizando los algoritmos k-nn y Sparse Autoencoder, por ser aquellos que mejores resultados nos generaron, esto nos dio como resultado un 73.8% de covers identificados correctamente.

CAPÍTULO 6

# CONCLUSIONES Y TRABAJO FUTURO

En este trabajo nos enfocamos en la creación de un sistema de identificación de covers, basándonos en técnicas de aprendizaje maquinal, procesamiento de señales y estadística de segundo orden.

Para el desarrollo de este sistema se realizaron cuatro tipos de experimentaciones, de las cuales podemos decir lo siguiente:

a. Identificación de covers mediante la comparación con un ensamble general y la delimitación por umbral. La creación de un límite que agrupara todas aquellas canciones cover, que generaron el ensamble, no impidió aislar el conjunto de covers del resto de las canciones, ya que estas en su gran mayoría quedaban por debajo del umbral establecido, por lo que el ensamble general fue lo que nos permitió obtener el porcentaje de aciertos alcanzados.
b. Identificación de covers mediante la comparación con un patrón general y la delimitación por umbral. De igual manera que en el punto anterior, la implementación de un umbral no impido aislar el conjunto de covers del resto de canciones comparadas, siendo el patrón general el que dio oportunidad de identificar los covers en un buen número de casos.

    En cuanto a la implementación del método Sparse Autoencoder este permitió la reducción de los datos aunado al hecho de que se pudo obtener un patrón que representara cada canción a la cual se le aplicara este método.

c. Identificación de covers con técnicas de aprendizaje supervisado y no supervisado. Se utilizaron los algoritmos K-nn y MLP para la identificación de covers, de los cuales K-nn permitió una mejor clasificación de éstos. Se utilizó la transformada de Fourier y transformada Wavelet, así como la implementación del Sparse Autoencoder para la obtención de aquellas características que permitieran una mejor representación de los datos utilizados, lo que nos llevó a inferir que en este caso el uso de la transformada de Fourier otorgaba datos más relevantes que la transformada Wavelet.
d. Identificación mediante mapas auto-organizados (SOM) utilizando el timbre. La utilización de los timbres no proporcionó una mejora sobre los resultados generados a partir del uso de los pitches. Por otro lado, la creación de los mapas de agrupamiento reveló algunas dificultades para la identificación del grupo, pero tambiém permitió identificar claves que pueden ocuparse mediante su combinación con otras técnicas de aprendizaje maquinal para mejorar los resultados.

Estos experimentos ayudaron a identificar las características que describían los covers de una mejor manera, con lo que se pudo apreciar que la TTF-2D aplicada sobre los parches o sobre los patrones describe mejor una canción. Por otro lado el uso de un K-nn mejora los resultados en comparación de otras técnicas. En la Figura 6.1 se puede apreciar la arquitectura de este sistema final. El mejor resultado se obtuvo con K-nn (64%) el cual se encuentra cerca del rango reportado [11] (66-82%); cabe aclarar que las pruebas no son las mismas.

En cuanto a trabajo futuro, se tiene las siguientes recomendaciones:

- Trabajar con señales crudas (audios) utilizando otras bases de datos que así las proporcionen.
- De igual manera, experimentar con la transformada de Fourier y la transformada Wavelet sobre el audio directo.
- Incorporar sistemas de clasificación más sofisticados como las Máquinas de Soporte Vectorial o redes de aprendizaje profundo.

- Intentar con sistemas de inferencia como los arboles de decisión. Todo esto con la finalidad de hacer sobresalir las características del patrón base de cada ensamble de canciones.

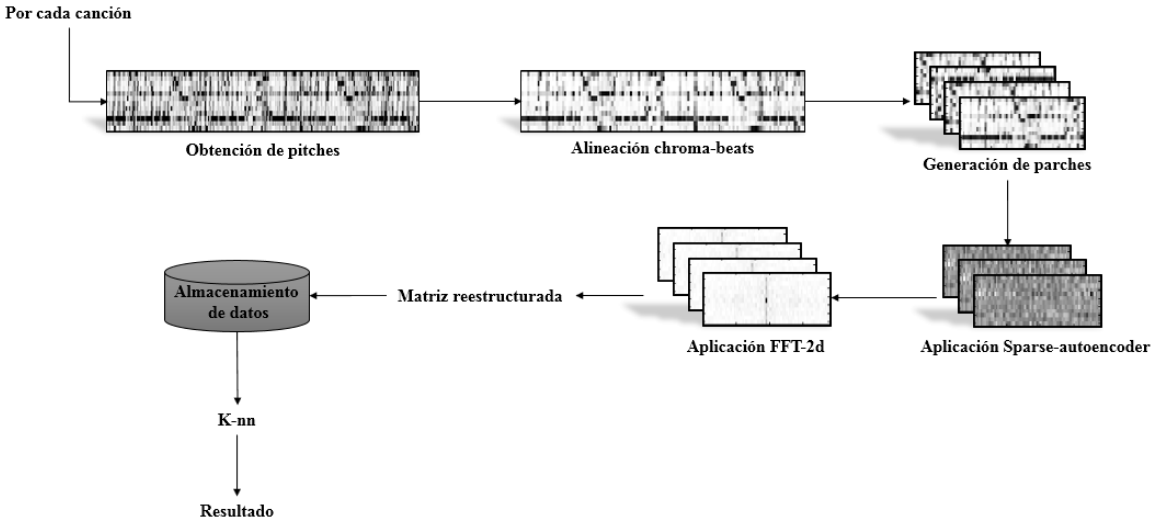



**Figura 6.1:** Arquitectura de sistema de identificación de covers.

ANEXO A

# ALGORITMOS DE APRENDIZAJE MAQUINAL

Aprendizaje maquinal es el resultado de la intersección entre las Ciencias de la Computación y la Estadística [25]. Se puede definir como aquella disciplina científica que se concentra en crear y estudiar algoritmos que pueden aprender automáticamente a partir de datos [24].

Los problemas propios del aprendizaje maquinal clasifican en tres categorías: supervisado, no supervisado y aprendizaje por refuerzo. Todo esto dependiendo del tipo de retroalimentación que se tenga disponible [26].

Algunos algoritmos de aprendizaje maquinal son descritos a continuación.

- **Naive-Bayes**

**Teorema de Bayes.** Sea X el patrón cuya clase es desconocida. Sea $H_i$ alguna hipótesis que indica la clase a la que pertenece X. Por ejemplo, $H_i$ es la hipótesis de que un patrón corresponde a la clase $C_i$. Suponiendo que la probabilidad a priori de Hi, dada por $P(H_i)$, es conocida. Para clasificar X, se necesita determinar $P(H_i \mid X)$ que es la probabilidad que sostiene la hipótesis $H_i$, dado el patrón observado X.

$P(H_i \mid X)$ es la probabilidad a posteriori de $H_i$ condicionado en $X$. En contraste, $P(H_i)$ es la probabilidad a priori de $H_i$. Es la probabilidad de la hipótesis independientemente del valor de $X$. La probabilidad posterior, $P(H_i \mid X)$ se basa en X y otra información (tal como el conocimiento previo), mientras que la probabilidad a priori, $P(H_i)$ es obtenida antes de observar $X$.

Del mismo modo, $P(X \mid H_i)$ es la probabilidad de $X$ condicionada en $H_i$. Nótese que $P(X)$ se define como una combinación ponderada de $P(X \mid H_i)$ y es:

$$P(\mathrm{X}) = \sum_i P(\mathrm{X} \mid \mathrm{H}_i) P(\mathrm{H}_i)$$

El Teorema de Bayes es útil, ya que proporciona una forma de calcular la probabilidad a posteriori, $P(H_i \mid X)$, por $P(H_i)$, $P(X)$ y $P(X \mid H_i)$. Indica.

$$P(\mathrm{H}_i \mid \mathrm{X}) = \frac{P(\mathrm{X} \mid \mathrm{H}_i) P(\mathrm{H}_i)}{P(\mathrm{X})}$$

Como es descrito en [27], el clasificador Naive-Bayes se base en el supuesto de que la información sobre las clases es conocida en forma de probabilidades a priori y la distribución de los patrones de la clase. Utiliza la probabilidad a posteriori para asignar a la clase un patrón de prueba. A un patrón se le asigna aquella clase que tenga la máxima probabilidad posteriori. El clasificador emplea el teorema de Bayes para convertir la probabilidad a priori en probabilidad a posteriori basado en el patrón a ser clasificado, utilizando los valores de probabilidad.

- **Perceptrón Multicapa (MLP, Multilayer Perceptron)**

Un ANN (Red neuronal artificial, por sus siglas en inglés) es un algoritmo de aprendizaje estadístico que se basa en la analogía del comportamiento de las neuronas del cerebro humano. La unidad de una ANN es un procesador denominado neurona, el cual calcula una suma ponderada de sus entradas. Posteriormente aplica una función de activación para obtener una señal que será transmitida a la próxima neurona. Estas neuronas artificiales se agrupan en capas o niveles y poseen un alto grado de conectividad entre ellas, conectividad que es ponderada por los pesos. A través de un algoritmo de aprendizaje supervisado o no supervisado, las ANN ajustan su arquitectura y parámetros que puedan minimizar alguna función de error que indique el grado de ajuste a los datos y la capacidad de generalización de las ANN. [28]

De esta manera, un MLP es un modelo de ANN. La arquitectura de un MLP está compuesta por una capa de entrada y una capa de salida, así como ninguna, uno o varias capas ocultas entre las capas de entrada y salida (ver Figura A.1). Las conexiones existen solo entre las neuronas de las capas consecutivas. [29]

Donde la función de entrada de cada neurona oculta y cada neurona de salida es la suma ponderada (ponderado con los pesos de conexión) de las entradas, es decir:

$$\forall u \in U_{\text{hidden}} \cup U_{\text{out}} : \quad f_{\text{net}}^{(u)}(\mathbf{w}_u, \mathbf{in}_u) = \mathbf{w}_u \mathbf{in}_u = \sum_{v \in \text{pred}(u)} w_{uv}\, \text{out}_v \, .$$

La función de activación de cada neurona es llamada como función sigmoidea y es definida por la siguiente formula:

$$P(t) = \frac{1}{1 + e^{-t}}$$

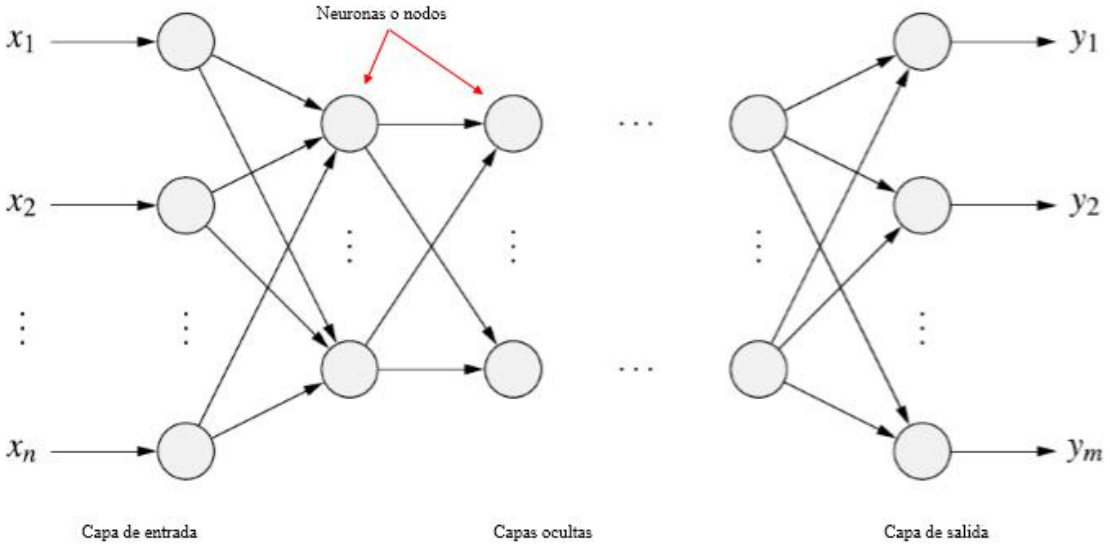



**Figura A.1:** Estructura general del perceptrón multicapa.

- **Máquinas de soporte vectorial (SVM, Support Vector Machine)**

Una SVM es un clasificador binario. Abstrae una frontera de decisión en el espacio multidimensional utilizando un subconjunto del conjunto de entrenamiento de vectores apropiados; los elementos de este sub-conjunto son los vectores de soporte. Geométricamente, los vectores de soporte son los patrones de entrenamiento que están más cerca de la frontera de decisión. [27]

Un SVM implementa conceptualmente la siguiente idea [30]: dado un conjunto de entrenamiento, los vectores de entrada x son mapeados a un espacio de mayor dimensión por la función Φ. En este espacio se construye una superficie lineal de decisión con el máximo margen. Para esto un SVM requiere la solución del siguiente problema de optimización:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i \qquad \text{sujeto a} \qquad \begin{array}{l} y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \\ \xi_i \geq 0. \end{array}$$

Donde $K(x_i, x_j) \equiv \Phi(x_i)^t \Phi(x_j)$ es la función núcleo que permite determinar en qué espacio de características se colocaran los datos de entrada. Otras funciones núcleo que se pueden aplicar son las siguientes:

| | |
|---|---|
| Lineal | $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T\mathbf{x}_j.$ |
| Polinominal | $K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma\mathbf{x}_i{}^T\mathbf{x}_j + r)^d,\ \gamma > 0.$ |
| Funciones de base radial | $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2),\ \gamma > 0.$ |
| Sigmoid | $\tanh(\gamma\mathbf{x}_i{}^T\mathbf{x}_j + r).$ |

- **K-means**

De acuerdo con [31], K-means es un algoritmo de agrupamiento utilizado comúnmente por su bajo cómputo y complejidad de almacenamiento, es considera como "barato y bueno". La manera en que opera es la siguiente: [27]

1. Se selecciona k por los n patrones dados como los centros de grupo iniciales. Asignando cada uno de los n-k patrones restantes a uno de los k grupos; un patrón se asigna a su centro/grupo más cercano.
2. Se calculan los centros de grupo basado en la asignación actual de los patrones.
3. Se asigna cada uno de los n patrones a su centro/grupo más cercano.

Detener si no hay cambio en la asignación de patrones a los grupos durante dos iteraciones sucesivas, de otros modo ir al paso 2.

ANEXO B

# BASE DE DATOS MILLION SONG DATASET (MSD)

Como se menciona en [37], MSD es una colección de características y metadatos de audio, de un millón de pistas de música contemporánea popular. MSD es un proyecto elaborado en colaboración entre The Echo Nest y LabROSA, con propósitos como:

- Fomentar la investigación de algoritmos que escalen a tamaños comerciales.
- Proveer una base de datos de referencia para evaluar las investigaciones.
- Como alternativa a la creación de una base de datos grande utilizando The Echo Nest´s API.
- Para auxiliar a los investigadores en su inicio en el campo de MIR (Recuperación de la Información Musical).

Esta base de datos fue creada a partir del uso de la The Echo Nest API[17], y es almacenada en archivos de formato HDF5 (Hierarchical Data Format), que son un tipo de archivos y librerías diseñadas para almacenar y organizar grandes cantidades de datos. Donde cada archivo HDF5 representa una interpretación musical con toda la información relacionada con dicha pista. En total la base de datos tiene un peso de 280 GB aproximadamente, y se encuentra dividida en 26 partes para facilitar su descarga, disponibles de manera gratuita.

[17] Más información de The Echo Nest API se encuentra en http://developer.echonest.com/

Se encuentra constituida por 55 campos que pueden ser características contextuales (nombre del artista, título de la pista, año de creación de la pista, artistas similares,…) o características acústicas (pitches, timbres, beats, volumen,…), en la Tabla B.1 se puede apreciar los campos que constituyen[18] cada archivo HDF5 de la base de datos.

**Tabla B.1:** Lista de los 55 campos que conforman cada archivo HDF5 en MSD. [8]

| | |
|---|---|
| analysis_sample_rate | artist_7digitalid |
| artist_familiarity | artist_hotttnesss |
| artist_id | artist_latitude |
| artist_location | artist_longitude |
| artist_mbid | artist_mbtags |
| artist_mbtags_count | artist_name |
| artist_playmeid | artist_terms |
| artist_terms_freq | artist_terms_weight |
| audio_md5 | bars_confidence |
| bars_start | beats_confidence |
| beats_start | danceability |
| duration | end_of_fade_in |
| energy | key |
| key_confidence | loudness |
| mode | mode_confidence |
| num_songs | release |
| release_7digitalid | sections_confidence |
| sections_start | segments_confidence |
| segments_loudness_max | segments_loudness_max_time |
| segments_loudness_start | segments_pitches |
| segments_start | segments_timbre |
| similar_artists | song_hotttnesss |
| song_id | start_of_fade_out |
| tatums_confidence | tatums_start |
| tempo | time_signature |
| time_signature_confidence | title |
| track_7digitalid | track_id |
| year | |

[18] La descripción de cada uno de estos campos se encuentra disponible para su consulta en http://labrosa.ee.columbia.edu/millionsong/pages/example-track-description y http://labrosa.ee.columbia.edu/millionsong/faq

Algunas investigaciones elaboradas a partir de la base de datos son: identificación del artista, etiquetado automático de la pista, sistemas de recomendación musical e identificación de covers entre otros.

ANEXO C

# SPARSE AUTOENCODER

El Autoencoder es una red neuronal de aprendizaje no supervisado que aplica retropropagacion y trata de establecer sus valores de salida lo más semejante posible a sus valores de entrada. El algoritmo intenta aprender una función $h_{W,b}(x) = x$. Es decir, está tratando de aprender una aproximación a la función identidad, tal que su salida $\hat{x}$ sea similar a $x$. En la Figura C.1 se puede apreciar la estructura de un Autoencoder. [32]

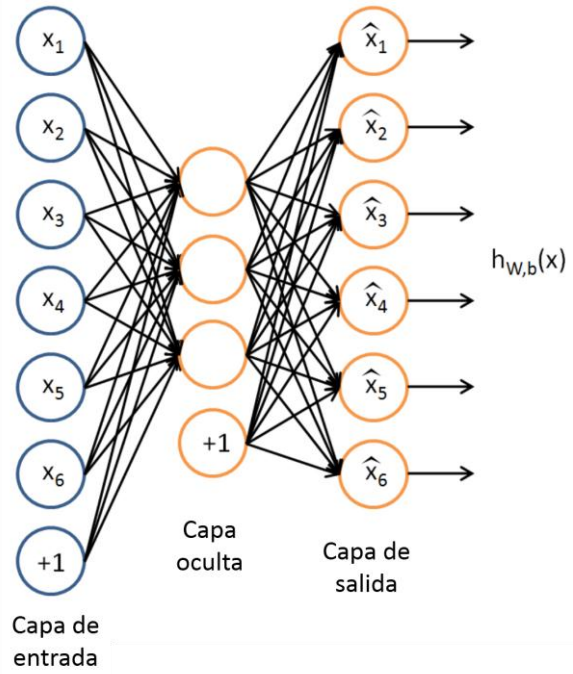



**Figura C.1:** Arquitectura del Autoencoder.

Por ejemplo, si tenemos 100 valores en la capa de entrada y 50 neuronas en la capa oculta, el algoritmo al tratar de reproducir esos 100 valores en su capa de salida se ve obligado a encontrar una representación comprimida (de 50 unidades) que le permita reconstruir sus datos de entrada. Esto último es lo que hace interesante a un Autoencoder, cuando le aplicamos restricciones como limitar el número de neuronas en su capa oculta a menudo termina aprendiendo una representación de pocas dimensiones, de manera muy similar que un PCA.

El ejemplo anterior se basó en colocar un número de neuronas ocultas que no fuera tan grande, pero que sucede si estas neuronas ocultas son de la misma cantidad o inclusive más que las que se encuentran en la capa de entrada. Si implementamos otras restricciones, como la de sparsity (dispersión), el autoencoder aún puede otorgarnos una estructura interesante de los datos.

Informalmente, podemos pensar en una neurona activada si su valor de salida es próximo a 1 o inactiva si su valor de salida es próximo a 0. Nos gustaría restringir a las neuronas a estar inactivas la mayor parte del tiempo. Todo esto asumiendo una función de activación sigmoide.

Si tómanos en consideración que $a_j^{(2)}$ denota la activación de una neurona escondida *j* en la autoencoder, esta notación no hace explicita cual fue la entrada x que condujo a la activación. En consecuencia, mejor se escribe $a_j^{(2)}(x)$ para indicar la activación de esta neurona escondida cuando es dada una entrada específica $x$ a la red. Por otra parte, dejar a

$$\hat{\rho} = \frac{1}{m}\sum_{i=1}^{m}\left[a_j^{(2)}\left(x^{(i)}\right)\right]$$

ser la media de activación de la neurona escondida *j*. Se proferiría cumplir la restricción $\hat{\rho} = \rho$, donde $\rho$ es un parámetro de dispersión con un valor cercano a cero (por ejemplo $\rho = 0.05$). Es decir, la media de activación de cada neurona escondida *j* debe ser cercana a 0.05.

Para lograr esto se añade un término de penalización extra, que penaliza a $\hat{\rho}_j$ si se desvía significativamente de $\rho$. Eligiendo como penalización la siguiente:

$$\sum_{j=1}^{s_2} \rho \log\frac{\rho}{\hat{\rho}_j} + (1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_j}$$

Donde $s_2$es el número total de neuronas en la capa oculta. Este término de penalización está basado en el concepto de divergencia de Kullback-Leibler (KL) y también puede escribirse como:

$$\sum_{j=1}^{s_2} KL\left(\rho||\hat{\rho}_j\right)$$

La divergencia de KL es una función estándar utilizada para medir que tan diferentes son dos distribuciones. Siendo ahora la función de costo total:

$$J_{sparse}(W,b) = J(W,b) + \beta\sum_{j=1}^{s_2} KL\left(\rho||\hat{\rho}_j\right)$$

donde $J(W,b)$ es la función de costo total previa, y $\beta$ controla el peso del termino de penalización de dispersión. La activación de una neurona oculta depende de los parámetros $W, b$.

ANEXO D

# TRANSFORMADA DE FOURIER

La esencia de la Transformada de Fourier es descomponer o separar una señal en una serie de sinusoides de diferentes frecuencias, mismas que corresponderían a la suma de la señal original. [39]

La transformada de Fourier identifica o distingue las diferentes sinusoides de frecuencia que combinan una señal arbitraria. Matemáticamente se define como:

$$S(f) = \int_{-\infty}^{\infty} s(t)e^{-j2\pi ft}dt$$

Donde:

- $s(t)$ es la señal a ser descompuesta (señal en el dominio del tiempo).
- $S(f)$ es la transformada de Fourier, señal transformada al dominio de la frecuencia.
- $j = \sqrt{-1}$

En la Figura D.1 se puede apreciar una interpretación grafica de la transformada de Fourier. En el inciso (a) se muestra una simple señal, en el inciso (b) están las dos sinusoides (que son la transformada de Fourier) que se suman para producir la señal. Por ultimo en (c) la representación gráfica de la Transformada de Fourier es un diagrama que muestra la amplitud y la frecuencia de cada una de las sinusoides.

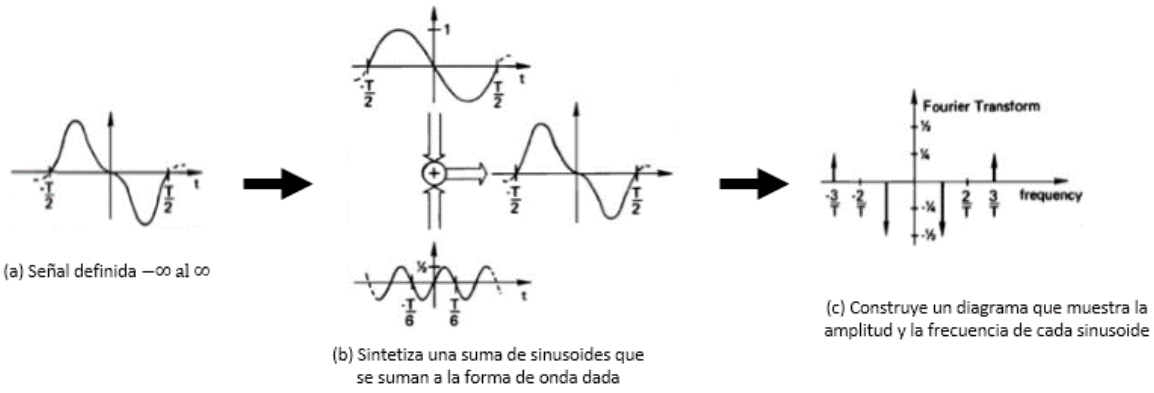



**Figura D.1:** Interpretación de la Transformada de Fourier.

ANEXO E

# TRANSFORMADA WAVELET

La transformada Wavelet no solo es local en el tiempo sino también en la frecuencia. Lo que permite generar datos de una señal en componente de escala tiempo-frecuencia. Las funciones wavelets, funciones básicas de la transformada Wavelet, son generadas a partir de una función wavelet básica llamada "Wavelet madre", mediante translaciones y dilataciones en el tiempo. [40]

De manera muy general la transformada Wavelet de una función $f(t) \in Z$ es la descomposición de $f(t)$ en un conjunto de funciones $h_{s,\tau}(t)$ que forman una base y son llamadas "Wavelets". Esta transformada se define como:

$$W_{s,\tau} = \int f(t) h^*_{s,\tau}(t) dt$$

Donde * indica el complejo conjugado. Las Wavelets son generadas por la translación y cambio de escala de una misma función Wavelet básica, conocida como Wavelet madre y que es definida como:

$$h_{s,\tau}(t) = \frac{1}{\sqrt{s}} h\left(\frac{t-\tau}{s}\right)$$

Donde s es el factor de escala y $\tau$ es el factor de translación. Las wavelets generadas de la misma función wavelet madre tienen diferente escala s y ubicación $\tau$, pero todas tienen la misma forma.

A continuación se da una introducción a las wavelet empleadas en esta investigación: [41]

**Wavelet Sinc.** Esta wavelet es obtenida a partir de la wavelet B-spline que está definida de la siguiente manera:

$$\psi(x) = \sqrt{\frac{f_b}{m}}\left(\frac{\sin\left(\frac{\pi f_b x}{m}\right)}{\frac{\pi f_b x}{m}}\right)^m e^{2i\pi f_c x}$$

y depende de tres parámetros: el orden de la wavelet dado por $m > 1$, $f_b$ es el ancho de la ventana y $f_c$ es la frecuencia central de la wavelet.

Finalmente si colocamos $m = 1$, obtenemos una familia de wavelets con dos parámetros, denominada wavelet Shannon o wavelet Sinc.

**Wavelet Daubechies.** Son una familia de wavelets que hacen posible el uso de wavelets ortogonales con soporte compacto y regularidad arbitraria. Esta familia contiene la wavelet Haar, que es la wavelet más antigua y es definida como:

$$\psi(x) = 1\ if\ x\ \in\ [0,0.5[, \psi(x) = -1\ if\ x\ \in [0.5,1[\ and\ 0\ if\ not$$

La función de escala asociada es:

$$\varphi(x) = 1\ if\ x\ \in [0,1]\ and\ 0\ if\ not$$

Con excepción de la wavelet Haar, las wavelets de esta familia no tienen un expresión explicita.

**Wavelet Meyer.** Es una de las primeras wavelets, fue construida por Meyer a mediados de los 80's. Las $\psi$ y $\varphi$ funciones son definidas en el dominio de la frecuencia, utilizando una función auxiliar $\nu$:

$$\hat{\psi}(\omega) = \begin{cases} (2\pi)^{-1/2} e^{i\omega/2} \sin\left(\frac{\pi}{2}\nu\left(\frac{3}{2\pi}|\omega| - 1\right)\right) & if\ \frac{2\pi}{3} \leq |\omega| \leq \frac{4\pi}{3} \\ (2\pi)^{-1/2} e^{i\omega/2} \cos\left(\frac{\pi}{2}\nu\left(\frac{3}{4\pi}|\omega| - 1\right)\right) & if\ \frac{4\pi}{3} \leq |\omega| \leq \frac{8\pi}{3} \\ 0 & if\ |\omega| \notin \left[\frac{2\pi}{3}, \frac{8\pi}{3}\right] \end{cases}$$

donde $\nu(a) = a^4(35 - 84a + 70a^2 - 20a^3)$ con $a \in [0,1]$.

$$\hat{\varphi}(\omega) = \begin{cases} (2\pi)^{-1/2} & if\ |\omega| \leq \frac{2\pi}{3} \\ (2\pi)^{-1/2} \cos\left(\frac{\pi}{2}\nu\left(\frac{3}{2\pi}|\omega| - 1\right)\right) & if\ \frac{2\pi}{3} \leq |\omega| \leq \frac{4\pi}{3} \\ 0 & if |\omega| \geq \frac{4\pi}{3} \end{cases}$$

Cambiando la función auxiliar $\nu$ se obtiene una familia de funciones que generan un análisis ortogonal bajo ciertas condiciones para $\nu$.

**Wavelet Gauss.** Esta familia es construida partiendo de la siguiente función Gaussiana compleja:

$$f(x) = C_p e^{-ix} e^{-x^2}$$

**Wavelet Morlet.** Esta función es definida como:

$$\psi(x) = \frac{1}{\sqrt{\pi f_b}} e^{2i\pi f_c x} e^{-\frac{x^2}{f_b}}$$

y depende de dos parámetros: $f_b$ es el ancho de la ventana y $f_c$ es la frecuencia central de la wavelet.

ANEXO F

# TIPOS DE DISTANCIAS

El uso de distintos tipos de métricas para el cálculo de similitud puede ayudar en el desarrollo de un algoritmo con mayor eficacia, que permita resolver el problema que se haya planteado. En este apartado se definen los distintos tipos de métricas utilizadas en esta investigación.

**Distancia Euclidiana**

También llamada métrica Euclidiana (conocida en literatura más antigua como distancia de Pitágoras), es la distancia entre un punto *x* y un punto *y* que se encuentran sobre el espacio Euclidiano $E^n$, es definida como: [47]

$$d = \|x - y\|_2 = \sqrt{(x_1 - y_1)^2 + \cdots + (x_n + y_n)^2} = \sqrt{\sum_{i=1}^{p} |x_i - y_i|^2}$$

Lo cual expresa la raíz cuadrada de la suma de los cuadrados de las diferencias existentes entre las variables asignadas.

**Distancia SEuclidiana (distancia Euclidiana Estandarizada)**

Es una modificación de la distancia Euclidiana donde se obtiene la desviación estándar de *x* y *y*. [48]

**Distancia City-block**

Denominada también distancia Manhattan o métrica Taxicab. En esta métrica, la distancia entre dos puntos es la suma de las diferencias absolutas que existe entre todas las variables elegidas, definida como: [49]

$$d_{xy} = \sum_{i=1}^{p} |x_i - y_i|$$

**Distancia Minkowski**

En un espacia P-dimensional esta distancia se encuentra definida por: [47]

$$d = \left[ \sum_{i=1}^{p} (|x_i - y_i|)^r \right]^{1/r}$$

Cuando $r$ = 2 es considerada la distancia Euclidiana y en $r$ = 1 es denominada distancia Manhattan. Por tal motivo la distancia Minkowshi puede ser considerada como una generalización de estas dos.

**Distancia Chebyshev**

También denominada métrica máxima o métrica $L\infty$. La distancia Chebyshev entre dos vectores es la mayor de las diferencias, en valor absoluto, existente entre las variables consideradas, definida de la siguiente manera. [49]

$$d = Max|x_i - y_i|$$

**Distancia coseno**

Más que una distancia se le considera una medida de similitud. La medida coseno se utiliza comúnmente para determinar el grado de similitud que dos vectores poseen entre sí. El coseno es calculado como el producto escalar normalizado de los dos vectores, utilizando comúnmente la norma Euclidiana para normalizar. [50]

El producto escalar es calculado como:

$$a \cdot b = \sum_{i=1}^{N} a_i b_i$$

Mientras que la norma es definida como:

$$\|x\| = \sqrt{x \cdot x}$$

Y la medida de similitud coseno se define como:

$$coseno(a, b) = \frac{a \cdot b}{\|a\| \times \|b\|}$$

Sustituyendo por el producto escalar y por la norma, tenemos:

$$coseno(a, b) = \frac{\sum_{i=1}^{N} a_i b_i}{\sqrt{\sum_{i=1}^{N} a_i^2} \sqrt{\sum_{i=1}^{N} b_i^2}}$$

**Correlación**

La correlación es el grado de asociación existente entre dos variables. Este grado de correlación es cuantificado mediante el uso de coeficientes de correlación. En este caso se utiliza el coeficiente de correlación de Pearson que es una medida de la relación lineal definida como:

$$r = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{x_i - \bar{x}}{\sigma_x}\right)\left(\frac{y_i - \bar{y}}{\sigma_y}\right)$$

Donde $\bar{x}$ y $\bar{y}$ son las medias de *x* y *y* respectivamente. Mientras $\sigma_x$ y $\sigma_y$ corresponde a la desviación estándar de *x* y *y* respectivamente. [51]

**Spearman**

Denominado Spearman rank correlation permite medir la fuerza de asociación entre dos variables, es computacionalmente equivalente a los coeficientes de Pearson calculado para filas (en lugar de datos contiguos originales). Está definido como:

$$r_s = 1 - \frac{6T}{n(n^2 - 1)}$$

Donde $T = \sum_i (r_i - s_i)^2$ y es la suma de los cuadrados de la diferencia entre las filas de cada par de muestras. [52]

ANEXO G

# MAPA AUTO-ORGANIZADO

Los mapas auto-organizado (SOM) son un tipo de red neuronal artificial basados en el aprendizaje competitivo y no supervisado. Fueron escritos por primera vez por el profesor Teuvo Kohonen, motivo por el cual también son conocidos como mapas o redes de Kohonen. Este tipo de red neuronal es utilizada para la visualización y análisis de datos de alta dimensión. Son un medio de organización automático de datos, de modo que las entradas similares son mapeadas a estar cerca entre sí.

La arquitectura de los SOM está compuesta por dos capas. Una capa de entrada con N neuronas, una por cada unidad del vector de entrada; y una capa de salida con M neuronas conocidas cada una como "nodos" usualmente dispuestas en forma de malla (ver Figura G.1 con representación gráfica), donde se van mapeando los datos del vector de entrada de tal manera que los nodos más similares queden adyacentes. Cada nodo está constituido por un vector de pesos y para determinar en qué nodo se mapeara el vector de entrada, se calcula la distancia euclidiana (o cualquier otro tipo de distancia) existente entre el vector de entrada y cada uno de los vectores de pesos, resultando vencedor aquel nodo que posea la menor distancia. [53][54]

El algoritmo de los SOM es el siguiente:

- Seleccionar la topología de la red para la capa de salida.
    - Inicializar la distancia del vecino actual, D (0), en un valor positivo.
- Inicializar los pesos con valores aleatorios pequeños.

- Dejar $t$ = 1.
- Mientras no se superen los límites computacionales, hacer
  1. Seleccionar una muestra de entrada $i_l$.
  2. Calcular el cuadrado de la distancia euclidiana de $i_l$ a los vectores de peso ($w_j$) asociados a cada nodo de salida.

$$\sum_{k=1}^{n} (i_{l,k} - w_{j,k}(t))^2$$

  3. Seleccionar el nodo de salida $j^*$ que contiene el vector de peso con valor mínimo del paso 2.
  4. Actualizar los pesos de todos los nodos que se encuentran dentro de una distancia D(t) de $j^*$, utilizando la regla de actualización de peso:

$$w_j(t+1) = w_j(t) + \eta(t)\,(t_l - w_j(t))$$

  5. Incrementar $t$
- Terminación del mientras

  La tasa de aprendizaje generalmente disminuye con el tiempo:

$$0 < \eta(t) \leq \eta(t-1) \leq l5$$

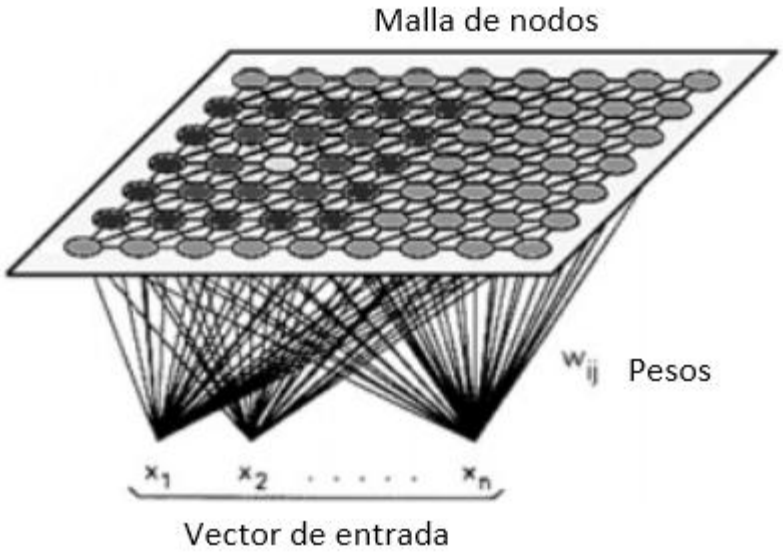



**Figura G.1:** Representación gráfica de arquitectura de SOM. Modificada de [53].

ANEXO H

# RESULTADOS: ANÁLISIS DE ATRIBUTOS DE MSD

En este apartado se detallan las pruebas realizadas para la identificación de aquellos atributos que fueran más representativos de MSD, utilizando solamente el grupo "analysis" de la base de datos ya que contiene las descripciones acústicas de las piezas musicales. De igual manera se muestran los resultados generados con dichas pruebas.

El subconjunto de datos con el que se trabajó está conformado por 4 títulos de canciones diferentes, todas con sus respectivos covers, este subconjunto es mostrado en la Tabla 3.2 del capítulo de Metodología. Los títulos fueron elegidos sobre otros, por ser aquellos que cuentan con un mayor número de covers realizados para una pieza musical.

Para realizar los experimentos se utilizaron los algoritmos perceptrón multicapa, Naive Bayes y Máquinas de Soporte Vectorial. Las pruebas consistieron en lo siguiente.

a) Cada versión representaba una clase.
    1. Se tomaron cuatro covers de cada título para extraer sus atributos. Se clasificaron con los algoritmos propuestos, donde cada clase representa un cover de la canción.
    2. De este conjunto se fueron eliminando atributos de uno en uno o en pares. Para posteriormente clasificarlos.
    3. Se repitió nuevamente el procedimiento aumentando el número de covers en 4.

La Figura H.1 muestra el porcentaje de instancias clasificadas correctamente utilizando Naive Bayes para las evaluaciones realizadas sobre la canción "Summertime" (utilizando 8, 16, 28 y 47 clases).

b) Cada instancia representaba los géneros de los covers.
   1. Se tomaron todos los covers de una canción.
   2. Se clasificaron de acuerdo a sus géneros, en el caso de "Summertime" solo hubo dos clases "blues" y "jazz".
   3. Se realizó la clasificación nuevamente, eliminando atributos de uno en uno o en pares.

En la Figura H.2 se muestra las diferencias entre los porcentajes de clasificación para "Summertime" utilizando dos clases que agrupan sus 47 covers.

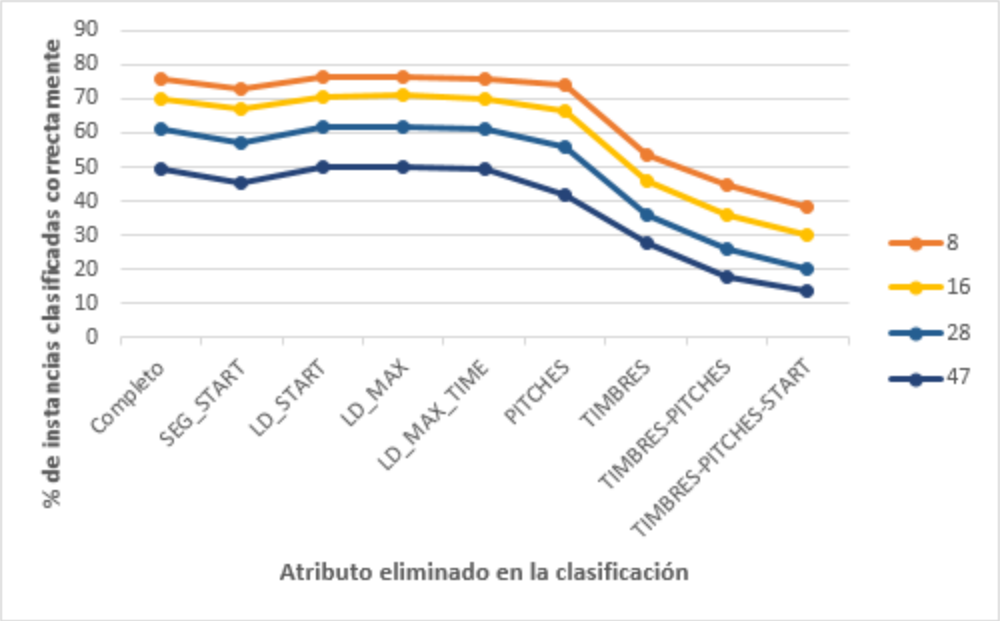



**Figura H.1:** Porcentaje de instancias clasificadas correctamente de la canción "Summertime" utilizando Naibe Bayes.

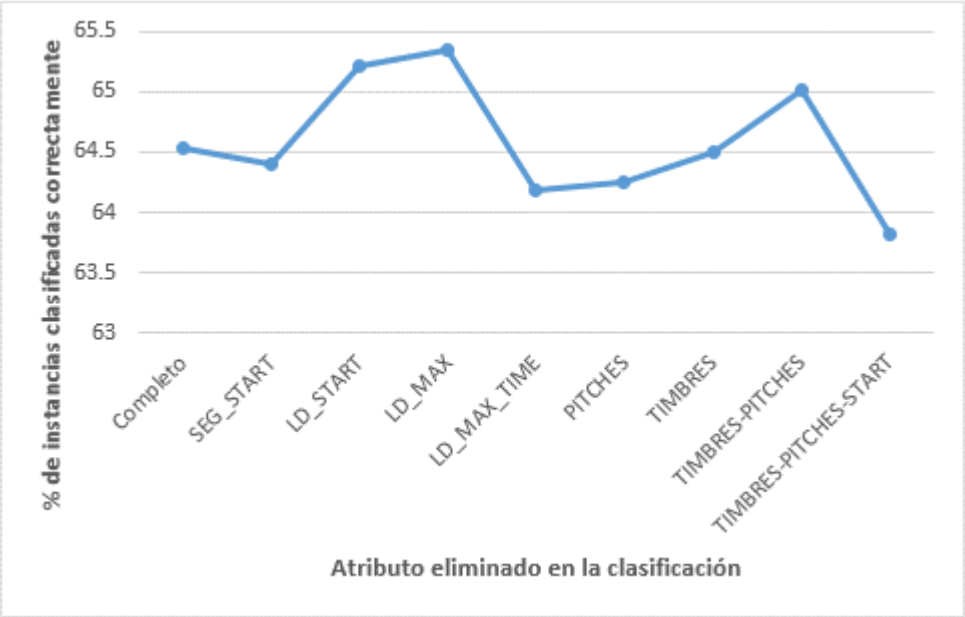



**Figura H.2:** Porcentaje de instancias clasificadas correctamente, de acuerdo a su género para la canción "Summertime", utilizando Naive Bayes como clasificador.

c) Se generan dos clases, es cover o no lo es.
   1. Se seleccionaron todas los covers de una canción, que genera la clase "es cover".
   2. Se generó la clase "no es cover" formada por el 25% del total de los covers de cada una de las tres canciones restantes, y se realizó la clasificación con los tres diferente algoritmos propuestos.
   3. Se realizó la clasificación nuevamente, eliminando atributos de uno en uno o en pares.
   4. Se repitió el procedimiento aumentando el porcentaje en 10, hasta llegar al 75%.

Se puede observar los resultados de estas pruebas para la canción "Summertime" en las Figuras H.3 y H.4. En la primera de estas se utilizó Naive Bayes como clasificador, mientras que en la segunda se muestran resultados utilizando un perceptrón Multicapa para clasificar el conjunto de datos.

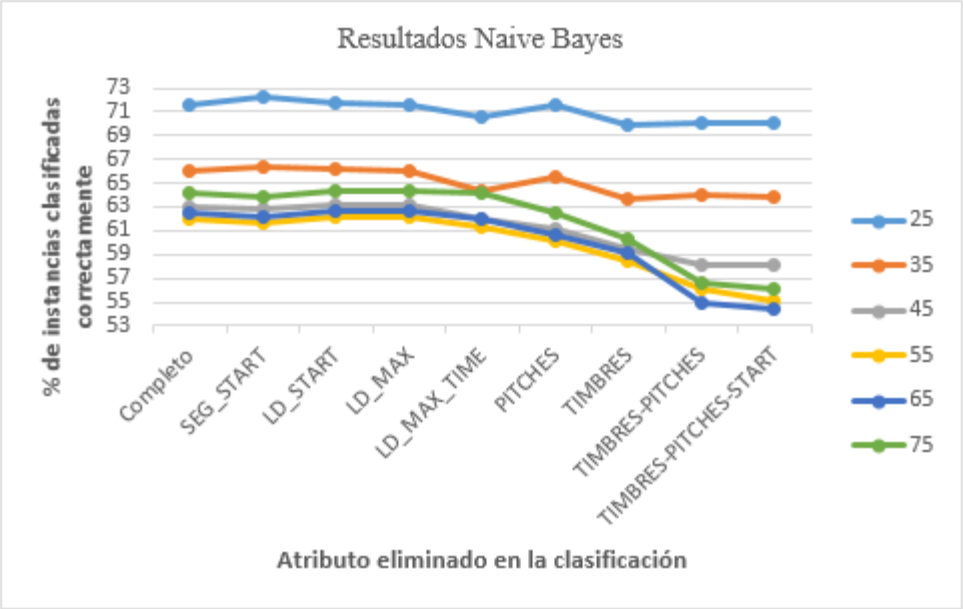



**Figura H.3:** Porcentaje de instancias clasificadas correctamente, con Naive Bayes, utilizando las clases es cover y no es cover para la canción "Summertime. Las leyendas indican el porcentaje de instancias utilizadas en la clasificación.

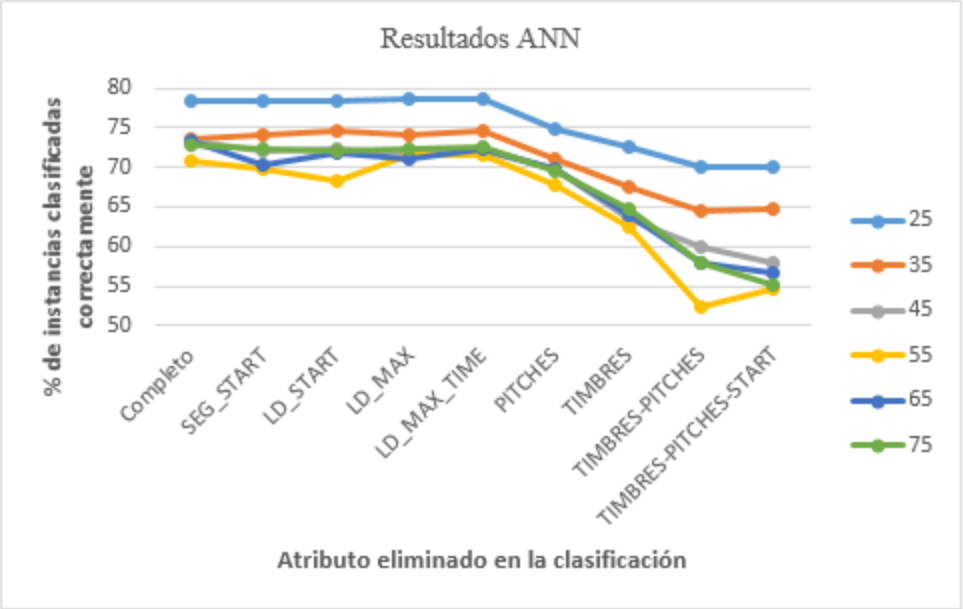



**Figura H.4:** Porcentaje de instancias clasificadas correctamente con una ANN, con solo las clases es cover y no es cover para la canción "Summertime. Las leyendas indican el porcentaje de instancias utilizadas en la clasificación.

Estas pruebas nos permitieron corroborar la importancia de los timbres y del pitches. Como se puede ver en las figuras mostradas con anterioridad se puede observar que mientras estas características existan dentro de la clasificación, el porcentaje de instancias clasificadas correctamente se mantiene constante, pero al momento de quitar estos atributos el porcentaje de clasificación decae.

# REFERENCIAS


[1] Humphrey E., Bello J.P., LeCun Y. “Moving Beyond Feature Design: Deep Architectures and Automatic Learning”. En Music Informatics, Proceedings of 13th International Society for Music Information Retrieval Conference ISMIR, pp. 403-408, 2012.

[2] Sociedad Mexicana de Productores de Fonogramas, Videogramas y Multimedia (Somexfon). <http://www.somexfon.com/> [consulta: 2015-06-09].

[3] Lee, K. “Identifying Cover Songs from Audio Using Harmonic Representation”. En Extended abstract submitted to Music Information Retrieval eXchanges task, BC, Canada, 2006.

[4] Jesper Højvang Jensen, Mads G. Christensen, Daniel P.W. Ellis, and Søren Holdt Jensen. “A Tempo-Insensitive Distance Measure For Cover Song Identification Based On Chroma Features”. En IEEE International Conference on Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. ISBN: 978-1-4244-1483-3, pp 2209 – 2212, Marzo 31, 2008.

[5] S. Ravuri and D. Ellis. “Cover Song Detection: From High Scores to General Classification”. En Proc. IEEE ICASSP, pp. 65-68, Dallas, Marzo 2010.

[6] Xiao Chuan; Sch. of Sci., Fudan Univ., Shanghai, China. “Cover song identification using an enhanced chroma over a binary classifier based similarity measurement framework”. En 2012 International Conference on Systems and Informatics (ICSAI). Print ISBN: 978-1-4673-0198-5, pp. 2170 – 2176, Mayo 2012.

[7] Thierry Bertin-Mahieux and Daniel P. W. Ellis. “Large-scale cover song recognition using hashed chroma landmarks”. En Applications of Signal

Processing to Audio and Acoustics (WASPAA), 2011 IEEE Workshop on. Print ISBN: 978-1-4577-0692-9. pp 117-120, Octubre 2011.

[8] T. Bertin-Mahieux, D. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2011), 2011.

[9] Alexander Schindler, Rudolf Mayer, Andreas Rauber. "Facilitating Comprehensive Benchmarking Experiments on the Million Song Dataset". En 13th International Society for Music Information Retrieval Conference, ISMIR 2012, pp. 469-474 Mosteiro S.Bento Da Vitória, Porto, Portugal, October 8-12, 2012.

[10] Peter Grosche and Meinard Maller. "Toward characteristic audio shingles for efficient cross-version music retrieval". En Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on. Print ISBN: 978-1-4673-0045-2, pp 473-476, Marzo 2012.

[11] Thierry Bertin-Mahieux, and Daniel P. W. Ellis. "Large-Scale Cover Song Recognition Using The 2D Fourier Transform Magnitude". En Proc. of the 13th International Society for Music Information Retrieval Conference, pp 241–246, 2012.

[12] Humphrey, E., Nieto, O., and Bello, J. "Data driven and discriminative projections for large-scale cover song identification". En Proceedings of the 14th International Society for Music Information Retrieval Conference, Noviembre 4-8, 2013.

[13] Martin, B., Brown, D., Hanna, P., and Ferraro, P. "BLAST for Audio Sequences Alignment: A Fast Scalable Cover Identification Tool". En Proceedings of the 13th International Society for Music Information Retrieval Conference. Porto, Portugal. Octubre 8-12, 2012.

[14] J. Osmalskyj, S. Piérard, M. Van Droogenbroeck, and J.J. Embrechts. "Efficient DataBase Pruning For Large-Scale Cover Song Recognition". In IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Vancouver, Canada. ISSN: 1520-6149, pp 714-718; Mayo 26-31, 2013.

[15] Khadkevich, M., and Omologo, M. “Large-scale cover song identification using chord profiles”. En Proceedings of the 14th International Society for Music Information Retrieval Conference, Noviembre 4-8, 2013.

[16] Salamon, J., Serrà J., & Gómez E. “Tonal Representations for Music Retrieval: From Version Identification to Query-by-Humming”. En International Journal of Multimedia Information Retrieval, special issue on Hybrid Music Information Retrieval, vol. 16, pp 45-58, 2013.

[17] Jan Van Balen, Dimitrios Bountouridis, Frans Wiering, Remco C. Veltkamp. “Cognition-inspired Descriptors for Scalable Cover Song Retrieval”. En 15th International Society for Music Information Retrieval Conference (ISMIR 2014), pp 379-384, Taipei, Taiwan, October 27-31, 2014.

[18] Serrà, J., Gómez, E., & Herrera, P. “Audio cover song identification and similarity: background, approaches, evaluation and beyond”. En Z. W. Ras & A. A. Wieczorkowska (Eds.) Adv. in Music Information Retrieval, Studies in Computational Intelligence, vol. 16, chap. 14, pp. 307–332. Berlin, Germany: Springer, 2010.

[19] SecondHandSongs dataset, the official list of cover songs within the Million Song Dataset, available at: http://labrosa.ee.columbia.edu/millionsong/secondhand

[20] Roy D. Patterson, Etienne Gaudrain, and Thomas C. Walters. "The Perception of Family and Register in Musical Tones". In Mari Riess Jones, Richard R. Fay, and Arthur N. Popper. Music Perception. Springer. ISBN 978-1-4419-6113-6. pp. 37–38; 2010.

[21] Anssi Klapuri and Manuel Davy. “Signal processing methods for music transcription”. Springer. ISBN 978-0-387-30667-4. Pp 8; 2006.

[22] Tristan Jehan, CSO, David DesRoches, Lead Audio Engineer. “Analyzer Documentation”. The Echo Nest Corporation. Disponible en: <http://developer.echonest.com/docs/v4/_static/AnalyzeDocumentation.pdf> [consulta: 2014-09-01].

[23] Christopher Hast. “Meter As Rhythm”. Oxford University Press, USA, 12/03/1997 – pp 129.

[24] Ron Kohavi, Foster Provost. “Glossary of Terms”. Machine Learning 30. 1998 Kluwer Academic Publishers, Boston, Manufactured in The Netherlands, pp. 271-274, 1998.

[25] Tom Mitchell. “The Discipline of Machine Learning”. CMU-ML-06-108, Carnegie Mellon University, 2006.

[26] Stuart Russell, Peter Norvig. “Artificial Intelligence: A Modern Approach (2nd ed.)”. Prentice Hall. ISBN: 978-0137903955. pp. 650. 2003, 1995.

[27] M. Narasimha Murty, V. Susheela Devi. “Pattern Recognition: An Algorithmic Approach”. Springer; Edición: 2011 (8 de julio de 2011). ISBN: 0857294946. pp. 86-87, 147, 229.

[28] Rodrigo Salas. “Redes Neuronales Artificiales”. Disponible en <http://www.inf.utfsm.cl/~rsalas/Pagina_Investigacion/docs/Apuntes/Redes%20Neuronales%20Artificiales.pdf> [conuslta: 2015-01-20].

[29] Rudolf Kruse, Christian Borgelt, Frank Klawonn, Christian Moewes, Matthias Steinbrecher, Pascal Held. “Computational Intelligence: A Methodological Introduction”. Springer; 2013 edition (March 28, 2013). ISBN: 978-1-4471-5013-8 (eBook). pp. 47-48.

[30] C.-W. Hsu, C.-C. Chang, C.-J. Lin. “A practical guide to support vector classification”. Technical report, Department of Computer Science, National Taiwan University. July, 2003.

[31] J. MacQueen. “Some methods for classification and analysis of multivariate observations”. En Proceedings of the Fifth Berkeley Symposium on Mathematics, Statistics and Probability, Vol. 1, pp. 281–296, 1967.

[32] Andrew Ng. “Sparse autoencoder”. CS294A Lecture notes <http://web.stanford.edu/class/cs294a/sparseAutoencoder.pdf> [consulta: 2015-02-19].

[33] Jesús Bernal Bermúdez, Jesús Bobadilla Sancho, Abraham Gutiérrez Rodríguez, Jorge Tejedor Cerbel, Víctor Martínez Hernando, Santiago Alonso Villaverde. “Enseñanza asistida de comunicación oral”. VIII congreso internacional de informática en la educación.

<http://espejos.unesco.org.uy/simplac2002/Ponencias/Inforedu/IE007%20Jes%FAs%20Bernal.doc> [consulta: 2015-07-25]

[34] Liliana R. Castro, Silvia M. Castro. "Wavelets y sus Aplicaciones". ler. Congreso Argentino de Ciencias de la Computación.

[35] Gregory E. Cox. "On the relationship between entropy and meaning in music: an exploration with recurrent neural networks". En Proceedings of the Annual Meeting of the Cognitive Science Society, 2010.

[36] <https://play.google.com/intl/ALL_es/about/music/> [consulta: 2015-10-08].

[37] Million Song Dataset, official website by Thierry Bertin-Mahieux, available at: <http://labrosa.ee.columbia.edu/millionsong/> [consulta: 2015-02-02].

[38] Baldi, P. "Autoencoders, Unsupervised Learning, and Deep Architectures". Workshop on Unsupervised and Transfer Learning. JMLR: Workshop and Conference Proceedings. 2012, pp. 27:37–50

[39] E. Oran Brigham. "The Fast Fourier Transform". Copyright © 1974 Prentice-Hall Inc, Englewood Cliffs, New Jersey. ISBN: 012307496X. pp. 3-4.

[40] Sheng, Y. "The Transforms and Applications Handbook: Second Edition", CRC Press LLC, 2000. Chapter 10 Wavelet Transform.

[41] Michel Misisti, Yves Misiti, Georges Oppenheim, Jean-Michel Poggi. "Wavelets and their Applications". Wiley-ISTE. ISBN: 978-1-905209-31-6. May 2007. pp. 93-108.

[42] Altman, N. S. (1992). "An introduction to kernel and nearest-neighbor nonparametric regression". The American Statistician 46 (3): 175–185.doi:10.1080/00031305.1992.10475879

[43] Teuvo Kohonen and Timo Honkela (2007). "Kohonen network". Scholarpedia, 2(1):1568., revision #122029

[44] <http://es.mathworks.com/company/aboutus/> [consulta: 2016-03-03].

[45] <http://www.amolgmahurkar.com/index.html#Codes> [consulta: 2015-02-19].

[46] <http://www.cis.hut.fi/projects/somtoolbox/download/> [consulta: 2016-03-03].

[47] Richard O. Duda, Peter E. Hart, David G. Stork. "Pattern Classification". 2nd Ed., Wiley, 2001. pp. 36, 188.

[48] Deza, Elena; Deza, Michel Marie (2009). Encyclopedia of Distances. Springer, pp. 331.

[49] David M. J. Tax, Robert Duin, and Dick De Ridder (2004). Classification, Parameter Estimation and State Estimation: An Engineering Approach Using MATLAB. John Wiley and Sons, pp. 359, 360.

[50] Sidorov, Grigori; Gelbukh, Alexander; Gómez-Adorno, Helena; Pinto, David. "Soft Similarity and Soft Cosine Measure: Similarity of Features in Vector Space Model". Computación y Sistemas 18 (3): 491–504. doi:10.13053/CyS-18-3-2043

[51] John F. Kenney. "Mathematics of Statistics part I". Chapman & Hall Ltd. 1939. pp. 159-163.

[52] P . Sprent and N . C . Smeeton. "Applied Nonparametric Statistical Methods, Third Edition". Chapman & Hall/CRC. ISBN: 1-58488-145-3.

[53] Nasim Osouli, Esmaeil Khoshbakht, Zinat Ansari, Mahdi Kazemi. "Using Self-Organizing Map (SOM) algorithm for Analysts' equities clustering". Adv. Environ. Biol., 8(1), 185-189, 2014.

[54] Teuvo Kohonen. "Essentials of the self-organizing map". Neural Networks. Volume 37, January 2013, pp. 52–65.