Universidad Carlos iii de Madrid

Escuela Politécnica Superior

Ingeniería de Telecomunicación

PROYECTO FIN DE CARRERA

DISEÑO DE UN CLASIFICADOR DE GÉNEROS MUSICALES BASADO EN MODELOS DINÁMICOS

Autor: Alberto García Durán
Tutor: Dr. Emilio Parrado Hernández

Julio de 2011

"*¿Pur qué?*"
- José Mourinho

Agradecimientos

Tras muchos exámenes y mucho esfuerzo he conseguido llegar al final de este importante periodo de mi vida con la realización de este proyecto. Durante estos años ha habido momentos muy buenos así como algunos más duros, pero en todos ellos siempre he estado rodeado de gente muy importante para mí.

En primer lugar, me gustaría agradecerles a mis papis, Mari Tere y Antonio, todo lo que me han dado. Soy consciente de todos los sacrificios que han hecho para poder darnos a mi hermano y a mí todo lo que necesitábamos y más. Muchas gracias. En particular a mi madre me gustaría agradecerle su gigantesco esfuerzo y todo lo que me/nos ha soportado durante tanto tiempo, aunque siempre ha tenido un rato para aguantar nuestras chanzas. Aunque nunca diga nada soy consciente de todo ello. Y a mi padre me gustaría agradecerle haberme hecho siempre querer superarme aunque lo hiciera de su manera tan peculiar. Quizás sin su manera de alentarme no hubiera tenido el afán de superación que tengo ahora, ya que siempre quería que destacara en todo lo que hiciera.

A mi *bro* Carlos, pese a que tiene un gusto nefasto para la vestimenta, me ha entretenido cientos y cientos de horas con nuestras tonterías. Aunque de vez en cuando haya algún pique, siempre ha habido *feeling*.

A Laura, por ser una de las personas más importantes de mi vida. Han sido innumerables los buenos momentos que hemos vivido juntos... y los que faltan por llegar!! Gracias!!

No me quiero olvidar de mis compañeros universitarios. Lo fácil sería que diera nombres y nombres, pero prefiero nombrar pocos pero muy importantes. Gracias Javi Frutos y Amaya.

Por último, quería agradecer a mi tutor Emilio por haberme dado la oportunidad de trabajar con él. En este tipo de trabajos, además de que la temática del mismo te debe atraer, es necesario encontrarte cómodo con la persona con quien trabajas, y he de decir que en mi caso tanto la relación profesional como la personal han sido inmejorables. Además querría agrade-

cerle especialmente que me haya permitido trabajar durante este curso en su grupo de investigación. Ha sido un honor para mí.

Resumen

Hoy en día, en un escenario con grandes cantidades de música digitalizada y la existencia de reproductores portátiles con gran capacidad de almacenamiento, se hace necesario el uso de sistemas de gestión y navegación que ayuden al usuario a organizar su música. Una buena manera de realizar esto es creando herramientas capaces de determinar automáticamente el género musical al que pertenece cada canción.

En este trabajo se estudian distintas alternativas con el objetivo de conseguir un clasificador de géneros musicales. Habitualmente, este problema se trataría como un problema de clasificación de datos i.i.d.. Sin embargo, con una adecuada caracterización de los fragmentos musicales es posible crear modelos dinámicos que capten la evolución temporal de los fragmentos.

Intentando aprovechar esta información temporal se propone una arquitectura general para solucionar este problema de clasificación, así como distintas estrategias en cada módulo de la arquitectura. Finalmente evaluamos las prestaciones de las distintas estrategias en una base de datos real, concluyendo con una aproximación de cómo debe se debe diseñar un clasificador que solucione este problema.

Palabras clave: música, clasificación, modelos dinámicos, modelos ocultos de Markov, bolsas de palabras.

Abstract

Due to the immense amount of digital music and to the existence of high storage capacity portable media players, the usage of organization and navigation systems that help users organize the music has become essential nowadays. A good way to achieve this is by creating tools capable of determining the genre of each song automatically.

The aim of this work is to implement a music genre classifier exploring different alternatives. This case os study is normally addressed as an usual multi-class problem with i.i.d. data. However, using a proper characterization of the musical fragments it is possible to create dynamic models that are able to capture the evolution in time of these fragments.

Using this time information, the project presents a general architecture solution with different strategies in each module to solve the classification problem. The performance of all the proposed strategies will be evaluated using real world data to conclude with an approach on how a classifier capable of solving this problem should be designed.

Keywords: music, classification, dynamic models, hidden Markov models, bags of words.

Índice general

1. Introducción **3**
1.1. Objetivos 5
1.2. Estructura de la memoria 6

2. Estado del arte **7**
2.1. Vista general del aprendizaje máquina 7
2.1.1. Una taxonomía del aprendizaje máquina 8
2.1.2. Otros aspectos del ML 9
2.1.3. Aplicaciones del ML 11
2.2. Extracción de características de bajo nivel en aplicaciones de audio 12
2.2.1. *Mel Frequency Cepstral Coefficients* 12
2.2.2. Modelos AR 15
2.3. Modelos ocultos de Markov 16
2.3.1. Introducción 16
2.3.2. Procesos de Markov discretos 17
2.3.3. Extensión a los HMM 19
2.3.4. Elementos de un HMM 20
2.3.5. Los tres problemas básicos para los HMMs 22
2.3.5.1. Solución al Problema 1 23
2.3.5.2. Solución al Problema 2 28
2.3.5.3. Solución al Problema 3 30
2.3.6. Caso particular: funciones de densidad continuas en los HMM 35
2.4. *Bag of words* 37
2.5. Clasificación automática 39
2.5.1. Algunos aspectos de la teoría de aprendizaje estadístico 39

2.5.2. Máquinas de vectores soporte lineales para clasificación 41
2.5.2.1. Caso separable 42
2.5.2.2. Caso no separable 48
2.5.3. Máquinas de vectores soportes no lineales para clasificación 53
2.5.4. K-Vecinos más próximos 56
2.5.4.1. Algoritmo 57
2.5.4.2. Selección de parámetro K 57
2.5.4.3. Propiedades 59
2.5.5. *Kernel Fisher Discriminant* 59
2.5.5.1. Descripción matemática del algoritmo 60
2.5.5.2. Esencia de la transformación KFD: KPCA + LDA 63
2.6. Clasificadores multiclase a partir de clasificadores binarios . . 64
2.6.1. Reducción del coste de clasificación 65

3. Diseño del clasificador de géneros musicales 67
3.1. Antecedentes 67
3.2. Diseño del clasificador 69
3.2.1. Arquitectura del clasificador 69
3.2.2. Estrategias en el procesamiento de bajo nivel 70
3.2.3. Estrategias en el procesamiento de alto nivel 72
3.2.4. Estrategias en la etapa de clasificación 77

4. Resultados experimentales 79
4.1. Base de datos 79
4.2. Descripción de los experimentos 80
4.3. Validación de la propuesta 80
4.3.1. Resultados 80
4.3.1.1. Discusión en: Procesamiento de bajo nivel . . 85
4.3.1.2. Discusión en: Procesamiento de alto nivel . . 86
4.3.1.3. Discusión en: Etapa de clasificación 87
4.4. Conclusiones 87
4.5. Futuras líneas de investigación 88

1. Presupuesto **91**

1.1. Coste del material 91

1.2. Coste de honorarios 92

1.3. Presupuesto total 93

Lista de Figuras

2.1. Taxonomía del aprendizaje máquina. 9
2.2. Proceso para crear características MFCC. 13
2.3. Escalado y suavizado de los coeficientes Mel. 14
2.4. La escala Mel. 15
2.5. Una cadena de *Markov* con 5 estados. 18
2.6. Ilustración de la secuencia de operaciones requeridas para el cálculo de la variable $\alpha_{t+1}(j)$. 25
2.7. Implementación del cálculo de los $\alpha_t(i)$ en términos de una rejilla de observaciones t y estados i. 26
2.8. Ilustración de la secuencia de operaciones requeridas para el cálculo de la variable $\beta_t(i)$. 28
2.9. Ilustración de la secuencia de operaciones requeridas para el cálculo del evento conjunto de que el sistema esté en el estado S_i en el instante t y en el estado S_j en el instante $t+1$. 31
2.10. Tabla de frecuencia de términos para minería de textos. 38
2.11. Frecuencia de las distintas texturas presente en una imagen. . 38
2.12. Cuatro posibles soluciones para el caso de separación lineal . . 42
2.13. Solución de la SVM para el caso separable 43
2.14. Posibles violaciones que ocurren en el caso no separable 49
2.15. Tres funciones de pérdidas: escalón, *hinge* y cuadrática 51
2.16. Transformación de un espacio de dimensión 2 a uno de dimensión 3. 53
2.17. Ejemplo de clasificación K-NN. Según el valor de K la muestra de test se clasificará como triángulo rojo o como cuadrado azul. 58

3.1. Arquitectura para un clasificador de géneros musicales. 69
3.2. Vector MAR. 71

3.3. Estrategias basadas en HMM en el módulo de procesamiento de alto nivel. 76

Lista de Tablas

4.1. Características de las pruebas realizadas. 84
4.2. Resultados de las distintas estrategias. 85
4.3. Resultados para la discusión en: Procesamiento de bajo nivel. 86

1.1. Coste de los materiales. 92
1.2. Coste de los honorarios. 93
1.3. Coste total del proyecto. 94

Prólogo

El 11 de Marzo de 1997, el para muchos mejor ajedrecista de la historia, Garry Kasparov, cayó derrotado ante la supercomputadora Deep Blue. El perfeccionamiento de las rutinas de aprendizaje máquina, entre otros aspectos, inclinó la balanza a favor de los chips. Pero esto no es suficiente, el objetivo real no es conseguir que la máquina "venza" al ser humano, sino que se comporte de manera similar a éste. En palabras de Marvin Minsky (experto en IA):

> *"Una vez que contemos con programas que tengan capacidad real de autoaprendizaje, estará garantizado un rápido desarrollo. Ya que la máquina se mejorará a sí misma así como a su modelo, estaremos en posición de observar todos los fenómenos conectados con los conceptos de razón, inteligencia y consciencia. Aunque es difícil decir cuándo se producirá tal desarrollo, no cabe duda de que cambiará el mundo"*

El empeño que muestran los científicos en intentar otorgar a las máquinas la característica posiblemente más distintiva de la inteligencia humana, el aprendizaje, está orientado hacia la consecución de programas que mejoren automáticamente con la experiencia. Un empeño que proporciona sus frutos cada día, pero que está muy lejos del objetivo definido por Marvin Minsky.

Sucesos como éste despertaron en mí un interés inusual que me llevaron a elegir un proyecto fin de carrera relacionado con esta temática: el aprendizaje máquina.

Así que decidí buscar un PFC que me permitiera consolidar y aumentar mis conocimientos en esta área, y poder aplicarlos a un caso práctico, un caso útil. Esta oportunidad se me presentó con el proyecto que he tenido la suerte de poder realizar:

DISEÑO DE UN CLASIFICADOR DE GÉNEROS MUSICALES BASADO EN MODELOS DINÁMICOS

Capítulo 1

Introducción

Hoy en día, existe un interés en aumento en los métodos automáticos para la organización y navegación de la música, algo que es principalmente motivado por la disponibilidad de gran cantidad de material musical en formato digital. La distribución de música no está limitada a medios físicos, sino que muchos usuarios se han convertido en usuarios frecuentes de servicios online como *Amazon* o *iTunes*, descargando música usando formatos de codificación estándar como MP3 o AAC. Además, la capacidad de los actuales reproductores portátiles permite a los usuarios almacenar sus colecciones personales de música y llevarlas a cualquier lugar.

En este contexto, resulta valioso disponer de métodos automáticos que infieran similitudes entre piezas musicales para desarrollar herramientas que ayuden al usuario a organizar y escuchar su música, así como para aumentar la efectividad de sistemas de recomendación musical, consiguiendo mejorar la experiencia del usuario cuando use estos servicios.

Es obvio que el criterio de similitud musical puede definirse de maneras muy diferentes: Por ejemplo, puede estar basado en el género, el ritmo o los instrumentos presentes en la pieza musical, por citar algunos ejemplo. Estas similitudes radican cada una en distintas escalas temporales. Por ejemplo, las características perceptuales como el ritmo pueden estudiarse directamente con pequeñas ventanas temporales, mientras que para la extracción del

género se requerirá de la revisión de toda la pieza musical.

En este proyecto se ha considerado que la similitud entre las canciones está basada en el género musical al que pertenecen. Esto resulta de interés puesto que es bastante común realizar una organización de la música disponible en géneros musicales. El inconveniente de este criterio de similitud es que es bastante subjetivo, pudiendo variar bastante dependiendo del instante temporal en el que se encuadre una canción o incluso dependiendo del artista. Este hecho hace más difícil la tarea de realizar un clasificador automático de géneros musicales.

Habitualmente este problema de clasificación se trata como un problema cualquiera de clasificación multiclase. En esta situación, la arquitectura de los clasificadores está formada por dos módulos: uno de procesado a bajo nivel y otro de clasificación. El primer módulo consistiría en la parametrización y extracción de características de la señal; una aproximación típica es la de dividir la señal de audio en varias ventanas de corta duración, de las cuales se extraen las características temporales del audio a corto plazo (ejemplos de tales características pueden ser Mel Frecuency Cepstral Coefficientes (MFCC) o Zero Crossing Rate (ZCR)). Sin embargo, a pesar de estar relacionadas con las características perceptuales de la señal de audio, estas características no contienen demasiada información valiosa sobre el género. Por tanto, con el objetivo de conseguir un mejor clasificación se realiza una integración temporal de las características ([1] y [2]). Estas nuevas características obtenidas ahora están más correlacionadas con la información del género musical. Este conjunto de características se pasan al segundo módulo, donde el clasificador establecerá el género de la canción mediante el voto mayoritario de la salida de todos los clasificadores para todos los segmentos de una canción. Ejemplos de éstos clasificadores pueden ser [3], [4], [5], OAR ([6]) o OAO ([7]).

Uno de los puntos débiles de estos procedimientos es la no captura de propiedades dinámicas de cada canción. En [8] se propone un procedimiento de clasificación que introduce un módulo de procesado a alto nivel que, a diferencia de los algoritmos anteriores, intenta aprovechar la información de

la dinámica temporal de las canciones.

1.1. Objetivos

Grosso modo, este proyecto se basará en el estudio de una base de datos reales compuesta de fragmentos musicales de unos 50 segundos. El trabajo comprenderá distintas fases en las que se intentarán responder a variadas cuestiones, con el objetivo final de encontrar un procedimiento adecuado para predecir el género musical al que pertenece un cierto fragmento musical.

Con el objetivo de diseñar un clasificador de géneros musicales se han tenido que realizar 3 etapas fundamentales:

1. Construcción de una arquitectura genérica.
2. Evaluación de distintas alternativas para los módulos de la arquitectura.
3. Verificación experimental.

En este trabajo, se propone una arquitectura de tres niveles: procesamiento a bajo nivel, a alto nivel y una etapa final de clasificación. Para evaluar la mejor estrategia a seguir en el diseño de un clasificador de géneros musicales, se propondrán distintas alternativas en cada nivel. Algunas de las cuestiones que estas alternativas nos responderán serán:

- Una vez parametrizado un fragmento musical, ¿son todos los coeficientes igual de importantes a la hora de realizar la clasificación? ¿se pueden eliminar coeficientes sin empeorar o incluso mejorar la tasa de acierto?
- ¿Utilizar estrategias que intenten capturar la dinámica temporal de las canciones mejora los resultados?
- ¿Utilizar un módulo de procesamiento de alto nivel en la arquitectura es mejor siempre que no utilizarlo?

- ¿Realizar un aprendizaje supervisado en el procesamiento de alto nivel es mejor que uno no supervisado?
- ...

1.2. Estructura de la memoria

En el Capítulo 2 se da, en primer lugar, una visión general del aprendizaje máquina. Esta visión permitirá al lector no iniciado en el aprendizaje máquina conocer las nociones básicas de en qué consisten, cómo se clasifican, cómo se usan y para qué sirven el conjunto de técnicas que engloba este área. Posteriormente, y más orientado hacia el lector familiarizado con este área pero no conocedor de alguna o de la totalidad de las herramientas utilizadas en este trabajo, se presentarán las distintas herramientas empleadas en este proyecto. Estas herramientas son variadas y nos permitirán desde la parametrización de una base de datos, pasando por la creación de modelos dinámicos, hasta realizar clasificación de diversas formas.

El Capítulo 3 presentará la arquitectura genérica propuesta para resolver este problema de clasificación, así como las distintas alternativas en los diferentes niveles en los que se ha desglosado a un clasificador de géneros musicales. Esto nos permitirá realizar un análisis sistemático tanto de diversas estrategias que aprovechan la información temporal como de otras que no.

En el Capítulo 4 se muestran los resultados obtenidos con distintas estrategias, en las que, en cada una, se han combinado diferentes alternativas en cada nivel. En base a éstos, se discutirá cuál parece ser la mejor estrategia a seguir en cada nivel para construir un clasificador de géneros musicales que dé las mejores prestaciones en términos de tasa de acierto. Además se sugerirán directrices dirigidas a diseñadores de clasificadores de géneros musicales para mejorar las prestaciones de sus diseños, así como líneas de investigación para futuros trabajos.

Capítulo 2

Estado del arte

Este capítulo persigue dos objetivos: dar al lector una visión general del aprendizaje máquina, y presentar en detalle las diferentes herramientas utilizadas en este trabajo.

Algunas de estas herramientas utilizadas son: los modelos ocultos de Markov (HMM), útiles para caracterizar sistemas dinámicos; y máquinas de vectores soporte o K-vecinos más próximos, entre otros, que servirán de alternativas a implementar en el módulo de clasificación de la arquitectura propuesta.

Por último, como en realidad, en este trabajo, los problemas de clasificación son multiclase se introducirá al lector en posibles maneras de combinar clasificadores binarios, en este caso máquinas de vectores soporte, para solucionar problemas multiclase.

2.1. Vista general del aprendizaje máquina

En esta sección se proporcionará una visión general acerca del aprendizaje máquina, especialmente útil para el lector no familiarizado en este área.

¿Qué es el aprendizaje máquina? Es la pregunta que se deberá contestar antes de adentrarnos en la taxonomía de los métodos de aprendizaje máquina

(*Machine Learning*, ML). El ML es una rama de la Inteligencia Artificial (*Artificial Intelligence*, AI) cuyo objetivo es desarrollar técnicas que permitan crear modelos capaces de generalizar comportamientos a partir de una información no estructurada suministrada en forma de ejemplos. Es, por lo tanto, un proceso de inducción del conocimiento. En muchas ocasiones el campo de actuación del ML se solapa con el de la estadística, ya que las dos disciplinas se basan en el análisis de datos.

El ML tiene una amplia gama de aplicaciones, incluyendo motores de búsqueda, diagnósticos médicos, detección de fraude en el uso de tarjetas de crédito, análisis del mercado de valores, clasificación de secuencias de ADN, reconocimiento del habla y del lenguaje escrito, juegos y robótica.

2.1.1. Una taxonomía del aprendizaje máquina

Como disciplina compleja, los distintos algoritmos del ML podrían ser clasificados de muy distinta manera. Aquí se va a presentar un posible árbol de clasificación (ver Figura 2.1) que divide a las técnicas en 3 grupos principales.

1. Aprendizaje supervisado: aquellas técnicas a las cuales hay que presentar, en primer lugar, un conjunto de entrenamiento que incluye tanto las entradas como las salidas deseadas, permitiendo, de esta manera, aprender una función que relacione entradas y salidas. La máquina deberá poder ser capaz de generalizar para ejemplos nuevos [9].

2. Aprendizaje no supervisado: aquí el problema a resolver es encontrar la estructura subyacente de un conjunto de datos [10]. No se le presenta a las técnicas de este tipo de aprendizaje el conjunto de salidas deseadas.

3. Aprendizaje por refuerzo: en sistemas cuyas salidas son una secuencia de acciones. Lo que busca este tipo de aprendizaje es aprender de aquellas "políticas" que generaron secuencias de acciones que consiguieron lograr sus objetivos y evaluar nuevas "políticas" [11].

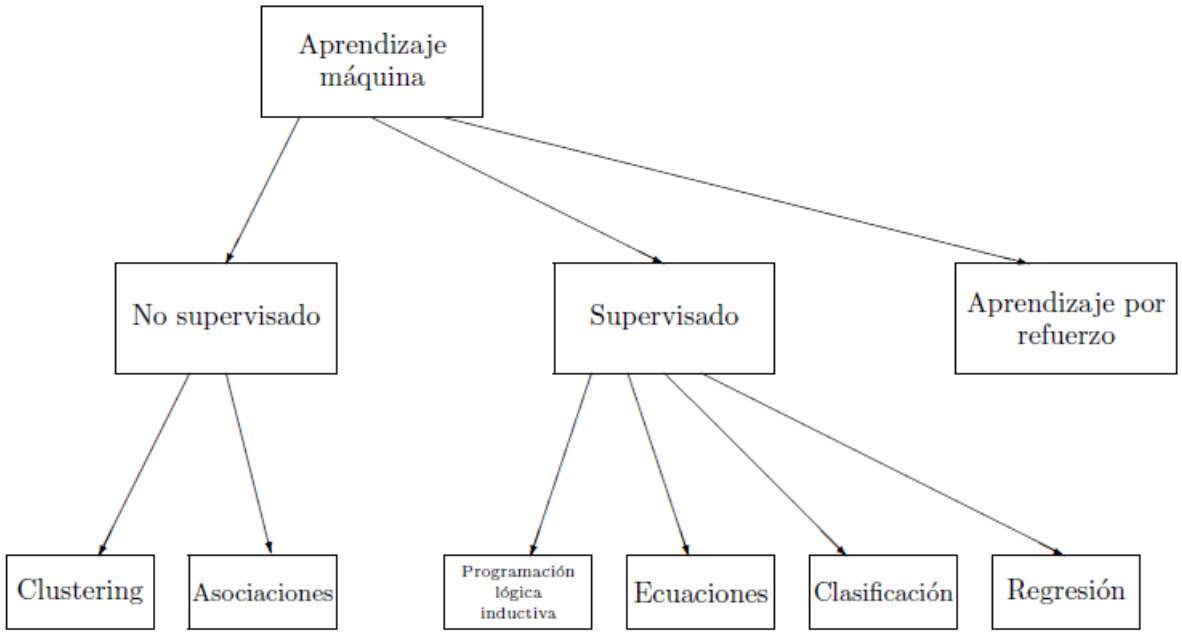

Figura 2.1: Taxonomía del aprendizaje máquina.

Taxonomía extraída de [12].

Para una información más detallada acerca de los distintos tipos de aprendizaje máquina consultar [12] y [13].

2.1.2. Otros aspectos del ML

Vectores de entrada y salida Los primeros están formados por componentes. Los valores de éstos pueden ser de 3 tipos (números reales, números discretos o valores categóricos).

De igual manera que en el caso de entrada, las salidas podrán ser números reales, discretos o valores categóricos. En este trabajo, por ejemplo, las salidas serán valores categóricos que indicarán el género musical.

Incertidumbre La incertidumbre hace referencia a la posibilidad de que una muestra sea etiquetada de manera errónea. Por ejemplo, en este problema cada canción tiene asociada una salida que corresponde al género musical al que pertenece, y como se ha comentado con anterioridad, este género puede depender de varios factores: época, artista, persona que toma la decisión de englobar a cada canción en un género...

Régimen de entrenamiento Según la manera en la que los datos de entrenamiento son presentados a la máquina:

1. *Batch*: todos los datos son dados a la máquina al principio del aprendizaje.
2. *On-line*: se les presentan los datos de uno en uno. Para cada entrada, da una estimación de la salida antes de conocer la salida deseada. Con cada nuevo ejemplo la máquina se irá actualizando.

Y según la interpretación de la salida obtenida:

1. Modelo generativo: obtiene el modelo que determina la probabilidad de la salida conociendo la entrada, lo que comúnmente se conoce como $p(y|x)$ $(y \in -1, +1, \dots)$.
2. Modelo discriminativo: el objetivo no es modelar a los datos analíticamente, sino tener un modelo capaz de etiquetar o clasificar a cada muestra en categorías.

Todos estos aspectos se encuentran detallados en [10].

Evaluación de los resultados Es importante tener métodos para poder evaluar la calidad del aprendizaje. Una manera ampliamente usada para ello es realizar validación cruzada sobre el conjunto de datos, que consiste en una técnica estadística para permitir que los resultados de un cierto análisis estadístico puedan ser generalizado a otro conjunto de muestras "generado" por la misma fuente de información. Es decir, la validación cruzada permite

obtener una estimación muestral del error promedio de test.

Hay varias formas de realizar la validación cruzada. Dos de las más usadas son:

1. *K-fold cross-validation*: en primer lugar se fija el parámetro K. Este parámetro dividirá la base de datos en K subconjuntos no solapados, de modo que en cada ocasión se utiliza una de las K particiones como conjunto de test y las $K-1$ restantes como conjunto de entrenamiento. La ventaja de realizar esta forma de validación cruzada es que todos los datos de entrenamiento son usados tanto para entrenamiento como para test.

2. *Repeated random sub-sampling validation*: este método divide aleatoriamente la base de datos en conjunto de entrenamiento y conjunto de test. El modelo se ajusta con el primer conjunto, mientras que con el segundo se evalúa la calidad del modelo entrenado. Este proceso se repite en varias ocasiones para luego poder promediar los resultados.
   La desventaja de este método es que una muestra puede no ser nunca seleccionada como parte del conjunto de test, mientras que otras pueden ser seleccionadas en varias ocasiones.
   La ventaja respecto a *K-fold cross-validation* es que se ha demostrado que ésta es asintóticamente consistente [14], resultando en predicciones más pesimistas para el conjunto de test que con *K-fold cross-validation*.

2.1.3. Aplicaciones del ML

Las áreas sobre las que el aprendizaje máquina puede ser empleado son innumerables. Ejemplos de éstas pueden ser:

- Sanidad: cómo diagnosticar mejor una enfermedad.
- Domótica: cómo autorregular eficientemente el consumo de energía, cómo limpiar el suelo...
- Banca: por qué conceder un crédito hipotecario, cuándo aceptar el cargo a una tarjeta...

- Marketing: qué tipo de personas compran cerveza los viernes, qué perfil de compra tiene la familia Martínez...
- Personalización: qué tipo de noticias le gusta leer los viernes por la mañana al usuario.
- Seguridad: qué perfil de uso de Linux tiene un determinado usuario, cuándo se produce una entrada ilegal...
- Videojuegos: en qué orden le gusta explorar el terreno al usuario.
- Sistemas de recuperación de información.

2.2. Extracción de características de bajo nivel en aplicaciones de audio

2.2.1. *Mel Frequency Cepstral Coefficients*

Los MFCC han sido las características dominantes usadas para reconocimiento del habla durante bastante tiempo. Su éxito se debe a la habilidad para representar el espectro de la amplitud del habla de una manera compacta. Cada paso en el proceso de creación de estas características está motivado por consideraciones perceptuales o computacionales. A continuación se examinan estos pasos con más detalle. Para una descripción más completa del proceso consúltese [15].

La Figura 2.2 muestra el proceso de creación de los MFCC. El primer paso es el de dividir la señal de habla en *frames*, normalmente aplicando una función de enventanado en intervalos fijos. El objetivo aquí es el de modelar pequeñas secciones de la señal que son estadísticamente estacionarias. La función de enventanado típicamente es una ventana de Hamming, para eliminar el efecto de los bordes. De manera que se genera un vector de características cepstrales para cada *frame*.

El paso siguiente es el de realizar la transformada discreta de Fourier (DFT) para cada *frame*. A continuación se toman logaritmos sobre el es-

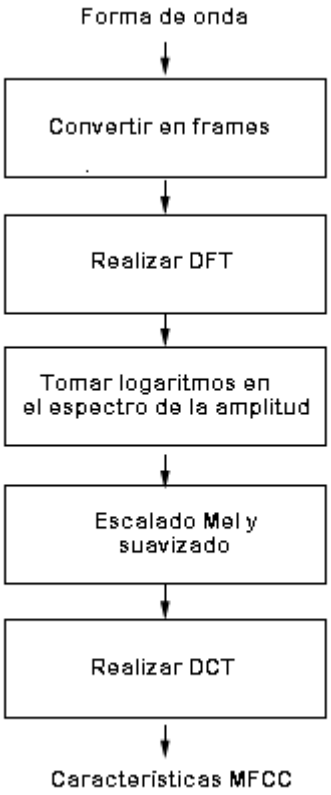

Figura 2.2: Proceso para crear características MFCC.

pectro de la amplitud. Se descarta la información de la fase porque estudios perceptuales han demostrado que la amplitud del espectro es mucho más importante que la fase. El hecho de tomar logaritmos sobre el espectro de la amplitud se debe a que se ha averiguado que el volumen percibido de la señal se puede aproximar de manera logarítmica [16].

El penúltimo paso consiste en suavizar y enfatizar las frecuencias más significativas perceptualmente. Esto se consigue "resumiendo" la totalidad de los componentes espectrales en un menor número de puntos frecuenciales como se muestra en la Figura 2.3. Aunque uno esperaría que estos puntos estuviesen equiespaciados en frecuencia, se ha averiguado que para el habla, las frecuencias más bajas son perceptualmente más importantes que las frecuencias más altas. Por tanto, el espaciado sigue la llamada escala frecuencial Mel.

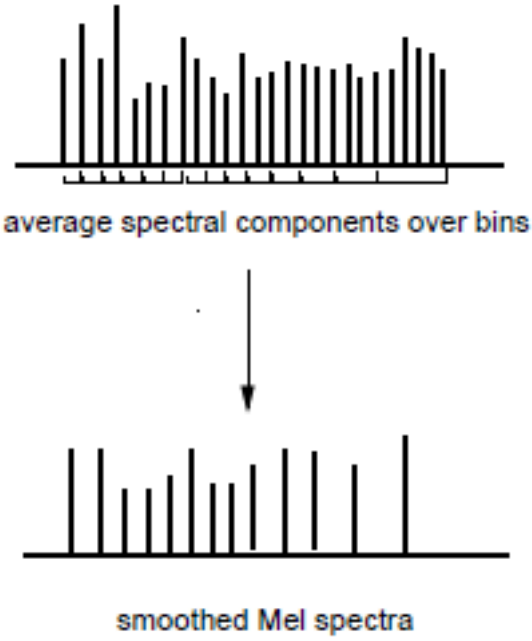

Figura 2.3: Escalado y suavizado de los coeficientes Mel.

La escala Mel está basada en una relación entre la frecuencia real y el tono percibido, ya que el sistema auditivo humano no funciona de manera lineal. Para aproximar esta escala normalmente se usa un banco de filtros.

La Figura 2.4 muestra la función Mel.

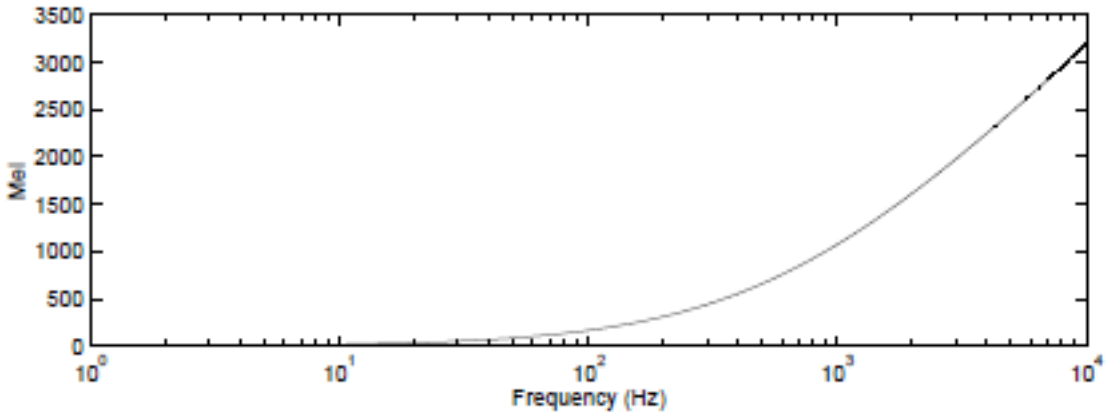

Figura 2.4: La escala Mel.

Los componentes de los vectores Mel-cepstral calculados para cada *frame* están altamente correlacionados. Por tanto, para reducir el número de parámetros en el sistema, el último paso en la construcción de las características MFCC es la de aplicar una transformada a los vectores Mel-cepstral para decorrelar sus componentes. Teóricamente, la transformada de Karhunen-Loeve (KL) consigue esto, pero en tareas del habla se suele aproximar por la transformada discreta del coseno (DCT) [17].

2.2.2. Modelos AR

Dada una serie temporal, sus medidas consecutivas contienen información sobre el proceso que lo generó. Un intento de describir el funcionamiento subyacente puede ser logrando modelar el actual valor de la variable como una suma lineal ponderada de los valores previos. Esto es un modelo autorregresivo (AR) y es una manera muy simple, aunque efectiva, de aproximar series temporales. El orden del modelo es el número de observaciones precedentes usadas para caracterizar la serie temporal.

Los modelos autorregresivos multivariables (MAR) extienden esta aproximación a series temporales múltiples, de manera que el vector de valores

actuales de varias variables es modelado como una suma lineal de actividades previas. Considérese d series temporales generadas a partir de d variables, y en donde m es el orden del modelo. Un modelo $MAR(m)$ predice el próximo valor de una serie temporal de dimensión d, $\mathbf{y}_n$ como una combinación lineal de los m vectores previos

$$\mathbf{y}_n = \sum_{i=1}^{m} \mathbf{y}_{n-1}\mathbf{A}(i) + \mathbf{e}_n \tag{2.1}$$

en donde $\mathbf{y}_n = [y_n(1), y_n(2), \ldots, y_n(d)]$ es la n-ésima muestra de la serie temporal d-dimensional, cada $\mathbf{A}(i)$ es una matriz $d \times d$ de pesos y $\mathbf{e}_n = [e_n(1), e_n(2), \ldots, e_n(d)]$ es ruido Gaussiano aditivo con media cero y covarianza R, habiendo asumido que la media de los datos ha sido sustraída de la serie temporal.

El modelo puede ser reescrito de una manera estándar como un modelo de regresión lineal multivariable de la siguiente manera

$$\mathbf{y}_n = \mathbf{x}_n\mathbf{W} + \mathbf{e}_n \tag{2.2}$$

en donde $\mathbf{x}_n = [\mathbf{y}_{n-1}\mathbf{y}_{n-2}, \ldots, \mathbf{y}_{n-m}]$ son los m vectores multivariables previos y $\mathbf{W}$ es una matriz $(m \times d) \times d$ con todos los pesos MAR. Por tanto, habrá un total de $m \times d \times d$ pesos MAR.

2.3. Modelos ocultos de Markov

2.3.1. Introducción

Un problema de interés fundamental es caracterizar las señales que producen algunos procesos en el mundo real en términos de modelos de señal. Hay varias razones por las que es interesante conseguir esta caracterización. En primer lugar, un modelo de señal puede proporcionar la base de una descripción teórica de un sistema de procesamiento de señales, algo que puede ser utilizado para procesar una señal y obtener la salida deseada. Por ejemplo, si estamos interesados en mejorar una señal corrupta por ruidos y distorsiones

presente en la transmisión de la misma, se puede usar el modelo de la señal para crear un sistema que elimine tanto el ruido como la distorsión de la transmisión. Una segunda razón es que estos modelos, potencialmente, pueden ser capaces de permitirnos conocer mejor la fuente de la señal sin tener que disponer de ésta. En último lugar, está el hecho de que estos sistemas funcionan extremadamente bien en la práctica.

Hay varias posibles elecciones para el tipo de modelo de señal a usar para caracterizar las propiedades de una cierta señal. En líneas generales, uno puede dicotomizar los tipos de modelo de señal en modelos deterministas y en estadísticos. Los modelos deterministas generalmente explotan algunas de las propiedades conocidas de la señal, requiriendo determinar los valores de los parámetros del modelo. El segundo conjunto engloba los modelos estadísticos en los cuales se intenta caracterizar exclusivamente las propiedades estadísticas de la señal. Ejemplos de éstos pueden ser: procesos gaussianos, procesos poissonianos, procesos de *Markov* o procesos ocultos de *Markov*, entre otros. La suposición subyacente en estos modelos radica en que la señal puede ser adecuadamente caracterizada como un proceso aleatorio paramétrico, y que estos parámetros pueden ser estimados de una manera precisa y bien definida.

En esta sección nos centraremos en explicar tanto teórica como matemáticamente los denominados **Modelos Ocultos de *Markov***, (HMM). Para ello, primero se repasará la teoría de las cadenas de Markov.

2.3.2. Procesos de Markov discretos

Considere un sistema que puede ser descrito en cualquier instante estando en uno del conjunto de los N estados distintos, $S_1, S_2, \ldots, S_N$, como se ilustra en la Figura 2.5. La salida del proceso en un cierto instante t es uno de los estados del modelo, correspondiendo cada estado a un cierto evento físico observable.

En instantes de tiempo discretos espaciados de manera síncrona o asíncro-

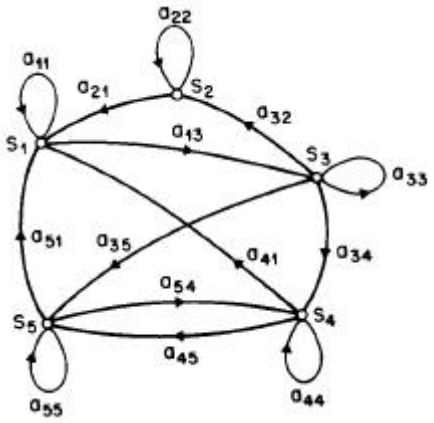

Figura 2.5: Una cadena de *Markov* con 5 estados.

na, el sistema experimenta un cambio de estado de acuerdo a un conjunto de probabilidades asociadas con el estado. Estos instantes de tiempo asociados con los cambios de estado los denotamos como $t = 1, 2, \ldots,$ y a q_t como el estado actual en el instante t. Por tanto, una descripción completa probabilística del sistema requeriría, en general, especificación tanto del estado actual como de sus estados predecesores. Para el caso de una cadena discreta de *Markov* de primer orden, esta descripción probabilística es truncada a únicamente el estado actual y el inmediatamente anterior (propiedad de Markov: el estado actual depende sólo del estado anterior). Además, se considera que estos procesos tienen unas probabilidades de transición estáticas, es decir, independientes del instante temporal en el que se encuentre el sistema.

$$a_{ij} = P[q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, \ldots] = P[q_t = S_j | q_{t-1} = S_i] \qquad 1 \leq i, j \leq N \qquad (2.3)$$

Los coeficientes de transición entre estados están sujetos a ciertas propiedades:

$$a_{ij} \geq 0 \qquad (2.4)$$

$$\sum_{j=1}^{N} a_{ij} = 1 \tag{2.5}$$

Además de estas transiciones, las cadenas de *Markov* quedan completamente caracterizadas definiendo las probabilidades de estado inicial:

$$\pi_i = P[q_1 = S_i] \qquad 1 \leq i \leq N \tag{2.6}$$

Una vez obtenidos estos parámetros que caracterizan al modelo se puede responder a distintas preguntas:

- Dado un cierto modelo, ¿cuál es la probabilidad de que ese modelo haya generado una cierta secuencia **O**, si por ejemplo esta secuencia es $\mathbf{O} = \{S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3\}$?

$$\begin{aligned} P(\mathbf{O}|Model) &= P[S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3|Model] \\ &= P[S_3] \cdot P[S_3|S_3] \cdot P[S_3|S_3] \cdot P[S_1|S_3] \\ &\quad \cdot P[S_1|S_1] \cdot P[S_3|S_1] \cdot P[S_2|S_3] \cdot P[S_3|S_2] \\ &= \pi_3 \cdot a_{33} \cdot a_{33} \cdot a_{31} \cdot a_{11} \cdot a_{13} \cdot a_{32} \cdot a_{23} \end{aligned}$$

- Dado un modelo que se encuentra en un cierto estado, ¿cuál es la probabilidad que permanezca en ese estado durante d instantes de tiempo exactamente?

$$P(\mathbf{O}|Model, q_1 = S_i) = (a_{ii})^{d-1}(1 - a_{ii})$$

2.3.3. Extensión a los HMM

Hasta ahora se han considerado modelos de *Markov* en donde cada estado se corresponde con un evento físico observable. Este modelo es demasiado restrictivo para ser aplicado a muchos problemas de interés. Por tanto, en

este punto extenderemos el concepto de modelo de *Markov* a casos donde el evento observable es una función probabilística del estado. Este modelo "extendido" es lo que se conoce como modelo oculto de Markov (HMM).

2.3.4. Elementos de un HMM

En este punto se van a definir formalmente los elementos de un HMM, y a explicar cómo los modelos generan secuencias observables.

Un HMM está caracterizado por lo siguiente:

1. El número N de estados en el modelo. Aunque los estados están ocultos, para muchas aplicaciones hay un significado físico adjunto a los estados o al conjunto de estados del modelo. Normalmente, los estados están interconectados de tal manera que cualquier estado puede ser alcanzado desde otro cualquiera.

2. M, el número de distintos elementos observables por estado, que corresponden a las salidas físicas del sistema que está siendo modelado. Se denotan los símbolos individuales como $V = \{v_1, v_2, \ldots, v_M\}$.

3. La distribución de probabilidades de transición entre estados $\mathbf{A} = \{a_{ij}\}$ donde:

$$a_{ij} = P[q_{t+1} = S_j | q_t = S_i] \qquad 1 \leq i, j \leq N \tag{2.7}$$

   En el caso especial en donde cada estado pueda alcanzar cualquier otro en un solo paso, tenemos $a_{ij} > 0$ para todo i, j. Para otros tipos de HMM, tendríamos $a_{ij} = 0$ para uno o más pares (i, j).

4. La distribución de probabilidad de símbolos observables en el estado j, $\mathbf{B} = \{b_j(k)\}$, donde:

$$\begin{aligned} b_j(k) = P[v_k \; en \; t | q_t = S_j] \qquad & 1 \leq j \leq N \\ & 1 \leq k \leq M \end{aligned} \tag{2.8}$$

5. La distribución de estados inicial $\boldsymbol{\pi} = \{\pi_i\}$ donde:

$$\pi_i = P[q_1 = S_i] \qquad 1 \leq i \leq N \tag{2.9}$$

Dados los valores apropiados de N, M, A, B, y π, el HMM puede usarse como generador para dar una secuencia de observaciones

$$\mathbf{O} = O_1 \; O_2 \; \ldots \; O_T \tag{2.10}$$

(en donde cada observación O_t es uno de los símbolos de V, y T es el número de observaciones en la secuencia) de la siguiente manera:

1. Elegir un estado inicial $q_1 = S_i$ de acuerdo a la distribución de estados inicial π.

2. Establecer $t = 1$.

3. Elegir $O_t = v_k$ de acuerdo a la distribución de probabilidad de símbolos en el estado S_i.

4. Transitar a un nuevo estado $q_{t+1} = S_j$ de acuerdo a la distribución de probabilidad de transición entre estados.

5. Establecer $t = t + 1$; volver al paso 3) si $t < T$. En caso contrario terminar el procedimiento.

El procedimiento descrito arriba puede ser usado tanto como generador de observaciones como de modelo para apreciar cómo una secuencia de observaciones fue generada por un HMM apropiado.

Según todo lo comentado anteriormente, se puede apreciar cómo para que un HMM quede completamente caracterizado se necesitan de la especificación tanto de los dos parámetros del modelo (N y M), como de los símbolos posibles de observación como de de las tres probabilidades $\mathbf{A}$, $\mathbf{B}$ y $\boldsymbol{\pi}$. Por conveniencia, se usará la notación compacta

$$\lambda = \{\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}\} \tag{2.11}$$

para indicar el conjunto de parámetros completo para el modelo.

2.3.5. Los tres problemas básicos para los HMMs

Hay tres problemas básicos de interés que deben ser resueltos para que el modelo sea útil en aplicaciones del mundo real. Como en este proyecto se ha hecho frente a estos tres problemas, a continuación se enumerarán tanto a éstos como a las soluciones a éstos. Los problemas son los siguientes:

1. Dada una secuencia de observaciones $\mathbf{O} = O_1\ O_2\ \ldots\ O_T$, y un modelo $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$, ¿cómo calcular eficientemente $P(\mathbf{O}|\lambda)$?

2. Dada una secuencia de observaciones $\mathbf{O} = O_1\ O_2\ \ldots\ O_T$, y un modelo $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$, ¿cómo se elige la secuencia de estados $\mathbf{Q} = q_1\ q_2\ \ldots\ q_T$ que mejor "explica" las observaciones?

3. ¿Cómo se ajustan los parámetros $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ del modelo para maximizar $P(\mathbf{O}|\lambda)$?

El problema 1 es el problema de evaluación, de dado un modelo y una secuencia de observaciones, ¿cómo calculamos la probabilidad de que la secuencia haya sido generada por ese modelo? También se puede ver este problema como uno que consista en "puntuar" cómo de bien un modelo se ajusta a una cierta secuencia. Este último punto de vista es extremadamente útil. Por ejemplo, si se considera el caso en el que se está intentando elegir entre varios modelos, la solución a este problema permitiría elegir el modelo que mejor se ajusta a las observaciones.

En el problema 2 se intenta encontrar la secuencia "correcta" de estados. El punto fundamental que se presenta en este punto es la presencia de diferentes criterios razonables para solucionar el problema.

En el problema 3 se intenta optimizar los parámetros del modelo para que describa de la mejor manera posible cómo una secuencia de observaciones ha sido generada. A la secuencia de observaciones usada para ajustar los parámetros del modelo se llama secuencia de entrenamiento, puesto que es usada para "entrenar" el HMM.

2.3.5.1. Solución al Problema 1

Se desea calcular la probabilidad de la secuencia de observaciones $\mathbf{O} = O_1\ O_2\ \ldots\ O_T$ dado el modelo λ. La manera más inmediata de hacer esto es enumerando todas las posibles secuencias de estados de longitud T. Si se considera una cierta secuencia de estados fijada

$$\mathbf{Q} = q_1\ q_2\ \ldots\ q_T \tag{2.12}$$

donde q_1 es el estado inicial. La probabilidad de la secuencia de observaciones O para la secuencia de estados (2.12) es

$$P(\mathbf{O}|\mathbf{Q}, \lambda) = \prod_{t=1}^{T} P(O_t|q_t, \lambda) \tag{2.13}$$

donde se asume independencia estadística de las observaciones. Por tanto, se tiene

$$P(\mathbf{O}|\mathbf{Q}, \lambda) = b_{q_1}(O_1) \cdot b_{q_2}(O_2) \ldots b_{q_t}(O_T). \tag{2.14}$$

La probabilidad de la secuencia de estados Q puede ser escrita como

$$P(\mathbf{Q}|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \ldots a_{q_{T-1} q_T}. \tag{2.15}$$

La probabilidad conjunta de $\mathbf{O}$ y $\mathbf{Q}$, es decir, la probabilidad de que $\mathbf{O}$ y $\mathbf{Q}$ ocurran simultáneamente es simplemente el producto de los términos (2.14) y (2.15)

$$P(\mathbf{O}, \mathbf{Q}|\lambda) = P(\mathbf{O}|\mathbf{Q}, \lambda) P(\mathbf{Q}|\lambda). \tag{2.16}$$

La probabilidad de $\mathbf{O}$ (dado el modelo λ) se obtiene sumando estas probabilidades conjuntas sobre todas las posibles secuencias de estado, resultando

$$P(\mathbf{O},\lambda) = \prod_{allQ} P(\mathbf{O}|\mathbf{Q},\lambda)P(\mathbf{Q}|\lambda) \tag{2.17}$$

$$= \prod_{q_1,q_2,\ldots,q_T} \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \ldots a_{q_{T-1}q_T} b_{q_T}(O_T) \tag{2.18}$$

La interpretación del cálculo de la ecuación (2.18) es la siguiente: Inicialmente (en $t = 1$) se está en el estado q_1 con probabilidad π_{q_1}, y se genera el símbolo O_1 (en este estado) con probabilidad $b_{q_1}(O_1)$. El reloj cambia y se transita desde el instante t a $t + 1$, y desde el estado q_1 al estado q_2 con probabilidad $a_{q_1 q_2}$, generando el símbolo O_2 con probabilidad $b_{q_2}(O_2)$. Este proceso continúa de la misma manera hasta que se hace la transición, en el instante T, del estado q_{T-1} al estado q_T con probabilidad $a_{q_{T-1}q_T}$ y se genera el símbolo O_T con probabilidad $b_{q_T}(O_T)$.

Explicado esto, se puede comprobar que el cálculo de (2.18) requiere del orden de $2T \cdot N^T$ cálculos. Este coste computacional es inviable, por lo que es necesario un procedimiento más eficiente. Afortunadamente tal procedimiento existe y se llama *forward-backward*.

El procedimiento *forward-backward* ([18] y [19]): Considérese la variable *forward* $\alpha_t(i)$ definida como

$$\alpha_t(i) = P(O_1\ O_2\ \ldots O_t, q_t = S_i|\lambda). \tag{2.19}$$

Se puede resolver el problema anterior mediante $\alpha_t(i)$ de manera inductiva, de la siguiente manera:

1. Inicialización:

$$\alpha_1(i) = \pi_i b_i(O_1) \qquad 1 \leq i \leq N. \tag{2.20}$$

2. Inducción:

$$\alpha_{t+1}(j) = [\sum_{i=1}^{N} \alpha_t(i)a_{ij}]b_j(O_{t+1}), \qquad \begin{array}{l} 1 \leq t \leq T-1 \\ 1 \leq j \leq N \end{array} \qquad (2.21)$$

3. Terminación:

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \qquad (2.22)$$

En primer lugar se inicializan las variables *forward*. El paso de la inducción, el cual es la parte clave del proceso, se ilustra en la Figura 2.6.

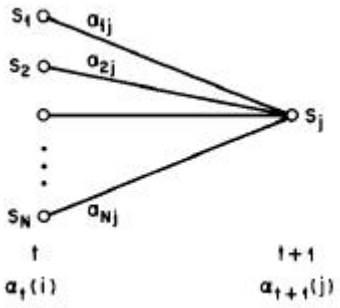

Figura 2.6: Ilustración de la secuencia de operaciones requeridas para el cálculo de la variable $\alpha_{t+1}(j)$.

Esta figura muestra cómo el estado S_j se puede alcanzar en el instante $t+1$ desde los N posibles estados, S_i, $1 \leq i \leq N$, en el instante t. Como $\alpha_t(i)$ es la probabilidad del evento conjunto de que la secuencia $O_1\ O_2\ \ldots\ O_t$ sea observada y el estado en el instante t sea S_i, el producto $\alpha_t(i)a_{ij}$ es entonces la probabilidad del evento conjunto de que la secuencia $O_1\ O_2\ \ldots\ O_t$ sea observada y el estado en el instante $t+1$ sea S_j, vía S_i en el instante t. Sumando este producto con todos los N posibles estados S_i, $1 \leq i \leq N$ en el instante t, resulta la probabilidad de S_j en el instante $t+1$ con todas las observaciones parciales previas. Una vez que se hace esto y S_j es conocido, es fácil ver que $\alpha_{t+1}(j)$ se obtiene multiplicando la cantidad sumada anteriormente por la probabilidad $b_j(O_{t+1})$. El cálculo de (2.21) se efectúa para todos los estados j, $1 \leq j \leq N$, para un instante dado t; esta misma iteración se produce para $t = 1, 2, \ldots, T-1$. Finalmente, (2.21) da el deseado

cálculo de $P(\mathbf{O}|\lambda)$ como la suma de todas las variables *forward* $\alpha_T(i)$. Esto es así ya que, por definición,

$$\alpha_T(i) = P(O_1\ O_2\ \ldots\ O_T, q_T = S_i|\lambda) \tag{2.23}$$

y por tanto $P(\mathbf{O}|\lambda)$ es la suma de todos los $\alpha_T(i)$'s.

Se puede apreciar cómo, en efecto, el cálculo de las variables *forward* está basado en una estructura de rejilla como se muestra en la Figura 2.7. La clave radica en el hecho de que sólo hay N estados en un mismo instante de tiempo, de donde salen todas las posibles secuencias de estados. En el instante $t = 1$ se necesitan calcular los valores de $\alpha_1(i)$, $1 \leq i \leq N$. En los instantes $t = 2, 3, \ldots, T$, sólo se necesitan calcular los valores de $\alpha_t(j)$, $1 \leq j \leq N$, en donde cada cálculo están "involucrados" sólo los N valores previos de $\alpha_{t-1}(i)$, debido al hecho de que cada uno de los N puntos de la rejilla en un instante de tiempo t se alcanza desde los N mismos estados del instante de tiempo anterior $t - 1$.

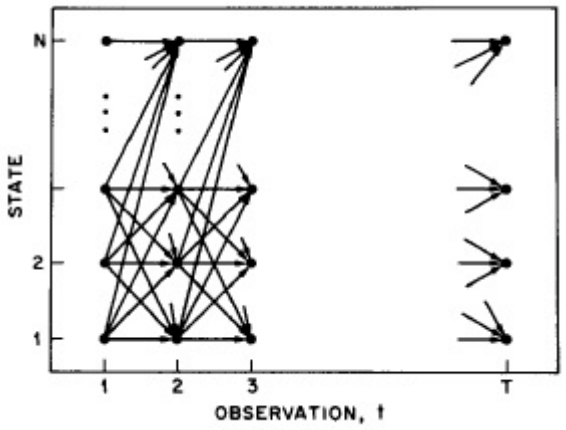

Figura 2.7: Implementación del cálculo de los $\alpha_t(i)$ en términos de una rejilla de observaciones t y estados i.

Si se examina el coste computacional de este procedimiento, se aprecia que se requiere del orden de N^2T cálculos, bastante menos que los $2TN^T$ necesarios con el otro procedimiento.

De manera similar a las variables *forward*, y aunque para resolver este

problema no son necesarias, se van a introducir en este punto a las variables *backward* $\beta_t(i)$, definidas como

$$\beta_t(i) = P(O_{t+1}\ O_{t+2}\ \ldots\ O_T|q_t = S_i, \lambda) \tag{2.24}$$

es decir, la probabilidad de la secuencia parcial de observaciones desde $t+1$ hasta el final, dado el estado S_i en el instante de tiempo t y el modelo λ.

Del mismo modo se puede solucionar el problema mediante $\beta_t(i)$ inductivamente de la siguiente manera:

1. Inicialización:

$$\beta_T(i) = 1, \qquad 1 \leq i \leq N \tag{2.25}$$

2. Inducción:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j), \qquad t = T-1, T-2, \ldots, 1 \qquad 1 \leq i \leq N \tag{2.26}$$

En el paso de inicialización se define $\beta_T(i)$ para ser 1 para todo i. En el paso 2, como se ilustra en la Figura 2.8, se realiza el proceso inverso que en el caso de las variables *forward*, en donde para estar en el estado S_i en el instante t se tienen que tener en cuenta la secuencia parcial de observaciones desde $t+1$ hasta el final, y es por ello que se tienen que tener en cuenta tanto las probabilidades de transición (a_{ij}) entre S_i a S_j, como la observación O_{t+1} en el estado j ($b_j(O_{t+1})$), como el resto de secuencia parcial desde el estado j ($\beta_{t+1}(j)$).

Como en el caso de las variables *forward*, el coste computacional de $\beta_t(i)$, para $1 \leq t \leq T$ y $1 \leq i \leq N$, requiere del orden de N^2T cálculos.

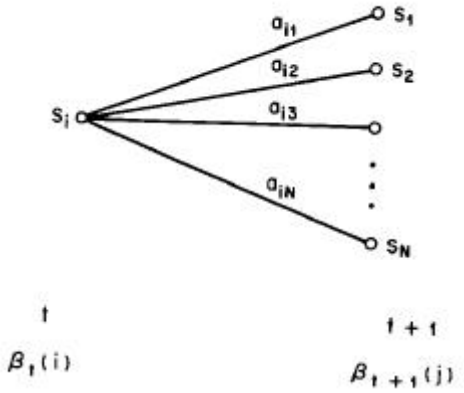

Figura 2.8: Ilustración de la secuencia de operaciones requeridas para el cálculo de la variable $\beta_t(i)$.

2.3.5.2. Solución al Problema 2

Al contrario del problema anterior, en éste no hay una solución exacta, sino que hay diversas maneras de solucionarlo. La dificultad radica en establecer qué criterio elegir. Por ejemplo, un posible criterio óptimo sería elegir los estados q_t que son, individualmente, los más probables. Este criterio maximiza el número de estados individuales correctos. Para implementar esta solución se define la variable

$$\gamma_t(i) = P(q_t = S_i|\mathbf{O}, \lambda) \tag{2.27}$$

que es la probabilidad de estar en el estado S_i en el instante t, dada una cierta secuencia de observaciones $\mathbf{O}$ y el modelo λ. La ecuación (2.27) se puede expresar en términos de variables *forward* y *backward*

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(\mathbf{O}|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \tag{2.28}$$

siendo las variables $\alpha_t(i)$ las que tienen en cuenta la secuencia parcial de observaciones $O_1,\ O_2,\ \ldots,\ O_t$ y el estado S_i en el instante t, mientras que las variables $\beta_t(i)$ son las que tienen en consideración a la secuencia de observaciones $O_{t+1},\ O_{t+2},\ \ldots,\ O_T$ dado el estado S_i en el instante t. El factor de normalización $P(\mathbf{O}|\lambda)$ hace de $\gamma_t(i)$ una medida de probabilidad que cumple

$$\sum_{i=1}^{N} \gamma_t(i) = 1 \tag{2.29}$$

Usando los $\gamma_t(i)$ podemos hallar los estados individuales más probables q_t en el instante t, como

$$q_t = \underset{1 \leq i \leq N}{\arg\,\text{máx}}[\gamma_t(i)] \qquad 1 \leq t \leq T \tag{2.30}$$

Aunque (2.30) maximiza el número esperado de estados correctos, podría haber problemas con la secuencia de estados resultantes. Es decir, esta solución no tiene en cuenta los estados anteriores seleccionados, pudiéndose, por ejemplo, producir transiciones entre estados cuya probabilidad es cero.

Una solución posible para resolver el problema anterior sería modificar el criterio. Por ejemplo, se podría buscar la secuencia de estados que maximiza el número de pares correctos de estados (q_t, q_{t+1}) o triples estados (q_t, q_{t+1}, q_{t+2}), etc. Aunque este criterio puede ser razonable para algunas aplicaciones,el criterio más usado es el de encontrar el camino de estados que maximiza $P(\mathbf{Q}|\mathbf{O}, \lambda)$, que es equivalente a maximizar $P(\mathbf{Q}, \mathbf{O}|\lambda)$. Existe una técnica formal para encontrar esto, y se llama algoritmo de *Viterbi*.

El algoritmo de *Viterbi* ([20] y [21]): para encontrar la mejor secuencia, $\mathbf{Q} = \{q_1\ q_2\ \ldots\ q_T\}$, para una cierta secuencia de observaciones $\mathbf{O} = \{O_1\ O_2\ \ldots\ O_T\}$ es necesario definir la cantidad

$$\delta_t(i) = \underset{q_1 q_2 \cdots q_{t-1}}{\text{máx}} P[q_1\ q_2\ \ldots\ q_t = i, O_1\ O_2\ \ldots\ O_t|\lambda] \tag{2.31}$$

que es la secuencia de estados más probable en el instante t teniendo en cuenta las primeras t observaciones y finalizando en el estado S_i. Por inducción se tiene

$$\delta_{t+1}(j) = [\underset{i}{\text{máx}}\ \delta_t(i)a_{ij}] \cdot b_j(O_{t+1}). \tag{2.32}$$

Por lo que para obtener la secuencia de estados hay que seguir la pista de

los argumentos, para cada t y j, que maximizan la ecuación (2.32). Esto se hace vía $\psi_t(j)$. El procedimiento completo para encontrar la secuencia óptima según este criterio tiene las siguientes etapas:

1. Inicialización:

$$\delta_1(i) = \pi_i b_i(O_1) \qquad 1 \leq i \leq N \tag{2.33}$$

$$\psi_1(i) = 0 \tag{2.34}$$

2. Recursión:

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] b_j(O_t) \qquad \begin{matrix} 2 \leq t \leq T \\ 1 \leq j \leq N \end{matrix} \tag{2.35}$$

$$\psi_t(j) = \underset{1 \leq i \leq N}{argmax} [\delta_{t-1}(i) a_{ij}] \qquad \begin{matrix} 2 \leq t \leq T \\ 1 \leq j \leq N \end{matrix} \tag{2.36}$$

3. Terminación:

$$q_T^* = \underset{1 \leq i \leq N}{argmax} [\delta_T(i)] \tag{2.37}$$

4. Secuencia de estados *backtracking*:

$$q_t^* = \psi_{t+1}(q_{t+1}^*) \qquad t = T-1, T-2, \ldots, 1 \tag{2.38}$$

Al igual que en el caso de las variables *forward* y *backward* el algoritmo de Viterbi puede ser implementado eficientemente para su computación con una estructura de rejilla.

2.3.5.3. Solución al Problema 3

El tercero, y de lejos más difícil problema, es el de determinar un método para ajustar los parámetros del modelo ($\mathbf{A}$, $\mathbf{B}$, $\boldsymbol{\pi}$) que maximice la

probabilidad de la secuencia de observaciones dado el modelo. No hay manera conocida de solucionar analíticamente el modelo que maximice una cierta secuencia de observaciones. De hecho, dada una finita secuencia de observaciones como datos de entrenamiento, no hay una manera óptima de estimar los parámetros del modelo. Sin embargo, se puede elegir ($\mathbf{A}$, $\mathbf{B}$, $\boldsymbol{\pi}$) de manera que $P(\mathbf{O}|\lambda)$ sea maximizada localmente usando un procedimiento iterativo como el método de Baum-Welch [22], o usando técnicas de gradiente [23]. En este punto nos vamos a centrar en un método iterativo basado en el trabajo de Baum-Welch, para elegir los parámetros del modelo.

Con el objetivo de describir el procedimiento de reestimación (actualización iterativa y mejora) de los parámetros del HMM, se va a definir primero la variable $\xi_t(i,j)$, que será la probabilidad de estar en el estado S_i en el instante t, y en el estado S_j en el instante $t+1$, dado un cierto modelo y una secuencia de observaciones

$$\xi_t(i,j) = P(q_t = S_i, q_{t+1} = S_j|\mathbf{O}, \lambda). \tag{2.39}$$

La secuencia de eventos que da lugar a la condición requerida (2.39) se ilustra en la Figura 2.9.

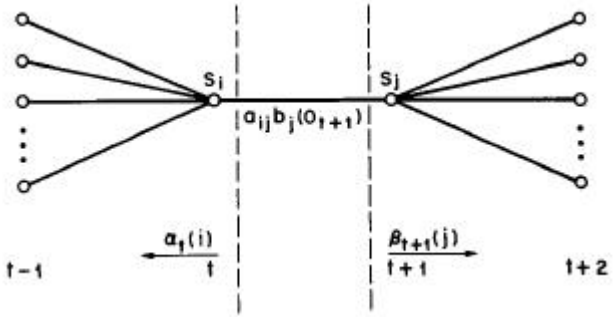

Figura 2.9: Ilustración de la secuencia de operaciones requeridas para el cálculo del evento conjunto de que el sistema esté en el estado S_i en el instante t y en el estado S_j en el instante $t+1$.

Gracias a las definiciones de las variables *forward* y *backward* se puede reescribir $\xi_t(i,j)$ de la forma

$$\begin{aligned}\xi_t(i,j) &= \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{P(\mathbf{O}|\lambda)} \\ &= \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}\end{aligned} \tag{2.40}$$

en donde el término del numerador es $P(q_t = s_i, q_{t+1} = S_j, \mathbf{O}|\lambda)$ y la división por $P(\mathbf{O}|\lambda)$ da la deseada medida de probabilidad.

Se ha definido previamente $\gamma_t(i)$ como la probabilidad de estar en el estado S_i en el instante t, dados una secuencia de observaciones y el modelo; por tanto se puede relacionar $\gamma_t(i)$ con $\xi_t(i,j)$ de la siguiente manera

$$\gamma_t(i) = \sum_{j=1}^{N}\xi_t(i,j). \tag{2.41}$$

Si se suman los $\gamma_t(i)$ en todos los instantes t de tiempo, se obtiene una cantidad que puede ser interpretada como el número esperado de veces que se visita el estado S_i, o equivalentemente, el número esperado de transiciones hechas desde el estado S_i. De manera similar, la suma de los $\xi_t(i,j)$ a lo largo de t se puede interpretar como el número esperado de transiciones desde el estado S_i al estado S_j. Esto es

$$\sum_{t=1}^{T-1}\gamma_t(i) = \textit{número esperado de transiciones desde el estado } S_i \tag{2.42}$$

$$\sum_{t=1}^{T-1}\xi_t(i,j) = \textit{número esperado de transiciones desde el estado } S_i \textit{ a } S_j \tag{2.43}$$

Usando estas fórmulas (y el concepto de contar ocurrencias de eventos) se puede dar un método de reestimación de los parámetros de un HMM. Un conjunto de razonable fórmulas de reestimación para $\boldsymbol{\pi}$, $\mathbf{A}$ y $\mathbf{B}$ son

$$\begin{aligned}\overline{\pi}_i &= frecuencia\ esperada\ de\ estar\ en\ el\ estado\ S_i\ en\ el\ instante\ (t=1) \\ &= \gamma_1(i) \end{aligned} \tag{2.44}$$

$$\begin{aligned}\overline{a}_{ij} &= \frac{número\ esperado\ de\ transiciones\ desde\ el\ estado\ S_i\ al\ estado\ S_j}{número\ esperado\ de\ transiciones\ desde\ el\ estado\ S_i} \\ &= \frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{\sum_{t=1}^{T-1}\gamma_t(i)} \end{aligned} \tag{2.45}$$

$$\begin{aligned}\overline{b}_j(k) &= \frac{número\ esperado\ de\ veces\ en\ el\ estado\ j\ y\ observando\ el\ s\acute{i}mbolo\ v_k}{número\ esperado\ de\ veces\ en\ el\ estado\ S_j} \\ &= \frac{\sum_{\substack{t=1 \\ s.t.\ O_t=v_k}}^{T-1}\xi_t(i,j)}{\sum_{t=1}^{T-1}\gamma_t(i)} \end{aligned} \tag{2.46}$$

Si se define el modelo actual como $\lambda = \mathbf{A},\ \mathbf{B},\ \boldsymbol{\pi}$, con las ecuaciones (2.44), (2.45) y (2.46), se puede reestimar el modelo como $\overline{\lambda} = (\overline{\mathbf{A}}, \overline{\mathbf{B}}, \overline{\boldsymbol{\pi}})$. Ha sido demostrado por Baum et al. ([24] y [19]) que o el modelo inicial λ define un punto crítico de la función de probabilidad, caso en el que $\overline{\lambda} = \lambda$; o sino el modelo $\overline{\lambda}$ es más probable que el modelo λ en el sentido que $P(\mathbf{O}|\overline{\lambda}) > P(\mathbf{O}|\lambda)$, es decir, se ha encontrado un nuevo modelo en donde la secuencia de observaciones es más probable que haya sido generada.

Basado en el proceso anterior, si se usa iterativamente $\overline{\lambda}$ en lugar de λ y se repite el cálculo de la estimación, entonces se puede mejorar la probabilidad de la secuencia de observaciones del modelo hasta que se alcance un cierto límite. Cabe destacar que el uso del algoritmo *forward-backward* lleva a un máximo local, y en la mayoría de los problemas de interés, la superficie de optimización es muy compleja y tiene muchos máximos locales.

Las fórmulas de reestimación (2.44), (2.45) y (2.46) pueden ser derivadas directamente maximizando (usando estándar técnicas de optimización con

restricciones) la función auxiliar de Baum:

$$Q(\lambda, \overline{\lambda}) = \sum_{Q} P(\mathbf{Q}|\mathbf{O}, \lambda)\ log[P(\mathbf{O}, \mathbf{Q}|\overline{\lambda})] \tag{2.47}$$

en todo $\overline{\lambda}$. Ha sido demotrado por Baum et al. ([24] y [19]) que la maximización de $Q(\lambda, \overline{\lambda})$ lleva al incremento de la función de probabilidad del modelo, es decir

$$\underset{\overline{\lambda}}{max}[Q(\lambda, \overline{\lambda})] = P(\mathbf{O}|\overline{\lambda}) \geq P(\mathbf{O}|\lambda). \tag{2.48}$$

Eventualmente la función de probabilidad converge a un punto crítico.

Notas sobre el procedimiento de reestimación Las fórmulas de reestimación pueden ser interpretadas como una implementación del algoritmo EM, en donde el paso E (esperanza) es el cálculo de la función auxiliar $Q(\lambda, \overline{\lambda})$, y el paso M (modificación) es la maximización en todo $\overline{\lambda}$. Por tanto las ecuaciones de reestimación del algoritmo de Baum-Welch son esencialmente idénticas a los pasos del algoritmo EM para este problema en particular.

Un aspecto importante del procedimiento de reestimación es que las restricciones estocásticas de los parámetros HMM

$$\sum_{i=1}^{N} \overline{\pi}_i = 1 \tag{2.49}$$

$$\sum_{j=1}^{N} \overline{a}_{ij} = 1, \qquad 1 \leq i \leq N \tag{2.50}$$

$$\sum_{k=1}^{M} \overline{b}_j(k) = 1, \qquad 1 \leq j \leq N \tag{2.51}$$

son automáticamente satisfechas en cada iteración. Considerando el problema de estimación de parámetros como uno de optimización con restricciones (2.49), (2.50) y (2.51) de $P(\mathbf{O}|\lambda)$, las técnicas de los multiplicadores de La-

grange pueden ser usados para encontrar los valores de π_i, a_{ij} y $b_j(k)$ que maximizan $P(\mathbf{O}|\lambda)$. En el caso de que se opte por una optimización estándar de Lagrange usando los multiplicadores de Lagrange, se puede mostrar que $P(\mathbf{O}|\lambda)$ se maximiza cuando se cumplen las siguientes condiciones:

$$\pi_i = \frac{\pi_i \frac{\partial P(\mathbf{O}|\lambda)}{\partial \pi_i}}{\sum_{k=1}^{N} \pi_k \frac{\partial P(\mathbf{O}|\lambda)}{\partial \pi_k}} \tag{2.52}$$

$$a_{ij} = \frac{a_{ij} \frac{\partial P(\mathbf{O}|\lambda)}{\partial a_{ij}}}{\sum_{k=1}^{N} a_{ik} \frac{\partial P(\mathbf{O}|\lambda)}{\partial a_{ik}}} \tag{2.53}$$

$$b_j(k) = \frac{b_j(k) \frac{\partial P(\mathbf{O}|\lambda)}{\partial b_j(k)}}{\sum_{l=1}^{M} b_j(l) \frac{\partial P(\mathbf{O}|\lambda)}{\partial b_j(l)}}. \tag{2.54}$$

Como se ha mencionado, el problema se puede resolver como uno de optimización, por lo que también las técnicas de gradiente pueden ser usadas para resolverlo. Tales procedimientos han sido usados y han mostrado que llevan a soluciones comparables a los conseguidos con el procedimiento de reestimación explicado con anterioridad.

2.3.6. Caso particular: funciones de densidad continuas en los HMM

En todo lo explicado con anterioridad, se ha considerado que las observaciones están caracterizadas por símbolos discretos elegidos de un alfabeto finito, y por tanto se puede usar una función de densidad discreta en cada estado del modelo. El problema con esta aproximación es que, al menos para algunas aplicaciones, las observaciones son señales continuas. Aunque es posible cuantificar estas señales mediante *codebooks*, etc., esto puede causar una severa degradación asociada con tal cuantificación. Por tanto sería ventajoso ser capaz de usar HMM con funciones de densidad continua.

Con el objetivo de usar funciones de densidad continuas, han de ponerse algunas restricciones en la forma de la función de densidad (pdf) para asegurar que los parámetros de la pdf pueden ser estimados de una manera

consistente. La representación más general de la pdf es una mezcla finita de la forma

$$b_j(\mathbf{O}) = \sum_{m=1}^{M} c_{jm}\Pi[\mathbf{O}, \boldsymbol{\mu}_{jm}, \mathbf{U}_{jm}] \qquad 1 \leq j \leq N \tag{2.55}$$

en donde $\mathbf{O}$ es el vector a ser modelado, c_{jm} es el coeficiente de mezcla para la m-ésima componente en el estado j y Π es cualquier densidad (por ejemplo, la *Gaussiana*) con vector de medias $\boldsymbol{\mu}_{jm}$ y matriz de covarianzas $\mathbf{U}_{jm}$ para la m-ésima componente del estado j. Normalmente se suele usar función de densidad *Gaussiana*. El factor de mezcla c_{jm} satisface las restricciones estocásticas

$$\sum_{m=1}^{M} c_{jm} = 1 \qquad 1 \leq j \leq N \tag{2.56}$$

$$c_{jm} \geq 0 \qquad 1 \leq j \leq N, 1 \leq m \leq M \tag{2.57}$$

de esta manera la fdp está adecuadamente normalizada, es decir

$$\int_{-\infty}^{\infty} b_j(\mathbf{x}) d\mathbf{x} = 1 \qquad 1 \leq j \leq N. \tag{2.58}$$

La fdp de (2.55) puede usarse para aproximar , de manera arbitrariamente certera, cualquier función de densidad finita y continua. De ahí que pueda ser aplicada en un amplio rango de problemas.

Se puede demostrar ([25] y [26]) que las fórmulas de reestimación para los coeficientes de la función de densidad, esto es, c_{jk}, $\boldsymbol{\mu}_{jk}$ y $\mathbf{U}_{jk}$ son de la forma

$$\overline{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{k=1}^{M} \gamma_t(j,k)} \tag{2.59}$$

$$\overline{\boldsymbol{\mu}}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)\mathbf{O}_t}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{2.60}$$

$$\overline{\mathbf{U}}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)(\mathbf{O}_t - \boldsymbol{\mu}_{jk})(\mathbf{O}_t - \boldsymbol{\mu}_{jk})'}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{2.61}$$

en donde el apóstrofe denota el vector traspuesto y $\gamma_t(j,k)$ la probabilidad de estar en el estado j en el instante t con la k-ésima componente de la mezcla "explicando" $\mathbf{O}_t$, esto es

$$\gamma_t(j,k) = [\frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}] = [\frac{c_{jk}\Pi[\mathbf{O}, \boldsymbol{\mu}_{jk}, \mathbf{U}_{jk}]}{\sum_{m=1}^{M} c_{jm}\Pi[\mathbf{O}, \boldsymbol{\mu}_{jm}, \mathbf{U}_{jm}]}] \tag{2.62}$$

La fórmula de reestimación para a_{ij} es idéntica a la usada para las densidades de observaciones discretas. La interpretación de (2.59), (2.60) y (2.61) es bastante sencilla. La fórmula de reestimación de c_{jk} es el cociente entre el número esperado de veces que el sistema está en el estado j usando la componente de mezcla k-ésima, y el número esperado de veces que el sistema está en el estado j. De manera similar, la fórmula de reestimación para el vector de medias $\overline{\boldsymbol{\mu}}_{jk}$ pondera el numerador de (2.59) mediante la observación, de modo que da el valor esperado de la porción del vector de observaciones "explicado" por la componente de mezcla k-ésima. Una interpretación similar se puede dar para el término de reestimación para la matriz de covarianza $\overline{\mathbf{U}}_{jk}$.

2.4. *Bag of words*

El modelo *Bag of Words* es una de las ideas más simples y útiles entre las utilizadas en el procesado de lenguaje natural. Dado un conjunto de documentos, se determina la frecuencia de aparición de cada palabra del conjunto en cada uno de los documentos. Esta información puede ser más o menos refinada en función de los filtros previos que se apliquen sobre los textos. Ver Figura 2.10.

	Términos				
	Cámara	Digital	Memoria	Impresora	...
Documento 1	3	2	0	1	
Documento 2	0	4	0	3	
...	...	...	...	...	

Figura 2.10: Tabla de frecuencia de términos para minería de textos.

La clave para extraer eficazmente la información que necesitamos de un grupo de textos está en el uso de estos filtros. Su función es eliminar información superflua, proporcionar formato para el análisis automatizado y evitar redundancia en los resultados. Los más utilizados son:

- Eliminación de *stop words*: generalmente determinantes, conjunciones, preposiciones, etc . . . Palabras muy frecuentes de bajo valor semántico.
- Un filtro simple pero básico: convertir a minúsculas todas las letras.
- *Stemming* o reducción de las palabras a su raíz, que permite contar como un mismo término a palabras independientemente de su número, género o de si se tratan de formas verbales distintas.

Además de en procesado de lenguaje natural, también se emplea en otras áreas, como reconocimiento de texturas (ver Figura 2.11) y en sistemas de recuperación de información en general.

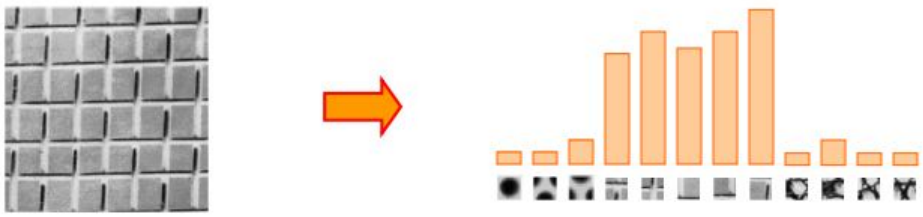

Figura 2.11: Frecuencia de las distintas texturas presente en una imagen.

De manera general las *bag of words* se pueden definir como vectores de frecuencia de aparición de elementos.

2.5. Clasificación automática

El problema de la clasificación es uno de los primeros que aparecen en la actividad científica y constituye un proceso consustancial con casi cualquier actividad humana, de tal manera que en la resolución de problema y en la toma de decisiones la primera parte de la tarea consiste precisamente en clasificar el problema o la situación, para después aplicar la metodología correspondiente y que en buena medida dependerá de esa clasificación.

Cuando hablamos de clasificar a un sujeto en un grupo determinado, a partir de los valores de una serie de parámetros medidos u observados, esa clasificación tiene un cierto grado de incertidumbre, por tanto, como es lógico, se deberá cuantificar esta incertidumbre de alguna manera, habitualmente en tanto por ciento.

2.5.1. Algunos aspectos de la teoría de aprendizaje estadístico

En el diseño de clasificadores habitualmente se utiliza el criterio de riesgo mínimo. Este criterio está basado en encontrar una función g que haga mínimo el *funcional de riesgo*. Es decir, si tenemos un problema de clasificación compuesto por l observaciones, y cada observación consiste en:

- un conjunto de atributos $\mathbf{x} \in \Re^n$, denominado vector de observación o entrada,
- una etiqueta $y \in \{1, ..., k\}$ que define la clase a la que pertenece el vector de observación y
- una función de densidad conjunta $f_{\mathbf{X},Y}(\mathbf{x}, y)$ para la totalidad de las observaciones,

encontrar la función $g : \Re^n \rightarrow \kappa$ que hace mínimo el funcional de riesgo, definido como:

$$R(g) = E_{\mathbf{X},Y}[L(g(\mathbf{x}, y))] = \int L(g(\mathbf{x}), y)\ df_{\mathbf{X},Y}(\mathbf{x}, y) \tag{2.63}$$

Siendo $L(g(\mathbf{x}), y)$ una función de pérdidas.

El problema radica en que habitualmente se desconoce $f_{\mathbf{X},Y}(\mathbf{x}, y)$. Se puede solucionar estimándola a partir del conjunto de muestras $\{\mathbf{x}_i, y_i\}\ \forall\ i$, pero esta tarea puede resultar más compleja que el propio diseño del clasificador. Otra opción sería asumir que $f_{\mathbf{X},Y}(\mathbf{x}, y)$ pertenece a una función de densidad conocida y, por tanto, sólo tendríamos que calcular los parámetros que caracterizan a dicha función. Pero corremos el riesgo de asumir una función de densidad que en realidad no se ajusta a la verdadera $f_{\mathbf{X},Y}(\mathbf{x}, y)$.

Es por ese motivo, por el cual se suele optar por estimar directamente el clasificador mediante la sustitución del funcional de riesgo por su versión muestral, conocida como riesgo empírico:

$$R_{emp}(g) = \frac{1}{l} \sum_{i=1}^{l} L(g(\mathbf{x}_i), y_i) \tag{2.64}$$

Escogiendo aquel clasificador g que minimice el riesgo empírico.

Como vemos en (2.64), el riesgo empírico es obtenido sobre un conjunto de entrenamiento de longitud l. Este error no sería el mismo si aplicamos el clasificador sobre otro conjunto de muestras generados por la misma $f_{\mathbf{X},Y}(\mathbf{x}, y)$. La cota máxima de error sobre este otro nuevo conjunto (u otro cualquiera) estaría determinada por la denominada cota de Vapnik:

$$R(g) \leq R_{emp}(g) + \sqrt{\frac{h(\log \frac{2l}{h} + 1) - \log \frac{\eta}{4}}{l}} \tag{2.65}$$

donde η es el intervalo de confianza y h es la llamada de dimensión Vapnik-Chervonenkis (dimensión VC) del clasificador g ([27] y [28]).

Por tanto, con probabilidad $1 - \eta$ (η toma valores entre 0 y 1) se cumple la cota de Vapnik (2.65).

Sobre esta cota se pueden apreciar 3 puntos a destacar:

1. Es independiente de $f_{\mathbf{X},Y}(\mathbf{x}, y)$.

2. No se puede calcular el valor exacto del lado izquierdo de la ecuación.

3. Conociendo la dimensión VC se puede hallar la cota máxima.

Además, la diferencia máxima entre el riesgo empírico y el verdadero, dependiente tanto del número de muestras como de la dimensión VC, es menor, para un cierto número fijo de muestras, cuanto menor sea h.

Visto esto, si dispusiéramos de un conjunto de máquinas de aprendizaje, y eligiendo un η suficientemente pequeño, deberíamos escoger aquella que minimice el lado derecho de la ecuación (2.65).

Información extraída de [27].

2.5.2. Máquinas de vectores soporte lineales para clasificación

Las máquinas de vectores soporte (*Support Vector Machine*, SVM) son un conjunto de métodos de aprendizaje supervisados usadas para clasificación y regresión.

Una propiedad especial de las SVM es que consiguen, simultáneamente, minimizar el error empírico de clasificación y maximizar el margen geométrico (explicado posteriormente), de ahí que también sean conocidos como clasificadores (en el caso de clasificación) de máximo margen.

2.5.2.1. Caso separable

Consideremos que tenemos el conjunto de muestras de entrenamiento $[\mathbf{x}_i, y_i]_{i=1}^N$, en donde $\mathbf{x}_i$ es el vector de entrada i-ésimo y y_i es la correspondiente salida deseada, pudiendo tomar los valores $[-1, 1]$.

En el caso separable, la clase representada por el subconjunto de muestras con salida $y_i = +1$ es separable linealmente de la clase representada por el subconjunto de muestras con salida $y_i = -1$. La ecuación del hiperplano que hace de frontera entre ambos subconjuntos sería de la forma:

$$\mathbf{w}^t\mathbf{x} + b = 0 \tag{2.66}$$

Siendo $\mathbf{x}$ el vector de entrada, $\mathbf{w}$ un vector de pesos ajustables y b el sesgo. Por tanto:

$$\begin{aligned} \mathbf{w}^t\mathbf{x} + b \geq 0 \qquad & para\ y_i = +1 \\ \mathbf{w}^t\mathbf{x} + b \leq 0 \qquad & para\ y_i = -1 \end{aligned} \tag{2.67}$$

Al ser un problema separable linealmente, las posibles soluciones, es decir, el número de posibles hiperplanos que son capaces de separar las muestras de ambas clases, son infinitas. En la Figura 2.12 se pueden apreciar 4 de las infinitas posibles soluciones.

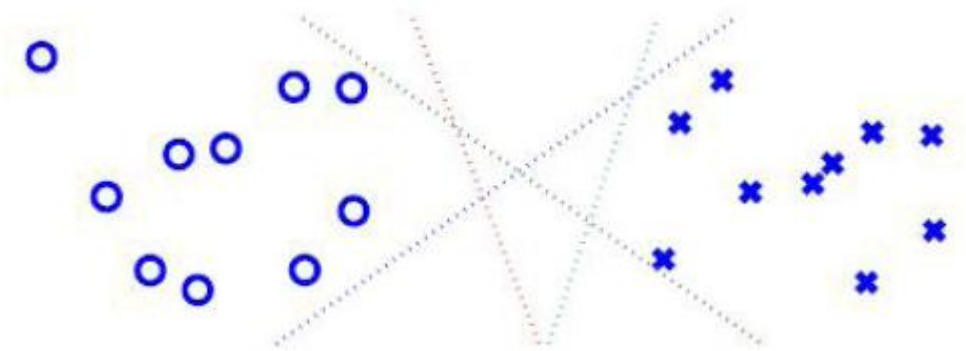

Figura 2.12: Cuatro posibles soluciones para el caso de separación lineal

Entonces, ¿qué solución elegir? Las SVM proporcionan la solución que maximiza el margen de separación entre las muestras positivas y negativas. A dicha superficie de decisión se le conoce como el hiperplano óptimo. Además esta frontera produce una tasa de error, vista anteriormente que es igual a la suma de la tasa de error del conjunto de entrenamiento (que en este caso es igual a cero) más un término que depende de la dimensión de Vapnik-Chervonenkis (2.65), en el caso separable, igual a cero (riesgo empírico) más el valor del otro término dependiente de la dimensión VC.

La Figura 2.13 nos muestra una ilustración de la idea del hiperplano óptimo para patrones separables linealmente.

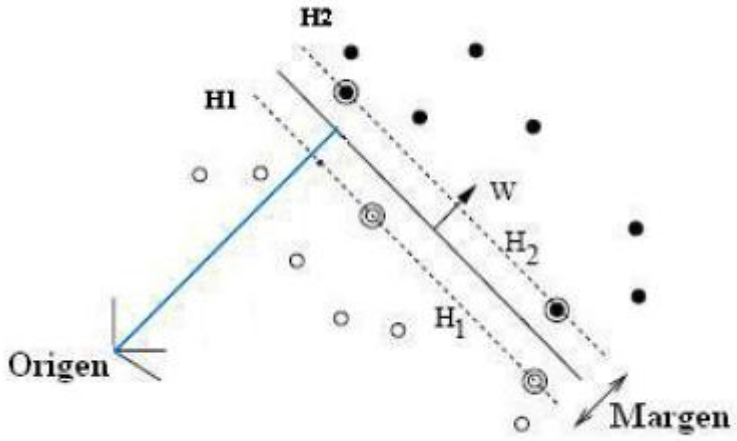

Figura 2.13: Solución de la SVM para el caso separable

Geométricamente se puede demostrar que la distancia del origen al hiperplano (en la Figura 2.13 representado por una línea azul) es igual a $-b/||\mathbf{w}||$.

Con la intención de calcular este hiperplano óptimo sujeto a las restricciones (2.67), expresamos a éstas en una sola inecuación de la siguiente manera:

$$\begin{aligned} y_i(\mathbf{w}^t\mathbf{x} + b) &\geq M \qquad \forall_i \\ y_i(\mathbf{w}^t\mathbf{x} + b) - M &\leq 0 \qquad \forall_i \end{aligned} \tag{2.68}$$

Donde M es un margen que imponemos y que podemos asumir, sin pérdida de generalidad, de valor $M = 1$.

Volviendo a la Figura 2.13, el margen es igual a la distancia entre los hiperplanos $H1$ y $H2$. O lo que es lo mismo, considerando las muestras que cumplen:

$$\begin{aligned} \mathbf{w}^t\mathbf{x}_i + b &= +1 \qquad para\ y_i = +1 \\ \mathbf{w}^t\mathbf{x}_i + b &= -1 \qquad para\ y_i = -1 \end{aligned} \tag{2.69}$$

se puede demostrar geométricamente que las muestras que cumplen la primera igualdad están a una distancia perpendicular del origen igual a $|1-b|/||\mathbf{w}||$, y las muestras que cumplen la segunda igualdad están a una distancia perpendicular del origen igual a $|-1-b|/||\mathbf{w}||$. De ahí que la distancia del hiperplano óptimo respecto a cada plano $H1$ y $H2$ sea igual a $1/||\mathbf{w}||$, y por tanto el margen sería igual a la suma de ambas distancias, $2/||\mathbf{w}||$.

Habiéndose demostrado que este margen es igual a $2/||\mathbf{w}||$, si queremos maximizar el margen para encontrar el hiperplano óptimo, entonces deberemos minimizar la norma Euclídea del vector de pesos $\mathbf{w}$, $||\mathbf{w}||$, pero siempre con la restricción (2.68).

Como se ve en la Figura 2.13, la solución se encuentra a una distancia igual de los hiperplanos $H1$ y $H2$, y éstos dependen exclusivamente de unas pocas de las muestras de entrenamiento, los denominados vectores soporte, cuya presencia es la que determinan a los planos $H1$ y $H2$ y, por tanto, al hiperplano óptimo.

Este es un problema que puede plantearse y resolverse usando el método de los multiplicadores de Lagrange.

Construyendo la función Lagrangiana:

$$J(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^t\mathbf{w} - \sum_{i=1}^{l} \alpha_i[y_i(\mathbf{w}^t\mathbf{x}_i + b) - 1] \tag{2.70}$$

El factor escalar 1/2 que multiplica a la norma Euclídea de $\mathbf{w}$ está incluida aquí sólo por conveniencia de presentación, pero no afecta de ningún modo a la solución final. Además, los multiplicadores de Lagrange, como sabemos, deben ser mayores que 0.

La solución al problema de optimización restringido está determinado por un punto de equilibro de $J(\mathbf{w}, b, \boldsymbol{\alpha})$ en donde $J(\mathbf{w}, b, \boldsymbol{\alpha})$ está minimizado respecto a $\mathbf{w}$ y b y maximizado con respecto a $\boldsymbol{\alpha}$, ya que así se demuestra que la solución obtenida es la óptima. Dicha demostración se puede encontrar en [29].

Derivando $J(\mathbf{w}, b, \alpha)$ respecto a $\mathbf{w}$ y respecto a b tenemos:

$$\begin{aligned} &Condición\ 1: && \frac{\partial J(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial \mathbf{w}} = 0 \\ &Condición\ 2: && \frac{\partial J(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial b} = 0 \end{aligned} \tag{2.71}$$

Obteniendo para cada condición:

$$\begin{aligned} \mathbf{w} &= \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i \\ 0 &= \sum_{i=1}^{l} \alpha_i y_i \end{aligned} \tag{2.72}$$

Con estas dos condiciones, más las tres ya conocidas:

$$\alpha_i \geq 0 \qquad \forall_i \tag{2.73}$$

$$y_i(\mathbf{w}^t\mathbf{x}_i + b) - 1 \leq 0 \qquad \forall_i \tag{2.74}$$

$$\alpha_i(y_i(\mathbf{w}^t\mathbf{x}_i + b) - 1) \leq 0 \qquad \forall_i \tag{2.75}$$

Tenemos las denominadas condiciones de Karush-Kuhn-Tucker (KKT) [29], condiciones necesarias y suficientes para que la solución de un problema de programación no lineal (como (2.70)) sea óptima, que nos permiten definir la solución. Nótese que se puede resolver como un sistema de ecuaciones de tres incógnitas ($\boldsymbol{\alpha}$, $\mathbf{w}$, b).

Analizando la condición (2.75), podemos apreciar cómo las muestras pueden ser agrupadas en dos subconjuntos:

1. Muestras que cumplen $(y_i(\mathbf{w}^t\mathbf{x}_i) + b) > 1$, y que por tanto, para que se cumpla (2.75) deben tener $\alpha_i = 0$.

2. Muestras que cumplen $(y_i(\mathbf{w}^t\mathbf{x}_i) + b) = 1$, y que por tanto, sus α_i no tienen por que ser obligatoriamente iguales a 0, ya que de todas maneras se cumple (2.75), por lo que sus $\alpha_i \geq 0$.

De modo que sólo las muestras pertenecientes al segundo subconjunto ($\alpha_i \geq 0$), los denominados vectores soporte, son los que determinan la solución del hiperplano óptimo $\mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i$. Pudiendo reescribirse como:

$$\mathbf{w} = \sum_{i \in SV} \alpha_i y_i \mathbf{x}_i \tag{2.76}$$

El valor del sesgo b, al igual que $\mathbf{w}$, se obtiene teniendo solamente en cuenta los vectores soporte ($\alpha_i > 0$) y cuyo valor se puede obtener despejando b en la ecuación (2.75) de KKT.

Bibliografía usada para la SVM lineal de clasificación para el caso separable: [30], [31], [32], [29], [33] y [34].

En este punto, con la función Lagrangiana y la teoría de la dualidad [31] es posible obtener la solución al problema.

En primer lugar antes de postular el problema dual a nuestro problema primal, se expandirá la expresión 2.70:

$$J(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^t\mathbf{w} - \sum_{i=1}^{l} \alpha_i y_i \mathbf{w}^t\mathbf{x}_i - b\sum_{i=1}^{l} \alpha_i y_i + \sum_{i=1}^{l} \alpha_i \tag{2.77}$$

Usando la información de las restricciones del problema primal, observamos que el tercer término es igual a 0 debido a la segunda condición de optimización ($\sum_{i=1}^{l} \alpha_i y_i = 0$). Además, recordando la primera condición de optimización ($\mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i$) del problema primal podemos comprobar cómo los dos primeros términos son idénticos entre sí e iguales a:

$$\mathbf{w}^t\mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{w}^t\mathbf{x}_i = \sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i\alpha_j y_i y_j \mathbf{x}_i^t\mathbf{x}_j \tag{2.78}$$

Reformulando la función objetivo a maximizar (recordamos que en este caso, ya que en el problema primal se buscaba minimizar, el objetivo es maximizar la expresión), puesto que ahora sólo depende de los multiplicadores de Lagrange $\boldsymbol{\alpha}$.

$$Q(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i\alpha_j y_i y_j \mathbf{x}_i^t\mathbf{x}_j - \sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i\alpha_j y_i y_j \mathbf{x}_i^t\mathbf{x}_j + \sum_{i=1}^{l} \alpha_i \tag{2.79}$$

$$= \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i\alpha_j y_i y_j \mathbf{x}_i^t\mathbf{x}_j \tag{2.80}$$

Si comparamos esta función objetivo a maximizar con la función objetivo a minimizar del problema primal (2.70), donde estaba sujeto a las 5 condiciones de KKT, observamos que ahora sólo tenemos que tener en cuenta dos restricciones:

$$\sum_{i=1}^{l} \alpha_i y_i = 0 \tag{2.81}$$

$$\alpha_i \geq 0 \tag{2.82}$$

Lo que computacionalmente es mucho más sencillo y dando lugar a la misma solución que en el caso primal.

2.5.2.2. Caso no separable

Hasta ahora nos hemos centrado en el caso linealmente separable. En este apartado consideraremos un caso más habitual, el caso de patrones no separables, en donde, dado un conjunto de entrenamiento, no nos sería posible "construir" un hiperplano de separación sin encontrar errores de clasificación. No obstante, nos gustaría encontrar un hiperplano óptimo que minimice el número de errores de clasificación en el conjunto de entrenamiento.

En este caso, al margen de separación entre clases lo denominamos suave ya que hay una o más muestras $(\mathbf{x}_i, y_i)$ que violan la siguiente condición:

$$y_i(\mathbf{w}^t\mathbf{x}_i + b) \geq +1 \qquad \forall_i \tag{2.83}$$

Esta violación se puede presentar de dos maneras diferentes:

- La muestra $(\mathbf{x}_i, y_i)$ cae en el lado correcto de la superficie de decisión, pero dentro de la superficie que se encuentra entre el hiperplano óptimo y el correspondiente $H1/H2$. Ilustrado en la Figura 2.14 (A).
- La muestra $(\mathbf{x}_i, y_i)$ cae en el lado incorrecto de la superficie de decisión como se puede ver en la Figura 2.14 (B).

Nótese que en el primer caso estamos estamos clasificando correctamente, mientras que en el segundo caso no.

Para el tratamiento de este problema, por tanto, introduciremos un nuevo conjunto de variables escalares no negativas, $\xi_{i=1}^{l}$ en la definición del hiperplano de separación:

$$y_i(\mathbf{w}^t\mathbf{x}_i + b) \geq +1 - \xi_i \qquad \forall_i \tag{2.84}$$

Los ξ_i son llamados variables "flojas" (*slack variables*), y miden la desviación del dato a la condición ideal de separabilidad. Para $0 \leq \xi_i \leq 1$, los datos se encuentran en la situación descrita en la Figura 2.14 (A). Para $\xi_i > 1$, los datos caen en el lado incorrecto del hiperplano de separación como muestra la Figura 2.14 (B). Los vectores soporte son aquellos que, al igual que en el caso separable, conforman los hiperplanos $H1$ y $H2$.

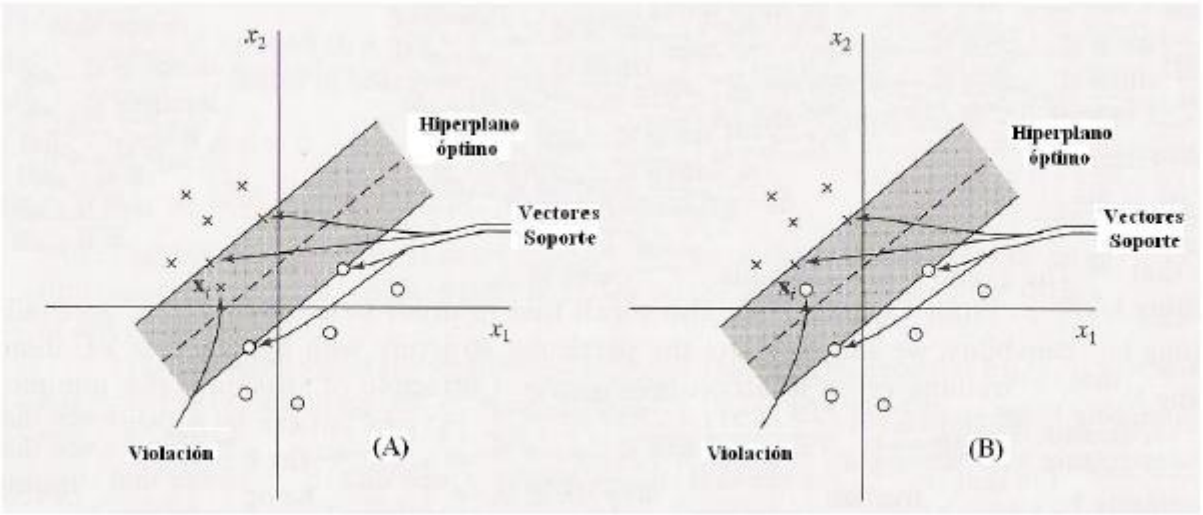

Figura 2.14: Posibles violaciones que ocurren en el caso no separable

Nuestro objetivo es encontrar un hiperplano de separación que alcance un compromiso entre la maximización del margen (minimización de $||\mathbf{w}||$) y la minimización de los errores. Por tanto, la función objetivo a minimizar será:

$$\frac{1}{2}\mathbf{w}^t\mathbf{w} + C\sum_{i=1}^{l}\xi_i \tag{2.85}$$

El parámetro C sirve para buscar un compromiso entre la complejidad de la máquina y el número de puntos no separables, por lo que puede ser

visto como una forma de regularización. Un coste alto implicará que damos prioridad a que se consiga una máquina con pocos errores en el conjunto de entrenamiento, aunque provoque que ésta sea compleja y no generalice correctamente para otro conjunto. Además, este parámetro C tiene que ser elegido por el usuario, lo que puede ser hecho en una de las dos siguientes maneras:

1. Experimentalmente, realizando una batería de pruebas sobre un conjunto de entrenamiento y su posterior validación en un conjunto de test o realizando validación cruzada.

2. Analíticamente, estimando la dimensión VC y usando límites en las prestaciones de las máquina basadas en esta dimensión de VC estimada.

Generalizando la expresión (2.85) para incluir otras funciones de pérdidas la expresamos como:

$$\frac{1}{2}\mathbf{w}^t\mathbf{w} + C\sum_{i=1}^{l} L(\xi_i) \tag{2.86}$$

siendo $L(\xi_i)$ la función de pérdidas elegida. En la Figura 2.15 se muestran distintas funciones de pérdidas. En verde la función de pérdidas aplicada (pérdidas *hinge*, $L(x, \hat{x}) = x - \hat{x}$) en (2.85), que, por comodidad, es la que se utilizará para todo el desarrollo matemático.

Cada función de pérdidas presenta una serie de características que da lugares a distintas soluciones y a máquinas de distinta complejidad.

De cualquier manera, la función objetivo a minimizar (2.85) está sujeta a las restricciones descritas en la ecuación (2.84) y a que $\xi_i \geq 0$. Construyendo la función Lagrangiana para este problema nos queda:

$$J(\mathbf{w}, \boldsymbol{\alpha}, b, \mathbf{r}) = \frac{1}{2}\mathbf{w}^t\mathbf{w} + C\sum_{i=1}^{l}\xi_i - \sum_{i=1}^{l}\alpha_i(y_i(\mathbf{w}^t\mathbf{x}_i + b) - 1 + \xi_i) - \sum_{i=1}^{l} r_i\xi_i \tag{2.87}$$

Donde $\alpha_i \geq 0$ y $r_i \geq 0$, son multiplicadores de Lagrange, y por tanto no

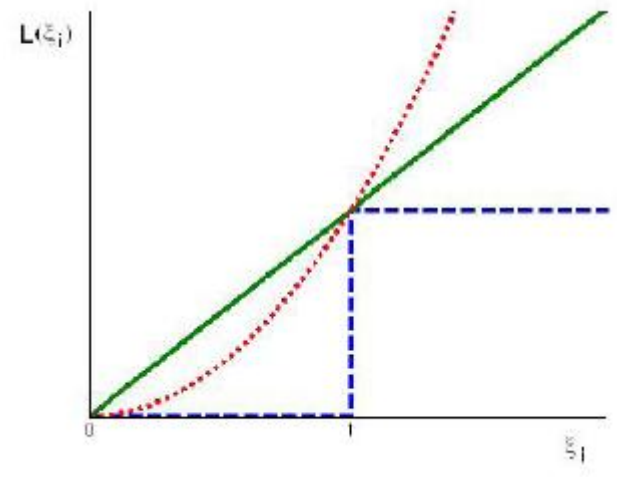

Figura 2.15: Tres funciones de pérdidas: escalón, *hinge* y cuadrática

negativos. Los r_i son introducidos para asegurar la positividad de los ξ_i.

Ya se ha planteado el problema de optimización para el caso de patrones no separables. Se puede observar que el problema de optimización para patrones separables linealmente es un caso especial, concretamente cuando los $\xi_i = 0$.

La solución, al igual que en el caso separable, queda definida por las condiciones de KKT. En este caso se obtienen encontrando al mismo tiempo el mínimo de (2.87) respecto a $\mathbf{w}$, ξ, b y el máximo respecto a todos los multiplicadores de Lagrange, es decir derivando respecto a las mencionadas variables e igualando a 0. Lo que nos queda:

$$\begin{aligned}
\frac{\partial J(\mathbf{w}, \boldsymbol{\alpha}, b, \mathbf{r})}{\partial \mathbf{w}} &= 0 \rightarrow \mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i \\
\frac{\partial J(\mathbf{w}, \boldsymbol{\alpha}, b, \mathbf{r})}{\partial \boldsymbol{\xi}} &= 0 \rightarrow 0 = \sum_{i=1}^{l} \alpha_i y_i \\
\frac{\partial J(\mathbf{w}, \boldsymbol{\alpha}, b, \mathbf{r})}{\partial b} &= 0 \rightarrow 0 = C - \alpha_i - r_i
\end{aligned} \tag{2.88}$$

Más las ya conocidas condiciones (2.89) conforman las condiciones de KKT, con lo que la solución óptima queda definida. Ya solo hay que resolver

el sistema de ecuaciones con 5 incógnitas ($\mathbf{w}$, $\boldsymbol{\alpha}$, b, $\boldsymbol{\xi}$ y $\mathbf{r}$).

$$\begin{aligned} & y_i(\mathbf{w}^t\mathbf{x}_i + b) \geq 1 - \xi_i \\ & \xi_i, \alpha_i, r_i \geq 0 \\ & \alpha_i(y_i(\mathbf{w}^t\mathbf{x}_i + b) - 1 + \xi_i) = 0 \\ & r_i\xi_i = 0 \end{aligned} \tag{2.89}$$

Al igual que en el caso separable, se puede seguir un razonamiento similar analizando algunas de las condiciones de KKT anteriores y agrupar a las muestras en 3 subconjuntos diferentes de la siguiente manera:

1. Muestras que cumplen $y_i(\mathbf{w}^t\mathbf{x}_i + b) > 1 \rightarrow \alpha_i = 0,\ \xi_i = 0,\ r_i = C$
2. Muestras que cumplen $y_i(\mathbf{w}^t\mathbf{x}_i + b) = 1 \rightarrow 0 \leq \alpha_i \leq C,\ \xi_i = 0,\ r_i = C - \alpha_i$
3. Muestras que cumplen $y_i(\mathbf{w}^t\mathbf{x}_i + b) < 1 \rightarrow \alpha_i = C,\ \xi_i = 1 - y_i(\mathbf{w}^t\mathbf{x}_i + b),\ r_i = 0$

Así que, en este caso, los vectores soporte serían aquellos pertenecientes a los dos últimos subconjuntos y determinan el hiperplano óptimo:

$$\mathbf{w} = \sum_{i \in SV} \alpha_i y_i \mathbf{x}_i \tag{2.90}$$

Procediendo en una manera similar al del caso separable, podemos desarrollar el problema dual para el caso de patrones no separables, siendo la función a maximizar:

$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^t \mathbf{x}_j \tag{2.91}$$

Sujeto a:

$$\begin{aligned} & \sum_{i=1}^{l} \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C \end{aligned} \tag{2.92}$$

Como se aprecia, el problema dual para el caso de patrones no separables es bastante similar al del caso separable excepto por una pequeña, pero importante, diferencia. La función objetivo $Q(\boldsymbol{\alpha})$ a maximizar es la misma en ambos casos, pero difiere el uno del otro en que la restricción $\alpha_i \geq 0$ es reemplazada por una restricción más exigente $0 \leq \alpha_i \leq C$.

Bibliografía utilizada para SVM lineal de clasificación para caso no separable: [30], [31], [32], [29], [33] y [34].

2.5.3. Máquinas de vectores soportes no lineales para clasificación

Para muchos problemas de clasificación, una solución lineal no da buenos resultados. En esos casos es necesaria una aproximación no lineal.

Los datos que inicialmente pertenecen al espacio $\Re^d$, donde no pueden ser separados sin error por un hiperplano, se proyectan, en primer lugar, a otro espacio (posiblemente de dimensión infinita) mediante un conjunto de transformaciones no lineales $\boldsymbol{\Psi}(\mathbf{x})$, donde ahora sí, en este nuevo espacio, los datos pueden ser separados con menos errores por un hiperplano. La Figura 2.16 muestra gráficamente el objetivo que se desea alcanzar.

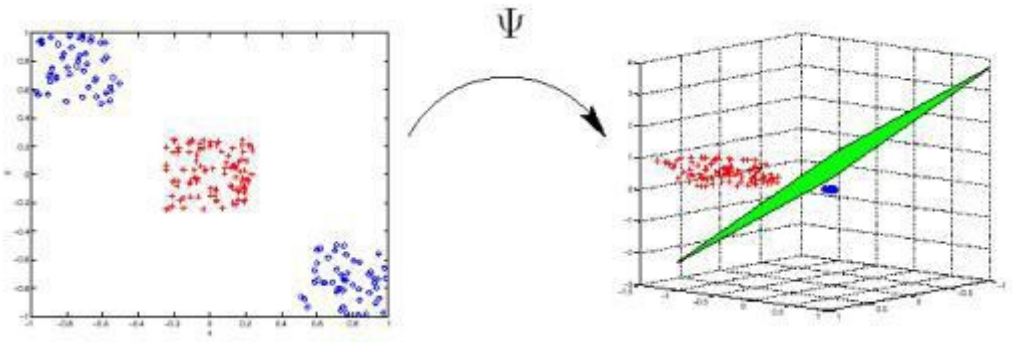

Figura 2.16: Transformación de un espacio de dimensión 2 a uno de dimensión 3.

Por tanto, el hiperplano que actúe como superficie de decisión será:

$$\sum_{j=1}^{m_1} w_j \Psi_j(\mathbf{x}) + b = 0 \tag{2.93}$$

donde los $[w]_{j=1}^{m_1}$ son el conjunto de pesos lineales, m_1 la dimensión del espacio de características y b el sesgo. Podemos simplificarlo escribiendo:

$$\sum_{j=0}^{m_1} w_j \Psi_j(\mathbf{x}) = 0 \tag{2.94}$$

asumiendo que $\Psi_0(\mathbf{x}) = 1$ y, por tanto, siendo w_0 el sesgo.

El desarrollo matemático es absolutamente idéntico a los casos anteriores, únicamente teniendo en cuenta esta transformación no lineal sobre los datos. Adaptando la expresión (2.91) a maximizar a este tipo de máquinas nos lineales obtenemos:

$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \Psi^t(\mathbf{x}_i) \Psi(\mathbf{x}_j) \tag{2.95}$$

El problema radica en que esta transformación $\boldsymbol{\Psi}$ nunca está explícita, esto es, es desconocida. Este contratiempo se soluciona mediante el denominado *truco del Kernel*, que nos permite expresar el producto de dos puntos en este espacio transformado con una función kernel que sí conocemos.

$$\begin{aligned} K(\mathbf{x}, \mathbf{x}_i) &= \boldsymbol{\Psi}^t(\mathbf{x}) \boldsymbol{\Psi}(\mathbf{x}_i) \\ &= \sum_{j=0}^{m_1} \Psi_j(\mathbf{x}) \Psi_j(\mathbf{x}_i) \end{aligned} \tag{2.96}$$

Quedando la función a maximizar:

$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{2.97}$$

Sujeto a:

$$\sum_{i=1}^{l} \alpha_i y_i = 0$$
$$0 \leq \alpha_i \leq C \tag{2.98}$$

Como vemos, la complejidad de este tipo de máquinas es bastante similar a las lineales. La única diferencia radica en que se reemplaza el producto escalar $(\mathbf{x}_i^T \mathbf{x}_j)$ en la dimensión en la que viven los datos por $K(\mathbf{x}_i, \mathbf{x}_j)$ en todas las partes del algoritmo de entrenamiento.

La máquina de vectores soporte llevará a cabo una separación lineal en este nuevo espacio, denominado espacio de Hilbert $\mathcal{H}$, el cual es dimensionalmente grande.

Para más información consultar [29] [30], [32] y [35].

Teorema de Mercer Se demuestra en [29] que estas funciones kernel, las cuales representan el producto punto en un espacio de Hilbert, son todas aquellas que cumplen la denominada condición de Mercer.

Una función es kernel

$$K(\mathbf{x}, \mathbf{y}) = \sum_i \Psi_i(\mathbf{x}) \Psi_i(\mathbf{y}) \tag{2.99}$$

si y solo si $\int f(\mathbf{x})^2 d\mathbf{x}$, entonces:

$$\int K(\mathbf{x}, \mathbf{y}) f(\mathbf{x}) f(\mathbf{y}) d\mathbf{x} d\mathbf{y} \geq 0 \tag{2.100}$$

Muchas veces el problema radica en que demostrar que se verifica (2.100) para una determinada función $f(\mathbf{x})$ no es una tarea sencilla.

Algunas de las funciones Kernels conocidas son:

1. Polinomial de grado p:

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T\mathbf{y} + 1)^p \tag{2.101}$$

2. Función base radial gaussiana (RBF):

$$K(\mathbf{x}, \mathbf{y}) = \exp^{\frac{||\mathbf{x}-\mathbf{y}||^2}{2\sigma^2}} \tag{2.102}$$

3. Red neuronal sigmoidal de dos capas:

$$K(\mathbf{x}, \mathbf{y}) = \tanh(\kappa\mathbf{x}^T\mathbf{y} - \delta) \tag{2.103}$$

4. Curva de plato delgado de grado n:

$$K(\mathbf{x}, \mathbf{y}) = ||\mathbf{x}, \mathbf{y}||^{2n} \ln ||\mathbf{x}, \mathbf{y}|| \tag{2.104}$$

Existen muchos otro tipos de kernels mucho más complejos que pueden ser estudiados en la bibliografía [29], [32], [9] y [31].

2.5.4. K-Vecinos más próximos

En reconocimiento de patrones, el algoritmo de los k vecinos más cercanos (K-NN) es un método de clasificación de objetos basado en las muestras de entrenamiento más cercanas en el espacio de características. El algoritmo $K - NN$ es un tipo de aprendizaje basado en ejemplos, resultando de los más simples dentro de los algoritmos de aprendizaje máquina: un objeto se clasifica mediante el voto mayoritario de sus vecinos, siendo asignado a la clase más común entre sus K vecinos más cercanos (K es un entero positivo, normalmente pequeño). Si $K = 1$, entonces el objeto se asigna a la clase del vecino más cercano.

Se puede usar el mismo método para regresión, simplemente asignando al objeto el valor de salida apropiado mediante el promedio de los salidas de sus K vecinos más cercanos. Puede ser útil ponderar las contribuciones de los vecinos, de manera que los vecinos más cercanos contribuyan más al valor de

salida que los más alejados del objeto. Un esquema habitual para esto último es dar a cada vecino un peso de $1/d$, en donde d es la distancia al vecino.

2.5.4.1. Algoritmo

Las muestras de entrenamiento son vectores en un espacio de características multidimensional, cada uno con una etiqueta de clase. La fase de entrenamiento del algoritmo consiste sólo en almacenar los vectores de características de las muestras de entrenamiento con sus respectivas etiquetas.

En la fase de clasificación, K es una constante definida por el usuario, y cada vector de test es clasificado asignando la etiqueta que es más frecuente entre las K muestras de entrenamiento más cercanos al vector de test.

Normalmente la distancia *Euclídea* se usa como la métrica de distancia; sin embargo esto es aplicable solamente a variables continuas. En casos como clasificación de textos, otras métricas como la distancia de *Hamming* pueden ser usadas. A menudo el acierto de clasificación del K-NN se puede mejorar significativamente si la métrica de distancia es elegida con algoritmos especializados, como *Large Margin Nearest Neighbour* o *Neighbourhood Components Analysis*.

Un inconveniente del básico "voto mayoritario" para clasificación es que las clases más frecuentes tienden a dominar la predicción del nuevo vector de test, ya que tienden a aparecer en los K vecinos más cercanos debido a su mayor presencia en el conjunto de datos. Una manera de solucionar este problema es ponderar la clasificación teniendo en cuenta la distancia desde el punto de test a cada uno de sus K vecinos más cercanos.

2.5.4.2. Selección de parámetro K

La mejor elección de K depende de los datos; generalmente, valores grandes de K reducen el efecto del ruido en clasificación, pero establece fronteras entre clases menos diferenciadas. Un buen valor de K puede ser seleccionado mediante varias técnicas heurísticas, como por ejemplo, validación cruzada.

El caso especial en donde la clase se predice de acuerdo a la clase de la muestra más cercana se llama el algoritmo del vecino más cercano.

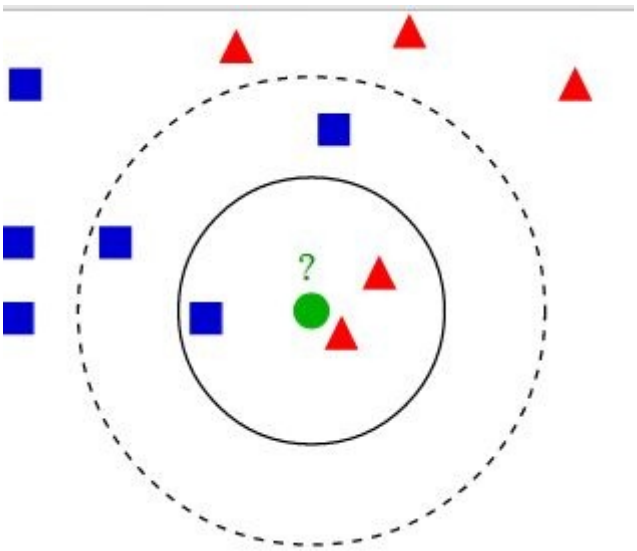

Figura 2.17: Ejemplo de clasificación K-NN. Según el valor de K la muestra de test se clasificará como triángulo rojo o como cuadrado azul.

El acierto del algoritmo K-NN se puede ver seriamente degradado por la presencia de características ruidosas o irrelevantes, o si los escalados de las características no son consistentes con su importancia. Se han realizado numerosos esfuerzos en la selección o escalado de características para mejorar la clasificación. Una aproximación particularmente popular es el de usar algoritmos evolutivos para optimizar el escalado de características [36]. Otra aproximación conocida es la de escalar características mediante la información mutua de los datos de entrenamiento con las clases de entrenamiento.

En problemas de clasificación binarios (dos clases), resulta útil elegir un número impar para el valor de K para evitar empates a votos. Una manera popular de elegir el K óptimo empíricamente en este escenario es mediante método de *bootstrap* [37].

2.5.4.3. Propiedades

La versión más simple del algoritmo es fácil de implementar mediante el cálculo de las distancias de la muestra de test al resto de vectores almacenados, pero es computacionalmente intensivo, especialmente cuando el tamaño del conjunto de entrenamiento crece. Se han propuesto muchos algoritmos de búsqueda de los vecinos más cercanos durante años; buscando, éstos, generalmente reducir el número de evaluaciones de distancia que se efectúan. Usando apropiados algoritmos de búsqueda de vecinos hacen que el algoritmo K-NN sea computacionalmente manejable incluso para conjuntos grandes de datos.

El algoritmo de los vecinos más cercano tiene resultados bastante consistentes. Cuando la cantidad de datos se aproxima a infinito, se garantiza que el algoritmo lleva a una tasa de error no peor que dos veces la tasa de error de Bayes (mínima tasa de error alcanzable conocida la distribución de los datos). En [38] se demuestra que el K-NN se aproxima a la tasa de error de Bayes para ciertos valores de K.

2.5.5. *Kernel Fisher Discriminant*

Fisher Linear Discriminant Analysis (LDA) es una técnica estadística tradicional para la reducción de la dimensión. Es una técnica ampliamente usada que ha tenido unos resultados satisfactorios para una gran cantidad de aplicaciones del mundo real. Pero, debido a su limitación de linealidad, sus prestaciones no son buenas en problemas no lineales. Para superar esta debilidad del LDA, se han propuesto durante años versiones no lineales de este algoritmo. Una de estas propuestas es el algoritmo de *kernel Fisher discriminant* (KFD), que resuelve el problema dividiéndolo en dos fases: *kernel principal component analysis* (KPCA) [39] más LDA.

Esta técnica, en este proyecto, se ha utilizado como base para un clasificador. A continuación se presenta toda la base matemática del algoritmo.

2.5.5.1. Descripción matemática del algoritmo

Sea una proyección no lineal Φ del subespacio de los datos de entrada $\Re^n$ a un espacio de características F

$$\mathbf{\Phi} : \Re^n \to F, \qquad \mathbf{x} \mapsto \Phi(\mathbf{x}) \tag{2.105}$$

Por tanto, un patrón en el espacio de entrada original $\Re^n$ es proyectado a un vector de características en un espacio de mayor conveniencia (F).

La idea del KFD es solucionar el problema del LDA en el espacio de características F. Esto se puede lograr maximizando el siguiente criterio de Fisher:

$$J^{\Phi}(\boldsymbol{\varphi}) = \frac{\boldsymbol{\varphi}^T \mathbf{S}_b^{\Phi} \boldsymbol{\varphi}}{\boldsymbol{\varphi}^T \mathbf{S}_t^{\Phi} \boldsymbol{\varphi}}, \qquad \boldsymbol{\varphi} \neq \mathbf{0} \tag{2.106}$$

en donde $\mathbf{S}_b^{\Phi}$ y $\mathbf{S}_t^{\Phi}$ son las matrices de varianza intraclase y total definidas en el espacio de características F respectivamente:

$$\mathbf{S}_b^{\Phi} = \frac{1}{M} \sum_{i=1}^{c} l_i (\mathbf{m}_i^{\Phi} - \mathbf{m}_0^{\Phi})(\mathbf{m}_i^{\Phi} - \mathbf{m}_0^{\Phi})^T \tag{2.107}$$

$$\mathbf{S}_t^{\Phi} = \frac{1}{M} \sum_{i=1}^{c} l_i (\Phi(\mathbf{x}_i) - \mathbf{m}_0^{\Phi})(\Phi(\mathbf{x}_i) - \mathbf{m}_0^{\Phi})^T \tag{2.108}$$

Aquí, $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M$ es un conjunto de M muestras de entrenamiento en el espacio de entrada; l_i es el número de muestras de entrenamiento de la clase i y satisface $\sum_{i=1}^{c} l_i = M$; $\mathbf{m}_i^{\Phi}$ es el vector de medias de las muestras de entrenamiento proyectadas de la clase i; $\mathbf{m}_0^{\Phi}$ es el vector de medias de todas las muestras de entrenamiento proyectadas.

$$\mathbf{m}_i^{\Phi} = \frac{1}{l_i} \sum_{j \in Clase\ i} \Phi(\mathbf{x}_j) \tag{2.109}$$

$$\mathbf{m}_0^{\Phi} = \frac{1}{M}\sum_{j=1}^{M}\Phi(\mathbf{x}_j) \tag{2.110}$$

Los vectores discriminadores óptimos con respecto al criterio de Fisher son los autovectores de ecuación $\mathbf{S}_b^{\Phi}\boldsymbol{\varphi} = \lambda\mathbf{S}_t^{\Phi}\boldsymbol{\varphi}$ (problema de autovalores generalizado). Como cualquiera de los autovectores se puede expresar mediante una combinación lineal de las observaciones en el espacio de características, se tiene

$$\boldsymbol{\varphi} = \sum_{j=1}^{M} a_j\Phi(\mathbf{x}_j) = \mathbf{Q}\boldsymbol{\alpha} \tag{2.111}$$

en donde $\mathbf{Q} = [\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_M)]$ y $\boldsymbol{\alpha} = (a_1, \ldots, a_M)^T$.

Substituyendo la Ecuación (2.111) en Ecuación (2.106), el criterio de Fisher se convierte en:

$$J^K(\boldsymbol{\alpha}) = \frac{\boldsymbol{\alpha}^T(\mathbf{KWK})\boldsymbol{\alpha}}{\boldsymbol{\alpha}^T(\mathbf{KK})\boldsymbol{\alpha}} \tag{2.112}$$

en donde la matriz $\mathbf{K}$ se define como

$$\mathbf{K} = \tilde{\mathbf{K}} - \mathbf{M}_1\tilde{\mathbf{K}} - \tilde{\mathbf{K}}\mathbf{1}_M + \mathbf{1}_M\tilde{\mathbf{K}}\mathbf{1}_M. \tag{2.113}$$

Aquí, $\mathbf{1}_M = (1/M)_{M\times M}$; $\tilde{\mathbf{K}} = \mathbf{Q}^T\mathbf{Q}$ es una matriz $M \times M$ cuyos elementos están determinados por

$$\tilde{\mathbf{K}}_{ij} = \Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j) = k(\mathbf{x}_i, \mathbf{y}_j) \tag{2.114}$$

donde $k(\mathbf{x}_i, \mathbf{y}_j)$ es la función kernel correspondiente a una cierta proyección no lineal Φ. Y, $\mathbf{W} = diag(\mathbf{W}_1, \ldots, \mathbf{W}_c)$, donde $\mathbf{W}_j$ es una matriz de $l_j \times l_j$ con todos los términos igual a $1/l_j$. Por tanto, $\mathbf{W}$ es una matriz diagonal en bloque de $M \times M$.

Ahora, considérese la descomposición QR de la matriz $\mathbf{K}$. Supóngase $\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_m$ son los autovectores ortonormales de $\mathbf{K}$ correspondientes a

los m (m es el rango de $\mathbf{K}$) autovectores distintos de cero $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m$. Entonces, $\mathbf{K}$ puede ser expresado como $\mathbf{K} = \mathbf{P}\Lambda\mathbf{P}^T$, en donde $\mathbf{P} = (\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_m)$ y $\Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_m)$.

Obviamente, $\mathbf{P}^T\mathbf{P} = \mathbf{I}$, en donde $\mathbf{I}$ es la matriz identidad. Substituyendo $\mathbf{K} = \mathbf{P}\Lambda\mathbf{P}^T$ en la Ecuación 2.112, se tiene

$$J^K(\alpha) = \frac{(\Lambda^{1/2}\mathbf{P}^T\boldsymbol{\alpha})^T(\Lambda^{1/2}\mathbf{P}^T\mathbf{W}\mathbf{P}\Lambda^{1/2})(\Lambda^{1/2}\mathbf{P}^T\boldsymbol{\alpha})}{(\Lambda^{1/2}\mathbf{P}^T\boldsymbol{\alpha})^T\Lambda(\Lambda^{1/2}\mathbf{P}^T\boldsymbol{\alpha})}. \tag{2.115}$$

Definiendo

$$\boldsymbol{\beta} = \Lambda^{1/2}\mathbf{P}^T\boldsymbol{\alpha} \tag{2.116}$$

Entonces, la ecuación 2.115 queda

$$J(\boldsymbol{\beta}) = \frac{\boldsymbol{\beta}^T\mathbf{S}_b\boldsymbol{\beta}}{\boldsymbol{\beta}^T\mathbf{S}_t\boldsymbol{\beta}} \tag{2.117}$$

en donde

$$\mathbf{S}_b = \Lambda^{1/2}\mathbf{P}^T\mathbf{W}\mathbf{P}\Lambda^{1/2} \text{ y } \mathbf{S}_t = \Lambda \tag{2.118}$$

Es fácil ver que $\mathbf{S}_t$ es definida positiva y $\mathbf{S}_b$ es semidefinida semipositiva. Así, la Ecuación (2.117) es el cociente de Rayleigh. Maximizando este cociente, se puede obtener un conjunto de soluciones óptimas $\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \ldots, \boldsymbol{\beta}_d$, que en realidad son los autovectores de $\mathbf{S}_t^{-1}\mathbf{S}_b$ que corresponden a los d ($d \leq c-1$) autovalores mayores.

De la Ecuación (2.116), se sabe que para un cierto $\boldsymbol{\beta}$ existe al menos un $\boldsymbol{\alpha}$ que satisface $\boldsymbol{\alpha} = \mathbf{P}\Lambda^{-1/2}\boldsymbol{\beta}$. Por tanto, después de determinar $\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \ldots, \boldsymbol{\beta}_d$, se pueden obtener un conjunto de soluciones óptimas $\boldsymbol{\alpha}_j = \mathbf{P}\Lambda^{-1/2}\boldsymbol{\beta}_j$ ($j = 1, \ldots, d$) con respecto al criterio de la Ecuación (2.112). De ese modo, los vectores discriminadores óptimos con respecto al criterio de Fisher (Ecuación (2.106)) en el espacio de características son

$$\boldsymbol{\varphi}_j = \mathbf{Q}\boldsymbol{\alpha}_j = \mathbf{QP}\Lambda^{-1/2}\boldsymbol{\beta}_j \qquad j = 1, \ldots, d \tag{2.119}$$

El problema radica en que en general no se tiene acceso a estos vectores puesto que se desconoce $\mathbf{Q}$, pero es solucionable gracias al *Kernel trick*.

2.5.5.2. Esencia de la transformación KFD: KPCA + LDA

Dada una muestra $\mathbf{x}$ y su imagen proyectada $\Phi(\mathbf{x})$, se puede obtener el vector de características discriminador $\mathbf{z}$ con la siguiente transformación:

$$\mathbf{z} = \Psi^T \Phi(\mathbf{x}) \tag{2.120}$$

en donde $\Psi = (\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2, \ldots, \boldsymbol{\varphi}_d) = (\mathbf{QP}\Lambda^{-1/2}\boldsymbol{\beta}_1, \mathbf{QP}\Lambda^{-1/2}\boldsymbol{\beta}_2, \ldots, \mathbf{QP}\Lambda^{-1/2}\boldsymbol{\beta}_d) = (\mathbf{QP}\Lambda^{-1/2})(\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \ldots, \boldsymbol{\beta}_d)$.

La transformación de arriba se puede dividir en dos partes

$$\mathbf{y} = (\mathbf{QP}\Lambda^{-1/2})^T \Phi(\mathbf{x}) \tag{2.121}$$

y

$$\mathbf{z} = \mathbf{G}^T\mathbf{y} \text{ en donde } \mathbf{G} = (\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \ldots, \boldsymbol{\beta}_d) \tag{2.122}$$

Considérese en primer lugar la Ecuación (2.121). Como $\mathbf{Q} = [\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_M)]$, $\mathbf{P} = (\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_m)$ y $\Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_m)$, la Ecuación 2.121 se puede reescribir como

$$\begin{aligned} \mathbf{y} &= (\frac{\boldsymbol{\gamma}_1}{\sqrt{\lambda_1}}, \ldots, \frac{\boldsymbol{\gamma}_m}{\sqrt{\lambda_m}})^T (\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_M))^T \Phi(\mathbf{x}) \\ &= (\frac{\boldsymbol{\gamma}_1}{\sqrt{\lambda_1}}, \ldots, \frac{\boldsymbol{\gamma}_m}{\sqrt{\lambda_m}})^T [k(\mathbf{x}_1, \mathbf{x}), \ldots, k(\mathbf{x}_M, \mathbf{x})] \end{aligned} \tag{2.123}$$

Como resumen del procedimiento descrito, las dos fases del algoritmo KFD son:

1. Paso 1: Utilizar el KPCA en el espacio de entrada $\Re^n$. En primer lugar "construyendo" la matriz $\mathbf{K}$ usando la Ecuación (2.113) y calculando los autovectores ortonormales $\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_m$ que corresponden a los m autovalores mayores distintos de cero $\lambda_1, \lambda_2, \ldots, \lambda_m$. LLevar a cabo la transformación KPCA mediante la Ecuación 2.123.

2. Paso 2: Utilizar el LDA en el espacio transformado-KPCA F. Para ello se calcula la matriz de varianza intraclases y la matriz de varianza total, $\mathbf{S}_b$ y $\mathbf{S}_t$ respectivamente, como se ha indicado en esta sección. Se calculan los autovectores $\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \ldots, \boldsymbol{\beta}_d$ de $\mathbf{S}_t^{-1}\mathbf{S}_b$ que corresponden a los d autovalores mayores. Realizar la transformación LDA mediante la Ecuación (2.122).

Para clasificar una muestra de test usando este algoritmo se realizan los siguientes pasos:

1. En primer lugar se calculan sobre el conjunto de entrenamiento las proyecciones que permitan proyectar los datos sobre el espacio de características. Posteriormente se proyectan tanto los datos de entrenamiento como los de test sobre estas proyecciones obtenidas a partir del conjunto de entrenamiento.

2. Del conjunto de entrenamiento se agrupan los vectores de características según a la clase a la que pertenecen. Posteriormente se crea, para cada clase, el centroide representativo como la media de todos los vectores característicos que pertenecen a esa clase.

3. Cada vector de características del conjunto de test se le asigna a la clase a cuyo centroide se encuentra más cercano, en términos de distancia Euclídea.

2.6. Clasificadores multiclase a partir de clasificadores binarios

El problema de la clasificación multiclase, en especial para sistemas binarios como las máquinas de vectores soporte (SVM), no presenta una solución

sencilla.

Comúnmente para solventar este problema se hace uso de dos algoritmos, los cuales, a partir de clasificadores binarios, permiten encontrar una solución para el problema de clasificación de N clases. El primero de ellos, también conocido como método estándar [7], hace uso de N diferentes clasificadores binarios. El i-ésimo clasificador binario es entrenado con todas las muestras de la clase i-ésima con etiquetas positivas, y las muestras de las otras clases con etiquetas negativas. Finalmente cada muestra del conjunto de test se pasa por la totalidad de los clasificadores, etiquetándola con la clase del clasificador cuya salida blanda ha dado el valor más grande. Habitualmente se conoce a este método como *one-against-rest* (OAR).

El otro método, propuesto por Knerr [6], construye $N \times (N-1)/2$ clasificadores, usando todos los pares de combinaciones de las N clases. A este algoritmo se le denomina *one-against-one* (OAO). Para combinar estos clasificadores Knerr propuso usar una puerta AND. Otra opción es la utilización del algoritmo Max Win, propuesta por Friedman [40], en donde la clase elegida para cada muestra del conjunto de test es la que ha obtenido más votos de entre todos los clasificadores.

El problema de estas técnicas es que el coste, considerando a éste como el número de clasificadores por el que debe pasar cada nueva muestra de test, es cuadrático en el caso de OAO y lineal en el caso de OAR, ambos respecto al número de categorías. De ahí nace la necesidad de buscar maneras de combinar clasificadores binarios que minimice el coste para cada muestra de test, sin repercutir negativamente en el porcentaje de acierto.

2.6.1. Reducción del coste de clasificación

La reducción del coste de clasificación es de especial interés en campos como, por ejemplo, en el reconocimiento de objetos, en donde lo ideal sería que el sistema pudiera reconocer la misma cantidad de categorías que un ser humano (entre 10^4 y 10^5).

La intuición nos dice que es posible escalar el coste de la clasificación sublinealmente con respecto al número de categorías que quieren ser reconocidas: cuando nosotros observamos un perro, no consideramos la posibilidad de que pueda ser clasificado como un avión o como un cono de un helado, por ejemplo. Por esta razón, es razonable pensar que, una vez se ha desarrollado una taxonomía apropiada para las categorías de nuestro problema, seremos capaces de reconocer objetos descendiendo por las ramas del árbol taxonómico y evitando, de esta manera, posibilidades remotas. Debido a ésto, los algoritmos basados en árboles aparecen como una solución a tener en cuenta.

Ejemplos de técnicas basadas en árboles de decisión que intentan reducir el coste de clasificación sublinealmente respecto al número de categorías pueden ser: [3], [4] y [5].

Como se verá en el último capítulo, en este proyecto al coger una cantidad reducida de géneros, el problema del coste computacional no será un factor limitante en el estudio. Sin embargo, servirán para comparar prestaciones, en términos de tasa de acierto, respecto a las diferentes estrategias que se propondrán en el siguiente capítulo.

Capítulo 3

Diseño del clasificador de géneros musicales

En este capítulo se propondrá, en primer lugar, la arquitectura general de clasificador de géneros musicales, y por último, algunas de las posibles alternativas que esta arquitectura nos permite llevar a cabo.

3.1. Antecedentes

Los algoritmos de clasificación multiclase presentados en el capítulo anterior (2.6) están pensados para una arquitectura de solamente dos módulos: procesado de bajo nivel y etapa de clasificación. Este esquema impide un procesamiento de más alto nivel, previo a la etapa de clasificación, para intentar capturar características tales como la dinámica temporal de la secuencia de vectores que conforman cada canción.

La intuición nos dice que la dinámica temporal de las canciones puede contener información importante para mejorar la tasa de acierto respecto a las técnicas que no aprovechan esta información. Es por ese motivo que en muchos casos es posible que la información discriminadora entre clases distintas no esté sólo en los vectores de datos, sino en la manera que ellos evolucionan a lo largo del tiempo.

Los modelos dinámicos generativos se presentan como una buena manera de capturar esta información temporal. Un ejemplo de éstos pueden ser los modelos ocultos de Markov o, de manera más general, las redes Bayesianas dinámicas (DBN) [41].

Una de las técnicas que sirvieron como fuente de inspiración fue la descrita en [8]. En ella se genera un único modelo HMM θ de K estados para todas las secuencias presentes en la base de datos. Posteriormente, el espacio-estado de θ es utilizado para representar todas las canciones. Para ello, la secuencia de vectores de cada canción $\mathbf{S}_n$ es introducido en este modelo θ y se obtiene una matriz de transición denotada como $\tilde{\mathbf{A}}^n = \{a_{ij}^n\}_{i,j=1}^K$, en donde

$$\tilde{a}_{ij}^n = p(q_{t+1}^n = s_j | q_t^n = s_i, \mathbf{S}_n, \theta). \tag{3.1}$$

Con el objetivo de obtener cada $\tilde{\mathbf{A}}^n$, se ejecuta el algoritmo *forward-backward* (2.3.5) sobre cada canción $\mathbf{S}_n$ bajo los parámetros del modelo HMM θ aprendido con anterioridad.

En este punto cada secuencia $\mathbf{S}_n$ está representada con una matriz de transición $\tilde{\mathbf{A}}^n$, pudiéndose considerar cada una de estas matrices $\tilde{\mathbf{A}}^n = [\mathbf{a}_{n1}, \ldots, \mathbf{a}_{nK}]^T$ como una colección de K funciones de probabilidad discreta $\mathbf{a}_{n1}, \ldots, \mathbf{a}_{nK}$, una por fila, correspondientes con las probabilidades de transición de cada estado a la totalidad de estados. Ahora el problema de determinar la afinidad entre secuencias puede ser transformado en uno de medición de la similitud entre distribuciones. En ese trabajo empleó la afinidad de Bhattacharyya, definida como

$$D_B(p_1, p_2) = \sum_x \sqrt{p_1(x) p_2(x)}. \tag{3.2}$$

Por tanto, la distancia entre dos secuencias $\mathbf{S}_i$ y $\mathbf{S}_j$ puede ser escrita como

$$d_{ij} = -log \frac{1}{K} \sum_{k=1}^{K} D_B(p_{ik}, p_{jk}) \tag{3.3}$$

Si se calculan todas las distancias entre todas las canciones se obtiene una matriz de distancias **D**. Por último, con la matriz de distancias **D** se pueden seguir varias estrategias (SVM, *clustering* espectral...) para clasificar los distintos fragmentos musicales.

3.2. Diseño del clasificador

3.2.1. Arquitectura del clasificador

Inspirados en este último trabajo ([8]), en este proyecto se generaliza a una arquitectura de tres módulos: uno de procesamiento a bajo nivel, otro de alto nivel y una etapa final de clasificación. Ver Figura 3.1.

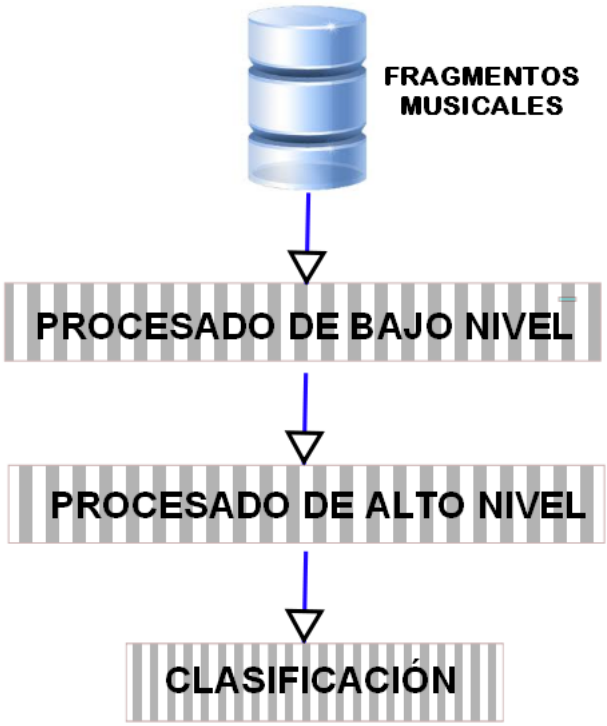

Figura 3.1: Arquitectura para un clasificador de géneros musicales.

El módulo de procesamiento a bajo nivel consistirá en la extracción de características MFCC de las canciones más la generación de modelos MAR. El segundo módulo es el encargado de realizar un procesamiento a un alto

nivel que puede, o no, capturar la dinámica temporal de las canciones. Y el último módulo será el encargado de realizar la predicción del género al que pertenece cada canción.

3.2.2. Estrategias en el procesamiento de bajo nivel

En el primer módulo de la arquitectura cada canción es preprocesada, extrayendo sus características en dos niveles temporales diferentes, de igual manera que realizan en el trabajo [8].

- **Extracción de las características breves en el tiempo:** En primer lugar, se extraen los MFCCs en ventanas que se solapan de corta duración. Estos parámetros fueron inicialmente desarrolladas para tareas de reconocimiento automático, pero también se han aplicado extensamente en tareas de Recuperación de Información Musical (*Music Information retrieval*, MIR), con buenos resultados en general. Estas características están inspiradas en la percepción auditiva de los humanos, y contienen información sobre las variaciones de la curva espectral de la señal de audio.

  En este estudio, se han usado la implementación de MFCC de [42], usando un banco con 30 filtros, y manteniendo solo los 6 coeficientes iniciales (sin embargo, el primer coeficiente, asociado a la dimensión perceptual del nivel sonoro, es descartado). El tamaño de la ventana y del salto se fijan a 30 y 15 ms, respectivamente. Por tanto, un fragmento de 60 segundos de música se representa por una matriz de tamaño 4000 $\times$ 6 después dela extracción de características MFCC.

- **Integración de las características temporales:** Es bien sabido que el uso directo de MFCC´s no proporciona una adecuada representación para tareas de de clasificación de géneros musicales. Debido a ésto, es necesario un proceso de integración en el tiempo con el objetivo de recuperar información más relevante. Como alternativa a procedimientos más simples, tales como el uso de la media y varianza de los MFCC´s, [1] propuso ajustar un modelo autoregresivo multivariante (*Multivariate Autoregressive*, MAR). Para ser más específicos, para un conjunto

de vectores MFCC´s consecutivos, se ajusta un modelo MAR de intervalos P usando la fórmula $\mathbf{z_j} = \sum_{\mathbf{p=1}}^{\mathbf{P}} \mathbf{B_p z_{j-p}} + \mathbf{e_j}$, en donde $\mathbf{z_j}$ son los MFCC´s extraídos en la ventana j-ésima, $\mathbf{e_j}$ es el error de predicción, y $\mathbf{B_p}$ son los parámetros del modelo. Los valores de las matrices $\mathbf{B_p}$, $p = 1, ..., P$, junto con la media y covarianza de los residuos $\mathbf{e_j}$ se concatenan en un vector de características (vector MAR).

En este trabajo, se han utilizado modelos MAR de orden $P = 3$, resultando vectores (ver Figura 3.2) MAR de tamaño igual a 135. Para esta fase de integración temporal, se han considerado un tamaño de ventana y de salto de 2 y 1 ms, respectivamente. Por tanto, un fragmento de audio de 60 segundos se representa con una matriz de tamaño 60×135 después de esta fase de integración temporal.

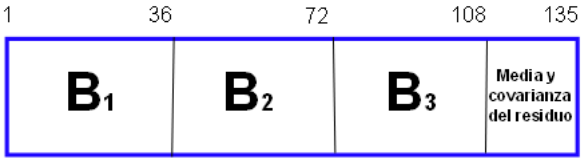

Figura 3.2: Vector MAR.

Una vez se ha fijado la caracterización de cada canción se pueden proponer las diferentes alternativas en este nivel. Para ello la pregunta que uno se puede hacer es: ¿es toda la información de los vectores MAR igual de relevante? ¿pueden provocar algunos de los coeficientes de los vectores MAR una disminución de la tasa de acierto?

Para responder a estas preguntas se han propuesto las siguientes alternativas:

1. Realizar todo el proceso restante con todos los coeficientes de los vectores MAR.

2. Realizar todo el proceso restante con los 108 primeros coeficientes de los vectores MAR.

3. Realizar todo el proceso restante con los coeficientes de los vectores MAR relacionados con la media y covarianza del residuo (del coeficiente 109 al 135).

3.2.3. Estrategias en el procesamiento de alto nivel

En este módulo se proponen distintas alternativas, la mayoría de ellas buscarán capturar la dinámica temporal de las canciones. Para ello se han analizado sistemáticamente todas las opciones basadas en HMM.

Las distintas estrategias llevadas a cabo en este nivel de la arquitectura son las siguientes:

1. La estrategia descrita con anterioridad, que se lleva a cabo en [8]. En ella se genera un único modelo HMM θ de M estados para todas las secuencias presentes en la base de datos. Posteriormente se calcula una matriz de distancias $\mathbf{D}$ entre las canciones a partir de las matrices de transición $\tilde{\mathbf{A}}^n$ de cada una de las canciones, que son obtenidas mediante la aplicación del algoritmo *forward-backward* (2.3.5) sobre cada canción bajo los parámetros del modelo HMM θ aprendido con anterioridad. Por último, esta matriz de distancias $\mathbf{D}$ pasa al módulo de la etapa de clasificación.

2. Otra alternativa consiste en la creación de un modelo HMM θ de K estados para cada uno de los géneros musicales de estudio, de manera que como paso previo se han debido agrupar todas las canciones según a la clase a la que pertenecen.

   Una vez se ha realizado este paso, se dispone de N (siendo N el número de géneros musicales presente en la base de datos) modelos ocultos de Markov. Cada uno de ellos ajustado a todas las canciones de ese género.

   En este punto, los fragmentos musicales se caracterizan con un vector de verosimilitudes, donde en cada posición del vector está la verosimilitud

de que cada modelo θ_i HMM haya generado un cierto fragmento musical C, esto es, $P(C|\theta_i)$, para $1 \leq i \leq N$.

3. Esta estrategia es una modificación de la alternativa anterior. Una vez se han entrenado los N modelos ocultos de Markov, se ha querido llegar más lejos, y se ha aspirado a crear un único modelo HMM que intente "explicar" toda la base de datos. Para ello se procede a crear un modelo en donde los estados de emisión sean los ya aprendidos en el paso anterior, de modo que el modelo dispondrá de $N \times K$ estados, y en este paso sólo se aprenderá la matriz de transiciones entre estados $\boldsymbol{A}$ y el vector de probabilidades iniciales de cada estado $\boldsymbol{\pi}$.

   La intuición nos hace pensar, y posteriormente se comprobó, que en este nuevo modelo las probabilidades de transición entre estados de géneros musicales con cierta similitud serán mayores que entre estados de géneros antagónicos. La intuición también nos hace pensar que la secuencia de vectores de un fragmento musical, por lo general, visitarán con mayor asiduidad los estados del género al que pertenece. Entonces... ¿Qué es lo que se espera de este modelo general?

   El objetivo de este modelo es aprovechar información subyacente en los distintos géneros que, a priori, desconocemos, para aprovecharla y hacer una clasificación más acertada. Esta información se podrá aprovechar como factor discriminador entre géneros. Por ejemplo, considérese que los fragmentos musicales pertenecientes al rock y al pop tienen un patrón de visita similar a estados que es esperado, ambos visitando en numerosas ocasiones estados de rock y pop; sin embargo, se puede dar el hecho de que los fragmentos musicales pertenecientes al primer género (rock) visiten con una frecuencia no despreciable estados del género *heavy metal*, mientras que los fragmentos musicales del segundo género (pop) jamás visitan estos estados. Este hecho puede ser el factor discriminador diferencial para ser aprovechado por una máquina de clasificación que permita crear fronteras de clasificación más acertadas. Además en esta estrategia introduces diversidad, algo importante desde el punto de vista del aprendizaje máquina, ya que tener proyecciones

sensatas incorrelacionadas e independientes suele ser positivo.

Como se ha podido extraer del párrafo anterior, en esta estrategia es necesario algún mecanismo para caracterizar cada fragmento musical en un vector que tenga en cuenta las "visitas" a cada estado. Para ello, en primer lugar será necesario estimar la secuencia de estados más probable vista una secuencia de observaciones, cosa que se consigue con el uso del algoritmo de Viterbi (ver Subsección 2.3.5). Posteriormente, y a semejanza del sistema de *bag of words* utilizado ampliamente en minería de textos, se ha creado lo que hemos denominado *bag of acoustic words*, de manera que cada fragmento musical está representado por un vector de $N \times K$ dimensiones, donde en la posición i-ésima del vector está el número de visitas que se ha realizado al estado i-ésimo.

Esta alternativa es la propuesta más novedosa aportada en este trabajo respecto al módulo del procesamiento de alto nivel.

4. Otra posibilidad es la de realizar la misma estrategia que en el punto anterior, con la salvedad de que el modelo HMM aprendido se hace en un único paso con todas las canciones de entrenamiento de todos los géneros.

5. Otra modificación de la propuesta principal, en donde se aprende una función de distribución gaussiana multivariable por género. Posteriormente se calcula, para todas las secuencias de vectores de cada fragmento musical, la verosimilitud de que haya sido generado por cada una de las funciones de distribución, y se asigna al género de mayor verosimilitud. Por último se crea una *bag of acoustic words*, caracterizando a cada fragmento musical con un vector que contenga las "visitas" a cada género.

   Esta alternativa se podría considerar la versión de la estrategia anterior que no aprovecha la información de la dinámica temporal.

6. No realizar ningún procesamiento de alto nivel.

En la Figura 3.3 se muestran las distintas alternativas basadas en HMM.

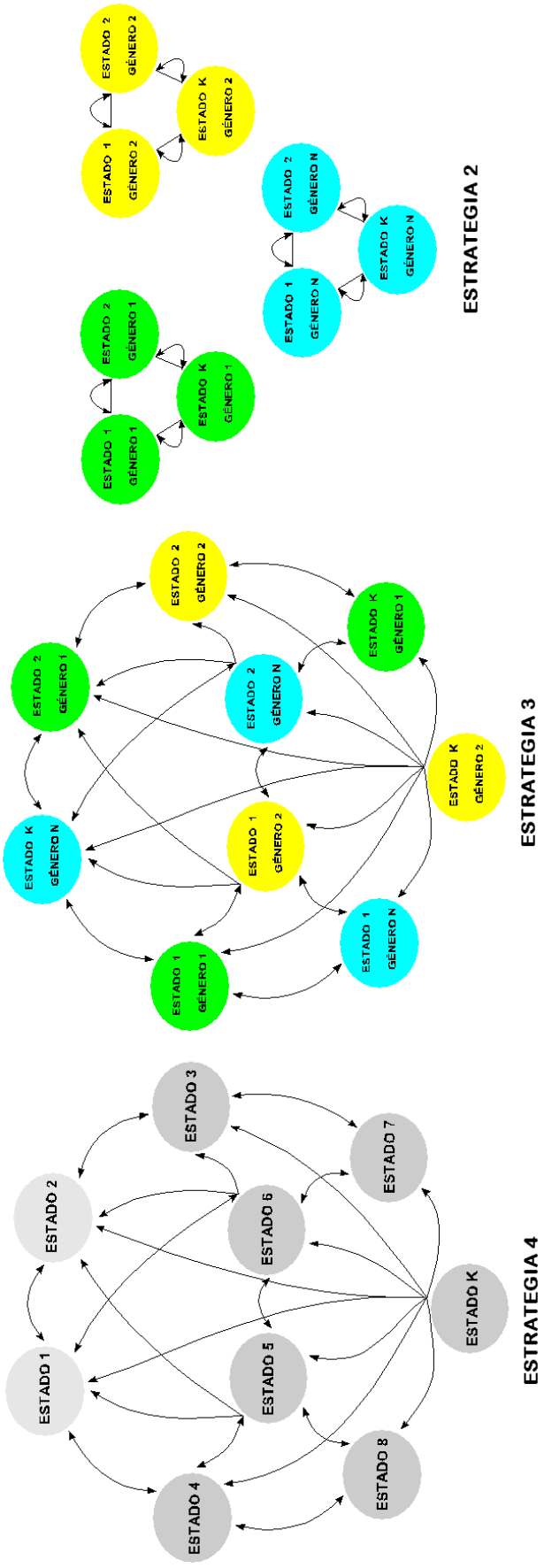

Figura 3.3: Estrategias basadas en HMM en el módulo de procesamiento de alto nivel.

3.2.4. Estrategias en la etapa de clasificación

La última fase consiste en la utilización de una técnica de clasificación. En este nivel, según la estrategia seguida en el módulo anterior de la arquitectura, la entrada, para cada canción, puede ser:

- Vector de verosimilitudes (teniendo en cuenta la dinámica temporal o no).
- La secuencia de vectores MAR.
- *Bag of acoustic words*.
- Matriz de distancias **D**.

Como clasificador se proponen varias posibilidades:

1. Máquinas de vectores soporte.
2. Árboles de decisión basados en SVMs.
3. K-Vecinos más próximos.
4. *Kernel Fisher Discriminant*.

Capítulo 4

Resultados experimentales

En este capítulo se mostrarán los resultados obtenidos con las distintas alternativas propuestas en el capítulo anterior. En base a éstos se realizarán discusiones sobre el diseño óptimo de un clasificador de géneros musicales.

Antes de eso, se presentarán la base de datos que ha sido utilizada en el estudio y la descripción de los experimentos. Para finalizar el capítulo se mostrarán las conclusiones obtenidas en este trabajo y se propondrán futuras líneas de investigación.

4.1. Base de datos

El conjunto de datos utilizados consiste en fragmentos musicales de alrededor de 60 segundos de 4412 canciones descargadas del sitio musical www.garageband.com. Las canciones están en formato MP3 y pertenecen a diferentes géneros musicales. Para este experimento se ha considerado un problema simplificado en donde el objetivo es discriminar entre cuatro diferentes géneros musicales: *Punk*, *Heavy Metal*, *Classical* y *Reggae*. De esta manera con los dos primeros géneros deberían ser, a priori, más difícil de distinguir uno del otro; mientras que los otros dos géneros musicales deberían ser más fácilmente distinguibles. De cada género se han seleccionado 300 canciones.

4.2. Descripción de los experimentos

Como régimen de entrenamiento se ha escogido uno de los dos descritos en la Subsección 2.1.2: *Repeated random sub-sampling validation*. Para ello, un experimento está formado por 10 simulaciones, donde en cada simulación se han seleccionado aleatoriamente para cada género, de las 300 disponibles para cada género, 175 canciones para entrenamiento y 25 para test. De este modo en una simulación habrá 700 muestras de entrenamiento y 100 de test.

Si, dentro de una misma simulación, se ha necesitado usar alguna técnica que requiera establecer uno o más parámetros, la manera de hallar el/los parámetro/s óptimos de esta técnica ha sido realizando *K-fold cross-validation* (Subsección 2.1.2) sobre el conjunto de entrenamiento con un barrido de posibles valores de los parámetros. Una vez estimados los valores óptimos de los parámetros, se ha procedido a realizar el entrenamiento sobre el conjunto de entrenamiento con la configuración "óptima" del algoritmo.

4.3. Validación de la propuesta

En esta sección, además de mostrar los resultados obtenidos siguiendo distintas estrategias, se han realizado diversas discusiones en los distintos niveles de la arquitectura propuesta (ver Figura 3.1).

Para ello, en un primer lugar se mostrarán los resultados de las distintas alternativas.

4.3.1. Resultados

En este apartado se estudia y discute las estrategias a seguir en cada nivel de la arquitectura, apoyadas en los resultados obtenidos en la batería de pruebas efectuadas.

A continuación una breve descripción de los experimentos:

A Uso de una SVM (OAR) directamente sobre todos los coeficientes de los vectores MAR. Cada canción es clasificada en el género que ha resultado mayoría entre los vectores de estos coeficientes.

B Algoritmo descrito en [8] (Estrategia 1 módulo procesamiento de alto nivel). Uso de los coeficientes relacionados con la media y covarianza del residuo $\mathbf{e_j}$ de los vectores MAR. Clasificador final: SVM RBF.

C Árbol de decisión descrito en [3]. Uso de todos los coeficientes MAR. Cada canción es clasificada en el género que ha resultado mayoría entre los vectores de estos coeficientes.

D Árbol de decisión descrito en [4]. Uso de todos los coeficientes MAR. Cada canción es clasificada en el género que ha resultado mayoría entre los vectores de estos coeficientes.

E Árbol de decisión descrito en [5]. Uso de todos los coeficientes MAR. Cada canción es clasificada en el género que ha resultado mayoría entre los vectores de estos coeficientes.

F Entrenamiento de un modelo HMM por género con todos los coeficientes. Cada canción está caracterizada por un vector con las verosimilitudes de que cada modelo haya generado esa secuencia de vectores MAR (Estrategia 2 módulo procesamiento de alto nivel). Será clasificada con la clase cuyo modelo haya sido más probable que haya generado la canción.

G Entrenamiento de un modelo HMM por género con los coeficientes relacionados con las matrices $\mathbf{B_1}$, $\mathbf{B_2}$ y $\mathbf{B_3}$ de los vectores MAR. Cada canción está caracterizada por un vector con las verosimilitudes de que cada modelo haya generado esa secuencia de vectores (Estrategia 2 módulo procesamiento de alto nivel). Será clasificada con la clase cuyo modelo haya sido más probable que haya generado la canción.

H Entrenamiento de un modelo HMM por género con los coeficientes relacionados con la media y covarianza del residuo $\mathbf{e_j}$ de los vectores MAR. Cada canción está caracterizada por un vector con las verosimilitudes

de que cada modelo haya generado esa secuencia de vectores (Estrategia 2 módulo procesamiento de alto nivel). Será clasificada con la clase cuyo modelo haya sido más probable que haya generado la canción.

I Entrenamiento de un único modelo HMM para todas las canciones con todos los coeficientes. Se caracterizará cada fragmento musical con un vector donde en la posición i-ésima esté el número de visitas al estado i-ésimo (*bag of acoustic words*) (Estrategia 4 módulo procesamiento de alto nivel). Posteriormente se hará uso de una SVM RBF como clasificador final.

J Entrenamiento de un único modelo HMM para todas las canciones con los coeficientes del error. Se caracterizará cada fragmento musical con un vector donde en la posición i-ésima esté el número de visitas al estado i-ésimo (*bag of acoustic words*) (Estrategia 4 módulo procesamiento de alto nivel). Posteriormente se hará uso de una SVM RBF como clasificador final.

K Modificación de la propuesta principal [1] en donde en los modelos aprendidos por género no se tiene en cuenta la dinámica temporal (Estrategia 5 módulo procesamiento de alto nivel). Clasificador final: SVM RBF. Uso de los coeficientes del error.

L Propuesta principal (Estrategia 3) del módulo de procesamiento de alto nivel con una SVM RBF de clasificador final. Uso de todos los coeficientes.

M Propuesta principal (Estrategia 3) del módulo de procesamiento de alto nivel con una SVM RBF de clasificador final. Uso de los coeficientes del error.

N Propuesta principal (Estrategia 3) del módulo de procesamiento de alto nivel con una SVM RBF de clasificador final. Uso de los coeficientes relacionados con las matrices $\mathbf{B_1}$, $\mathbf{B_2}$ y $\mathbf{B_3}$ de los vectores MAR..

[1] Hace referencia a la estrategia 3 del módulo de procesamiento de alto nivel. Ver Capítulo 3

O Propuesta principal (Estrategia 3) del módulo de procesamiento de alto nivel con una SVM lineal de clasificador final. Uso de los coeficientes del error.

P Propuesta principal (Estrategia 3) del módulo de procesamiento de alto nivel con KFD de clasificador final. Uso de los coeficientes del error.

Q Propuesta principal (Estrategia 3) del módulo de procesamiento de alto nivel con KNN de clasificador final. Uso de los coeficientes del error.

Como se aprecia, todas estas alternativas no son más que una combinación de las distintas propuestas de cada nivel expuestas en el capítulo anterior. A continuación una tabla (4.1) que resume las principales características de estas estrategias.

Características \ Algoritmo		[A]	[B]	[C]	[D]	[E]	[F]	[G]	[H]	[I]	[J]	[K]	[L]	[M]	[N]	[O]	[P]	[Q]
Bajo Nivel	Todos coefs.	✓		✓	✓	✓	✓			✓			✓					✓
	Coefs. AR							✓							✓			
	Coefs. Error		✓						✓		✓	✓		✓		✓	✓	✓
Alto Nivel	Info. Temporal		✓				✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓
	Modelo por género						✓	✓	✓			✓	✓	✓	✓	✓	✓	✓
	Modelo único/fusionado		✓							✓	✓		✓	✓	✓	✓	✓	✓
	Bag of words									✓	✓	✓	✓	✓	✓	✓	✓	✓
	Matriz de transiciones		✓						✓									
	Matriz de verosimilitudes						✓	✓										
Clasificación	SVM	✓	✓							✓	✓	✓	✓	✓	✓	✓		✓
	KNN																✓	
	KFD				✓	✓												
	Árbol decisión			✓	✓	✓												

Tabla 4.1: Características de las pruebas realizadas.

Las tasas de acierto, bajo las condiciones de entrenamiento descritas en 4.2, son las mostradas en la Tabla 4.2.

Algoritmo	**Tasa de acierto**
A	72.16 %
B	76.7 %
C	74.3 %
D	72.5 %
E	73.8 %
F	73.2 %
G	27.33 %
H	74.6 %
I	72.8 %
J	75 %
K	74 %
L	77.2 %
M	78.6 %
N	36 %
O	76.5 %
P	75.9 %
Q	77 %

Tabla 4.2: Resultados de las distintas estrategias.

4.3.1.1. Discusión en: Procesamiento de bajo nivel

La principal cuestión a responder en este módulo era la comprobación de si todos los coeficientes que conforman un vector MAR son igual de relevantes. Es decir, si la no utilización de algunos de ellos eliminaba información que, posteriormente, se traducía en una significante menor tasa de acierto del género musical al que pertenece el fragmento musical. Uno de los principales motivos por los que interesa estudiar ésto es debido a que si se es capaz de reducir la dimensionalidad del problema, sin un deterioro sensible de las prestaciones, el coste computacional también se ve reducido.

Para ello se han realizado pruebas en donde se ha fijado la estrategia a seguir en el módulo de procesamiento de alto nivel y en la etapa de clasificación, y se ha probado con distintas configuraciones en el módulo de

procesamiento de bajo nivel. Como en este trabajo se fijó la parametrización de los fragmentos musicales, las alternativas de estudio son las relacionadas con la eliminación de coeficientes de los vectores MAR, tal y como se mencionó en el capítulo anterior.

Los resultados de la Tabla 4.3 son los referentes a las pruebas relacionadas con la cuestión a responder.

Algoritmo	**Tasa de acierto**
F	73.2 %
G	27.33 %
H	74.6 %
L	77.2 %
M	78.6 %
N	36 %

Tabla 4.3: Resultados para la discusión en: Procesamiento de bajo nivel.

Si comparamos las tasas de acierto entre los experimentos [F], [G] y [H] por un lado; y por otro lado los experimentos [L], [M] y [N], vemos cómo el patrón es el mismo: el uso exclusivo de los coeficientes del error proporciona los mejores resultados, por lo que no sólo es posible reducir de manera ostensible la dimensionalidad del problema sin un empeoramiento de las prestaciones, sino que incluso mejora.

La principal conclusión extraída de estas pruebas es que los coeficientes referente a las matrices $\mathbf{B}_i$ son perjudiciales si usamos los modelos ocultos de Markov como modelos dinámicos para capturar la información temporal.

4.3.1.2. Discusión en: Procesamiento de alto nivel

En primer lugar la cuestión a responder es si una arquitectura de tres niveles puede dar lugar a mejores prestaciones que una de dos módulos. Los resultados referentes a la arquitectura de dos módulos son los de los experimentos [A], [C], [D] y [E], mientras que en el resto de pruebas se ha utilizado una arquitectura de tres niveles. A la vista de los resultados no se puede afir-

mar que los resultados proporcionados con una arquitectura de tres niveles es siempre mejor, pero sí que ésta permite llegar a tasas de acierto más altas. Por lo tanto, la primera conclusión es que siguiendo una estrategia adecuada en el módulo de procesamiento de alto nivel se pueden conseguir tasas de acierto mayores que sin la existencia de este módulo.

Dentro de las estrategias de la arquitectura de tres niveles, las mejores prestaciones son las de los experimentos que en el módulo de procesamiento de alto nivel se ha empleado la propuesta más novedosa aportada en este trabajo. Una de las ventajas que se intuían en esa estrategia era que, gracias a la diversidad aportada al aprender de manera independiente un modelo HMM por género, las relaciones subyacentes entre los distintos géneros podrían descubrirse.

4.3.1.3. Discusión en: Etapa de clasificación

En este módulo se podrían aplicar infinidad de técnicas de clasificación. La eficacia de cada una de ellos depende del problema en cuestión, por lo que no se podría afirmar la "supremacía" de un cierto algoritmo de clasificación sobre el resto. Sin embargo, con la base de datos con la que se ha trabajado, de entre los algoritmos con los que se han realizado las distintas pruebas (SVM, KNN y KFD), las máquinas de vectores soporte son las que, en idénticas condiciones en el resto de módulos, han mostrado las mejores tasas de acierto.

4.4. Conclusiones

Las directrices que se proponen a continuación son una sugerencia sobre cómo se puede diseñar un clasificador de géneros musicales. No se puede extrapolar la totalidad de las conclusiones extraídas sobre esta base de datos a cualquier otra base de datos de temática similar. Sin embargo sí sirve para demostrar que es posible conseguir una reducción de la complejidad del problema o el uso de la información temporal como factor de mejora para diseñar un clasificador de géneros musicales, por ejemplo.

Una vez se ha puntualizado ésto, las principales sugerencias dirigidas a un diseñador de este tipo de clasificadores serían las siguientes:

- Si se emplean HMMs como modelos dinámicos, utilizando solo los coeficientes del error de los vectores MAR se pueden lograr mejores prestaciones. Además, al reducir notablemente la dimensión del problema se reduce el coste computacional tanto en el entrenamiento de la máquina de clasificación como en el test.
- Una arquitectura de tres niveles para el clasificador de géneros musical permite un mayor procesado de la información, que puede traducirse a una mejor tasa de acierto.
- Relacionado con el punto anterior: los fragmentos musicales contienen información temporal que puede ser capturada para diseñar clasificadores con mejores prestaciones.

4.5. Futuras líneas de investigación

En base a los resultados obtenidos en este trabajo las líneas de investigación que proponemos son:

- Incidir en una arquitectura de clasificación de 3 módulos, buscando para el procesamiento de alto nivel nuevas alternativas que sigan aprovechando la información temporal de la parametrización de los fragmentos musicales.
- Realizar una optimización en las alternativas propuestas en este trabajo basadas en HMM sobre el número de estados y el número de componentes gaussianas de las funciones de distribución de estos estados.
- Estudiar si los coeficientes de los vectores MAR relacionados con las matrices $\mathbf{B}_i$, $1 \leq i \leq 3$, contienen en realidad información diferencial, y si es así, por qué resultan perjudiciales en el proceso de clasificación cuando se emplean modelos ocultos de Markov.
- La realización de pruebas con una mayor cantidad de géneros musicales. De esta manera es posible que, en el algoritmo de la propuesta más

novedosa, encontrar nuevas relaciones subyacentes entre géneros que sean el factor discriminador diferencial parar mejorar la tasa de acierto.

- Comprobar las prestaciones de otras extracciones de características sobre la base de datos.
- Utilizar filtro de Kalman en lugar de modelos ocultos de Markov.
- Medir la complejidad de las distintas estrategias.
- Analizar del contenido de las *bags of words* resultantes.
- Realizar pruebas sobre otras base de datos.
- Extender las propuestas del trabajo a base de datos de escritura, páginas web...

Anexo 1

Presupuesto

A continuación se detalla el cálculo de los costes aproximados de realización del proyecto descrito en la presente memoria. El presupuesto está dividido en dos partes, por un lado los costes asociados a materiales empleados y por otro, los costes debidos a honorarios de las personas que han participado en el proyecto. Todo ello se une en un resumen en el que se contabilizan los gastos totales acumulados durante el periodo de desarrollo del proyecto.

1.1. Coste del material

El material empleado en la realización del proyecto consta de los siguientes elementos:

- **Equipo informático**: Se ha hecho uso de un ordenador personal cuyo valor aproximado es de 800€. Aunque el equipo se ha empleado exclusivamente en este proyecto, tras la finalización del mismo se aprovechará en otras labores, por lo que en concepto de amortización del material sólo se considerará el 25 % de su precio, es decir, 200€.

- **Lugar de trabajo**, con las debidas condiciones de luz, calefacción, mantenimiento, más el mobiliario necesario; tiene un coste asociado de unos 900€/mes. Al tratarse de un espacio compartido por 9 personas,

tendrá un coste individual asociado de 100€/mes. Puesto que el proyecto ha durado aproximadamente 8 meses, el coste asociado asciende a 800€.

- **Material de oficina**: Dentro de este concepto se incluye todo el material desechable empleado en el proyecto, impresión de artículos, hojas, carpetas, etc. El total estimado es de 60€.
- **Coste de licencias software**: Para la realización de este proyecto se ha empleado el sistema operativo Windows XP Professional de Microsoft, con un coste por licencia de 220€, a amortizar en 4 años, por lo que en 8 meses sólo se consideran 37€aplicables al proyecto. Para el desarrollo del software se ha empleado el programa Matlab de Mathworks, con un coste de licencia de 1950 €, a amortizar en 4 años, por lo que el coste a cargar en el proyecto es de 325€.
- **Conexión a Internet**: La tarifa plana ADSL tiene un coste mensual de 35€. A lo largo de 8 meses, el coste será de 280€.

En la siguiente tabla (1.1) están resumidos los costes relacionados con el material.

Concepto	**Coste**	**Cantidad**	**Total**
Equipo informático	200€	1 unidad	200€
Lugar de trabajo	100€/mes	8 meses	800€
Material de oficina	60€	-	60€
Costes de licencias	362€	1 unidad	362€
Conexión a Internet	35€/mes	8 meses	280 €
Total			1702 €

Tabla 1.1: Coste de los materiales.

1.2. Coste de honorarios

La duración total del proyecto ha sido de ocho meses. De forma general se puede establecer un horario de trabajo de cuatro horas diarias. Teniendo en cuenta la semana laboral, de lunes a viernes, obtendríamos 20 horas semanales o 80 horas mensuales. En total el proyecto se traduciría en 640 horas

de trabajo. En ese periodo se incluye el desarrollo de la memoria del proyecto.

Hoy en día, el Ministerio de Economía y Hacienda ha remitido a todos los colegios profesionales una nota en la que se indica que no se debe, ni siquiera, publicar un baremo con los honorarios ya que, éstos son libres y responden al libre acuerdo entre el profesional y el cliente. Dada la situación, los honorarios deben definirse en función de una serie de factores: costes del ingeniero, desplazamientos, mecanografía, volumen de la actividad, etc.

Hasta hace poco, la Junta General del Colegio Oficial de Ingenieros Técnicos de Telecomunicación ofrecía una cantidad a modo de ejemplo para los libres ejercientes de la profesión, de forma que pudieran disponer de una referencia de los honorarios. Dicha cantidad definía unos honorarios de 65 €/hora. En mi caso, al tratarse de un Ingeniero de Telecomunicación estimaré el honorario en 75 €/hora. El salario del director de proyecto se estima, de forma general, como un 7 % del coste total del proyecto. Dado que el coste, sin contar el salario del director, es de 49702 €, los honorarios de dirección ascienden a 3479.14 €.

A continuación se muestra la tabla (1.2) que detalla el presupuesto de los gastos personales con los datos descritos anteriormente:

Concepto	**Coste**	**Cantidad**	**Total**
Ingeniero encargado del proyecto	75€/hora	640 horas	48000€
Director del proyecto	7 %	49702 €	3479.14€
		TOTAL (sin IVA)	51479.14€
		IVA 16 %	8236.66€
		TOTAL	59715.8 €

Tabla 1.2: Coste de los honorarios.

1.3. Presupuesto total

En total, teniendo en cuenta los gastos materiales y los gastos personales, obtenemos el presupuesto final (Tabla 1.3):

Concepto	**Coste**
Coste del material	1702 €
Coste de honorarios	59715.8 €
Total	**61417,8e**

Tabla 1.3: Coste total del proyecto.

El presupuesto total de este proyecto asciende a SESENTA Y UN MIL CUATROCIENTOS DIECISIETE CON OCHO CÉNTIMOS.

Fdo: Alberto García Durán

Ingeniero de Telecomunicación (estudiante)

Bibliografía

[1] Anders Meng, Peter Ahrendt, Jan Larsen, and Lars K. Hansen. Temporal feature integration for music genre classification. *IEEE Trans. Audio Speech and Lang. Process., vol.15, pp. 1654-1664*, 2007.

[2] M. F. McKinney and J. Breebart. *Features for audio and music classification*. Proc. Intl. Symp. Music Information Retrieval (ISMIR), 2003.

[3] Gregory Griffin and Pietro Perona. Learning and using taxonomies for fast visual categorization. *CVPR*, 2008.

[4] John C. Platt, Nello Cristianini, and John Shawe-Taylor. Large margin dags for multiclass classification. *MIT Press*, 2000.

[5] Volkan Vural and Jennifer G. Dy. A hierarchical method for multi-class support vector machines. *ACM 2004*, 2004.

[6] S. Knerr, L. Personnaz, and G. Dreyfus. *A stepwise procedure for building and training a neural network*. Springer, 1990.

[7] V. Vapnik. *The nature of statiscal learning theory*. New York: Springer, 1995.

[8] Darío García-García, Emilio Parrado-Hernández, Jerónimo Arenas-García, and Fernando Díaz de María. Music genre classification using the temporal structure songs. *MLSP*, 2010.

[9] Ralf Herbrich. *Learning kernel classifiers : theory and algorithms*. MIT Press, 2002.

[10] Nils J. Nilsson. *Introduction to Machine Learning*. Standford University, 1996.

[11] Richard S. Sutton. *Reinforcement learning*. Kluwer Academic, 1992.

[12] Igor Kononenko. *Machine learning and data mining : introduction to principles and algorithms*. Horwood Publishing, 2007.

[13] Igor Kononenko. *Introduction to Machine Learning*. The MIT Press, 2004.

[14] J. Am. J. Shao. *Linear model selection by cross-validation*. 1993.

[15] L. R. Rabiner and B. H. Juang. *Fundamentals of Speech Recognition*. 2003.

[16] Beth Logan. Mel frequency cepstral coefficients for music modeling. In *In International Symposium on Music Information Retrieval*, 2000.

[17] N. Marhav and C. H. Lee. On the asymptotic statistical behavior of empirical cepstral coefficients. *IEEE Trans. on Signal Processing, vol. 41, pp. 1990-1993*, 1993.

[18] L. E. Baum and J. A. Egon. An inequality with application to statistical estimation for probabilistic functions of a markov process and to a model for ecologyy. *Bull. Amer. Meteorol. Soc. 73 (360-363)*, 1967.

[19] L. E. Baum and G. R. Sell. Growth functions for transformations on manifolds. *Pac. J. Math 27 (211-227)*, 1968.

[20] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *Information Theory vol.13 (260-269)*, 1967.

[21] G. D. Forney. The viterbi algorithm. *Proc. IEEE vol.61 (268-278)*, 1973.

[22] A. P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *J. Roy. Stat. Soc. vol. 39 (1-38)*, 1977.

[23] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi. An introduction to the application of theory of probabilistic functions of a markov process to automatic speech recognition vol. 32 (307-309). *Bell. Syst. Tech.*, 1983.

[24] J. K. Baker. The dragon system - an overview. *IEEE Trans. Acoust. Speech Signal Processing vol.23 (24-49)*, 1975.

[25] L. A. Liporace. Maximum likelihood estimation for multivariate observations of markov sources. *IEEE Trans. Information Theory vol.28 (729-734)*, 1982.

[26] B. H. Juang, S. E. Levinson, and M. M. Sondhi. Maximum likelihood estimation for multivariate mixture observations of markov chains. *IEEE Trans. Information Theory*, 1986.

[27] Antonio Artés-Rodriguez, José Miguel Leiva-Murillo, Mario de Prado-Cumplido, Ricardo Santiago-Mozos, Fernando Pérez-Cruz, Ángel Navia-Vázquez, and Aníbal R. Figueiras-Vidal. Estado del arte y líneas de investigación abiertas en máquinas de vectores soporte.

[28] Vladimir Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1989.

[29] Bernhard Schölkopf and Alex Smola. Learning with kernels. *The MIT Press*, 2002.

[30] Simon Haykin. *Neural Networks*. Prentice-Hall, 1999.

[31] John Shawe-Taylor and Nello Cristianini. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

[32] Andrew Webb. *Statiscal Pattern Recognition*. Wiley, 2002.

[33] Chih-Jen Lin, Chich-Wei Hsu, and Chih-Chung Chang. *A practical guide to support vector classification*, 2007.

[34] Christopher J.C. Burges. *A tutorial on support vector machines for pattern recognition*, 1998.

[35] Robert J. Howlett. *Radial basis function networks*. Physica, 2001.

[36] Nigsch F., Bender A., van Buuren B., Tissen J., Nigsch E., and Mitchell JB. Melting point prediction employing k-nearest neighbor algorithms and genetic parameter optimization. *Journal of Chemical Information and Modeling*, 2006.

[37] Peter Hall, Byeong U. Park, and Richard J. Samworth. Choice of neighbor order in nearest-neighbor classification. *Annals of Statistics vol. 36 (2135-2152)*, 2008.

[38] T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory vol. 13 (21-27)*, 1967.

[39] B. Schölkopf, A. Smola, and K.R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput. vol. 10 (1299-1319)*, 1998.

[40] J. H. Friedman. Another approach to polychotomous classification. *Technical Report, Stanford University*, 1996.

[41] K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning.* PhD thesis, UC Berkeley, 2002.

[42] Sigurdur Sigurdsson, Kaare Brandt Petersen, and Tue Lehn-Schioler. Mel frequency cepstral coefficientes: An evaluation of robustness of mp3 encoded music. *Pro- Intl. Symp. Music Information Retrieval (ISMIR)*, 2006.