ESCUELA TÉCNICA SUPERIOR
INGENIERÍA INFORMÁTICA

Tesis Doctoral

Análisis de cadenas derivadas de modelos desconocidos. Aplicación al análisis del cante flamenco.

Autor
Javier María Mora Merchán

Directores
Prof. Dr. Carlos León de Mora
Prof. Dr. Joaquín Mora Roche

Sevilla, enero 2018

UNIVERSIDAD D SEVILLA

Agradecimientos

Un trabajo de estas características suele requerir de un esfuerzo considerable mantenido durante largo tiempo. Es imposible sostener dicho esfuerzo sin la ayuda de un grupo considerable de personas.

Agradezco profunda y sinceramente, más allá de lo que pide el protocolo, la contribución de mis directores de tesis. Mi control de versiones es testigo de la gran diferencia entre lo que yo hubiera entregado en primera instancia y el trabajo final obtenido. Me han guiado, motivado y aconsejado para que diera lo mejor de mí.

Una tesis afecta a todos los aspectos de la vida de un doctorando. Un grupo de personas en el departamento de Tecnología Electrónica han estado al quite para asistirme cuando la tesis me ha restado tiempo de otras obligaciones y compromisos. Aun con las molestias que les he ocasionado, continúan saludándome por el pasillo, animándome y ofreciendo su ayuda. En especial, Javier cuyo apoyo ha sido constante desde mi incorporación a la universidad, y Diego y Enrique que demuestran una fe en mí superior a la que posiblemente merezco.

La realización de la tesis también afecta en el plano personal. Familiares se han movilizado para facilitarme el trabajo intentando reducir las distracciones al mínimo y los amigos no han dejado de alentarme y perdonar mis ausencias. Mi Inés, que todavía no sabe qué es una tesis, pero me sonríe cada vez que me ve y Antonia que tanto me soporta y me soporta.

A todos ellos, muchas gracias.

Índice

Listado de siglas empleadas

AMT	*Automatic Music Transcription*
ASCII	*American Standard Code for Information Interchange*
ASL	*Average Shot Length*
CART	*Classification And Regression Trees*
CDF	*Cumulative Distribution Function*
CE	*Computational Ethnomusicology*
CHAID	*CHi-squared Automatic Interaction Detector*
COFLA	*Computational analysis of FLAmenco Music*
CSPA	*Cluster-based Similarity Partitioning Algorithm*
CSV	*Comma-Separated Value*
DBCLASD	*Distribution Based Clustering of Large Spatial Databases*
DBSCAN	*Density Based Spatial Clustering of Applications with Noise*
DE	Distancia de Edición
DEC	Distancia de Edición desde el Centroide
DEMC	Distancia de Edición Media al Centroide
DMC	Distancia Media al Centroide
JCR	*Journal Citation Reports*
JMR	Clasificación de fandangos por el Dr. Joaquín Mora Roche
LCS	*Longest Common Substring*
LDA	*Latent Dirichlet Allocation*
LLE	*Locally-Linear Embedding*
LZW	Lempel-Ziv-Welch
MDR	*Multifactor Dimensionality Reduction*
MDS	*Multidimensional Scaled*
MIDI	*Musical Instrument Digital Interface*
MIR	*Music Information Retrieval*
MIREX	*Music Information Retrieval Evaluation eXchange*
MPEG-7	*Multimedia Content Description Interface*
MTG	*Music Technology Group*
MVU	*Maximum Variance Unfolding*

NFL	Principio *No Free Lunch*
OPTICS	*Ordering Points To Identify the Clustering Structure*
ORB	*Oriented FAST and Rotated BRIEF*
PCA	*Principal Component Analysis*
PDDP	*Principal Direction Divisive Partitioning*
QUEST	*Quick, Unbiased and Efficient Statistical Tree*
SIFT	*Scale-Invariant Feature Transform*
SURF	*Speeded Up Robust Features*
SVD	*Singular Value Descomposition*
SVM	*Support Vector Machine*
THAID	*THeta Automatic Interaction Detection*
tf-idf	*Term Frecuency - Inverse Document Frecuency*
UNESCO	*United Nations Educational, Scientific and Cultural Organization*
UPGMA	*Unweighted Pair Group Method with Arithmetic mean*
UTF8	*Unicode Transformation Format – 8-bit*

Índice de figuras

Índice de tablas

Índice de algoritmos

1 Introducción

1.1 Planteamiento inicial

La presente tesis se centra en el desarrollo de herramientas de análisis sobre productos expresables en forma de cadena y que derivan de unos modelos originales desconocidos.

Modelos desconocidos

El problema de los modelos originales desconocidos (a partir de ahora se mencionarán simplemente como «modelos desconocidos») puede presentarse a partir de un conjunto de restricciones aplicado al modelo de comunicación Shannon-Waver (**figura 1.1**) que identifica los elementos intervinientes en un proceso de comunicación y las relaciones entre ellos. Estas restricciones son:

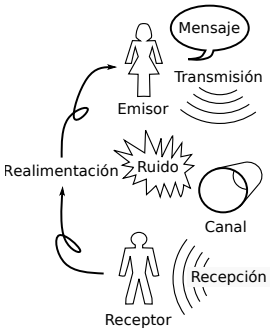


Figura 1.1: Modelo de la comunicación Shannon-Waver (Shannon & Weaver, 1949)

- Existe un número limitado de mensajes originales de contenido desconocido.
- Cada mensaje se puede enviar un número indeterminado de veces.
- Aunque los mensajes sean desconocidos, el código en el que se expresa el mensaje es conocido y este puede ser transcrito en una cadena.
- No hay realimentación. La comunicación es unidireccional y no hay mecanismos de control sobre la transmisión. No se puede negociar un código ni solicitar retransmisiones.
- Por último, el canal de transmisión no es fidedigno: los mensajes pueden alterarse o perderse sin que el receptor pueda detectar dicha alteración.

El resultado es un sistema de transmisión en el que los mensajes recibidos son alteraciones de los mensajes originales (los modelos desconocidos) y cuyas características dependerán en gran

medida de estos. Según este modelo, las distintas instancias mostradas en la **figura 1.2**, no serían más que retransmisiones de un mismo mensaje que han recibido distintas alteraciones en el canal de transmisión.

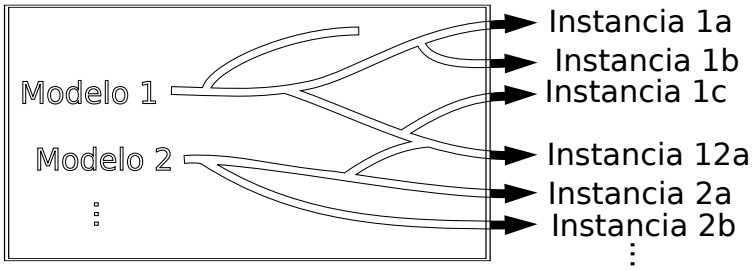


Figura 1.2: Representación de la primera hipótesis de trabajo

Los sistemas basados en modelos desconocidos, con las restricciones que hemos impuesto, no son tan difícil de encontrar como pudiera parecer. En general, los sistemas de transmisión oral, ya sean de historias en verso como los romances, historias cortas como chistes o canciones populares (por citar algunas), sufren este mismo tipo de transformación donde no existe garantía de que lo que escuchamos sea una copia fiel de la creación original y no tenemos medio de comunicarnos con el autor.

La transmisión oral no es el único mecanismo de transmisión que presenta estas características. Muchos mecanismos de transmisión biológica (como la transmisión de genes o enfermedades) pueden sufrir las mismas restricciones, ya que aunque pudieran conocerse los mecanismos de transmisión, pueden aparecer otros factores que alteren los datos observados. Así, los signos patognomónicos de una enfermedad (los síntomas que la definen) representa el modelo desconocido y la sintomatología del paciente la instancia observada (que no tiene que coincidir completamente con los anteriores).

Un último ejemplo de sistemas de estas características son ciertos problemas de análisis literario como el que se produce al analizar las variaciones entre distintas versiones de una obra (quizás los estudios en lengua castellana más conocidos sean las comparaciones entre las ediciones iniciales del Quijote como la realizada por (Rico, 1998)).

En general, todos estos sistemas de comunicación presentan un problema de trazabilidad en la que no es posible acceder ni al autor, ni al mensaje original, ni conocer en plenitud el canal de transmisión.

La cadena como soporte de información

Como ya se ha mencionado, en este estudio se ha limitado el problema de los modelos desconocidos a aquellos casos en los que los mensajes pueden transcribirse en forma de cadenas. Una cadena[1] no es más que una sucesión de términos o *tokens* tomados de un conjunto finito. Es, por tanto, una estructura ideal para almacenar información en la que los elementos siguen una relación de orden como en los casos de un muestreo temporal o espacial.

[1.1] En la **sección 2.1.1** se dará una definición formal del concepto de cadena.

Las cadenas son estructuras de información simples que presentan facilidad de acceso y manipulación de los datos almacenados. La versatilidad de las cadenas para almacenar todo tipo de información y su capacidad de reflejar[2] la misma información que otras estructuras de datos más complejas, hacen que la restricción impuesta de trabajar con cadenas no sea realmente un factor muy limitante.

[1.2] Una serialización de una estructura de datos muy compleja, proporcionará cadenas de gran complejidad. Es posible sacrificar partes de la información original para obtener cadenas más sencillas.

El flamenco como campo de pruebas

Además de desarrollar una serie de técnicas de análisis, se ha decidido incorporar el ensayo de las mismas sobre un campo de estudio en desarrollo: el análisis de cantes flamencos.

El flamenco es una manifestación cultural de casi doscientos años de antigüedad que actualmente goza de un interés creciente. Según la *Wolfram Alpha Knowledgebase* (WolframAlpha, 2017) el uso del término «flamenco» se ha sextuplicado desde 1980 y se ha duplicado entre 2000 y 2008 (**figura 1.3**)[3].

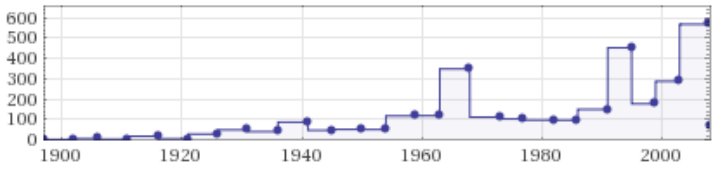


Figura 1.3: Frecuencia de aparición del término flamenco en miles de millones de apariciones (WolframAlpha, 2017)

[1.3] No hay disponibles datos más recientes; pero la tendencia en este indicador y en otros similares como Google Ngrams o Google Trends van en el mismo sentido.

Las peculiaridades específicas del flamenco, requiere que la industria de contenidos audiovisuales se adapte con el fin de proveer sistemas descripción, clasificación e identificación automáticas de piezas que permitan, posteriormente, desarrollar sistemas de recomendación adaptados, así como otras herramientas de evaluación de piezas.

Desde el punto de vista académico, la investigación del flamenco comparte algunos objetivos con la industria; pero añade la necesidad de incorporar mecanismos de difusión del conocimiento ya sea a otras personas que quieran aprender sobre el tema, a otros expertos o para el desarrollo de sistemas de gestión de la información automáticos.

El flamenco es una manifestación cultural muy rica que comprende áreas muy variadas como análisis musical (melodía, armonía, ritmo, acompañamiento, ...), análisis desde las humanidades (evolución histórica, análisis lingüístico-literario, ...) o el plástico (el baile, escultura, pintura, cine, ...).

Es un campo demasiado amplio para intentar abarcarlo en un trabajo de estas características por lo que se ha decidido trabajar en el estudio computacional de la melodía flamenca centrado en unos pocos estilos. La selección de dichos estilos dependerá de la disponibilidad de un corpus representativo de los estilos y de la información que pudiera servir para validar los resultados obtenidos.

Finalmente, además de ser un dominio apto para el ensayo de las herramientas usadas, existe un aliciente extra, de tipo cultural, para usar el flamenco como campo de estudio. El flamenco es una parte fundamental del patrimonio cultural de Andalucía. Así se recoge en el estatuto de autonomía de nuestra comunidad, junto con el compromiso de los poderes públicos de financiar, entre otros, la investigación en este campo. En las últimas convocatorias de proyectos de investigación y proyectos de excelencia se ha potenciado la investigación universitaria sobre este campo.

Aún más, el flamenco ha llegado a ser uno de los elementos de la Marca España desbordando los límites de nuestra comunidad. En consecuencia, se utiliza en la promoción de eventos como en la pasada presentación de la candidatura olímpica de Madrid o siendo usada la música sobre la que el equipo español de natación sincronizada ha realizado sus ejercicios en las recientes olimpiadas de Rio 2016. Diversas comunidades españolas amparan el estudio del flamenco en sus programas de investigación destacando en este sentido Cataluña, Madrid, Extremadura y Murcia.

La confluencia entre inteligencia computacional y flamenco tiene ya su reconocimiento internacional. Hay numerosas aportaciones a congresos especializados de orientación MIR (*Music Information Retrieval*) y se ha publicado ya un primer artículo (Mora, Gómez, Gómez, & Díaz-Báñez, 2016) en revista indexada

por el JCR. Sin embargo, la tecnología aplicada a la música clásica, a la música pop o a la folklórica no es, en su mayoría, aplicable al análisis del flamenco precisamente por las irregularidades derivadas de su carácter de improvisación. Esto hace que los estudios publicados estén aún en los albores de su desarrollo científico. Por su dificultad, especificidad y por el desconocimiento, hasta ahora, de la comunidad científica de estos temas, ha sido poco tratado el análisis del flamenco en el área de la inteligencia computacional, convirtiéndose esta tesis en una de las pioneras.

El problema del análisis de las melodías flamencas es un reto difícil ya que está incluido en la categoría, desde el punto de vista lógico, de los problemas mal definidos: no existe un corpus de análisis claro y existen distintos enfoques para analizar los datos que proporcionan múltiples posibles respuestas válidas. Los problemas mal definidos presentan dificultades sustanciales para evaluar posibles soluciones cuando se intenta decidir cuál de ellas es la mejor.

El análisis a partir de modelos desconocidos presenta un enfoque independiente de las distintas corrientes de estudio del flamenco representando un paso significativo en la etnomusicología computacional.

1.2 Objetivos planteados

A partir de las consideraciones expuestas, es posible establecer los objetivos perseguidos en este trabajo.

El objeto de estudio de la presente tesis doctoral es el análisis de instancias (codificadas como cadenas) derivadas de modelos desconocidos.

El desconocimiento de los modelos originales es el gran hándicap de estos problemas. Conocer los originales permitiría una extracción efectiva de información de las cadenas, las clasificaciones en grupos (Juan & Vidal, 2000), el estudio de los mecanismos de deformación de las cadenas (Subramaniam, Roy, Faruquie, & Negi, 2009) o la posibilidad de establecer relaciones entre las cadenas (Lodhi, Saunders, Shawe-Taylor, Cristianini, & Watkins, 2002a).

La restricción impuesta de que las instancias estén codificadas en cadenas permite concretar las mismas en elementos procesables (ya que una instancia como idea es un elemento intangible). Esta imposición no es demasiado restrictiva ya que la cadena es una estructura de datos general capaz de contener información

muy variopinta procedente de muestreos (temporales o espaciales), procedentes de conversiones desde otras estructuras de información (como un árbol o una matriz), datos en los que pueda establecerse un orden, proyecciones lineales de conjuntos de datos o combinaciones entre estas.

Concretamente, se ha planteado como objetivo dar respuesta, para un conjunto dado de cadenas, a las siguientes cuestiones:

Objetivo 1
- ¿Qué cadenas derivan de un mismo modelo?

Objetivo 2
- ¿Qué caracteriza a las cadenas que derivan de un mismo modelo?

Objetivo 3
- ¿Qué partes de las cadenas son más probables que procedan del modelo?

Cuestiones que se resolverán primeramente desde un punto de vista general y, posteriormente, aplicado a los cantes flamencos.

1.3 Estructura de la tesis

La presente memoria está estructurada en 5 capítulos donde se describe el trabajo realizado y los resultados obtenidos. Adicionalmente al texto principal, un conjunto de anexos incluyen información complementaria como otros materiales desarrollados en el transcurso de la investigación o tablas de resultados que por su extensión no se han incluido en el texto principal de la memoria.

En el **capítulo 2** se efectúa una doble revisión del estado del arte. Por un lado se efectúa un estudio de técnicas de análisis sobre las que se encuadran las propuestas en este trabajo. La segunda parte del capítulo está dedicada a un estudio de las características del flamenco y sus lineas actuales de investigación en análisis computacional.

Los capítulos **3** y **4** exponen las técnicas genéricas desarrolladas con el fin de satisfacer los objetivos planteados en la tesis y su aplicación sobre un corpus de cantes flamencos.

Finalmente, el **capítulo 5** recoge las principales conclusiones y resultados del trabajo realizado así como una enumeración de las aportaciones originales presentadas y de futuras líneas de investigación que se abren a tenor de los resultados obtenidos.

Tres anexos cierran el volumen con material complementario:

– El **anexo A** describe una formalización en la descripción de funciones más apropiada para algoritmos que la notación estándar matemática.

– El **anexo B** presenta la herramienta de transcripción desarrollada, sus motivaciones y un manual de uso.

– El **anexo C** compila una serie de tablas de resultados obtenidos que, por su tamaño, romperían el discurso si se colocaran dentro del texto principal de la memoria. No obstante, los resultados obtenidos justifican su inclusión para futuras consultas.

2 Revisión del estado del problema

El presente capítulo de revisión del estado del problema se presenta dividido en dos grandes secciones diferenciadas por su contenido: herramientas de procesado de cadenas y la problemática del flamenco.

La primera sección está dedicada al estudio de técnicas de análisis de cadenas y presenta una forma original de caracterizar las fases de los algoritmos de aprendizaje automático (*machine learning*) en función de su objetivo: la «adecuación» y la «aplicación». En esta primera sección describiremos las diferencias entre la adecuación y la aplicación procesado y haremos una revisión de algoritmos de análisis de cadenas centrándonos en aquellos útiles para los objetivos de la presente tesis.

Para organizar el conjunto de algoritmos descritos en la sección de análisis de cadenas, se han dividido en bloques siguiendo la **tabla 2.1** (que se verá un poco más adelante). Estos bloques son: métricas, agrupaciones, descriptores, clasificadores y por último generación de gramáticas.

La segunda parte del capítulo está dedicado a realizar una presentación de cante flamenco, sus características, la problemática específica de su análisis y lineas actuales de investigación en el campo del MIR (*Music Information Retrieval*).

2.1 Análisis de cadenas

Bajo el nombre de *Machine Learning* se identifican un conjunto de técnicas que se engloban en dos grandes familias (en función del tipo de objetivo que persiguen): métodos de predicción y métodos de descripción. Los métodos de predicción pretenden determinar el comportamiento de un conjunto de datos asociados (que llamaremos observación), mientras que los métodos de descripción se centran en explicar el porqué de la observación efectuada. Estos grandes grupos presentan cierto solapamiento ya que algunos modelos predictivos pueden ser capaces de justificar sus resultados de forma inteligible y, de igual forma,

algunos modelos descriptivos pueden ser usados para efectuar predicciones.

Independientemente del tipo de objetivo buscado, (Fayyad, Piatetsky-Shapiro, Smyth, et al., 1996) identifica seis categorías en función del objetivo concreto perseguido. Estas son:

— **Clasificación** que identifica cada observación con una o más etiquetas predefinidas.

— **Regresión** que asocia a cada observación con un número.

— **Agrupación** que identifica un número finito de categorías que describen los datos.

— **Resumen** que identifica descripciones compactas para subconjuntos de datos.

— **Dependencias de modelado** que busca un modelo que explique dependencias significativas entre variables.

— **Detección de cambio o deriva** centrado en descubrir los cambios más significativos en los datos desde la última medida, así como detectar un comportamiento que se desvíe de valores considerados normales.

El éxito de un proceso de aprendizaje automático, de cualquier tipo de los mencionados, depende de distintos factores de entre los que destacamos: la elección de un algoritmo adecuado para nuestro propósito y una correcta adecuación de los datos para dicho algoritmo (Witten, Frank, Hall, & Pal, 2016). Estos dos factores nos permiten dividir, idealmente, la aplicación de la minería en dos etapas. La primera etapa (la adecuación) es específica de cada tipo de objeto considerado para el análisis (ya sea una imagen, una muestra de audio, una cadena de proteínas, un texto…) y su objetivo es llegar a una serie de descripciones normalizadas. Estas descripciones normalizadas son las que, posteriormente en la segunda etapa, usan los algoritmos de análisis.

Aunque se insistirá de ello al inicio del **capítulo 3**, es importante destacar la diferencia, en concepto, entre el preprocesado y la adecuación. El preprocesado de datos son las operaciones que se efectúan sobre cada observación para adecuarlo a su procesado[1]. La adecuación se encarga de las operaciones que se aplican a todos los datos de un tipo determinado para ajustar las características del tipo a las necesidades del procesado incluyendo la unificación de datos de fuentes heterogéneas (Guerrero, García, Personal, Luque, & León, 2017). Si bien, algunas de las operaciones pueden ser comunes a ambos conceptos, existen operaciones

[1] Algunas operaciones son el filtrado de datos redundantes o incorrectos, escalado o normalización de las entradas, etc.

específicas a cada una como técnicas de sustitución de datos perdidos (específica del preprocesado) o la definición de una métrica para efectuar una agrupación (específica de la adecuación).

Esta estructura en dos pasos tiene beneficios en ambos sentidos. Cualquier tipo de objeto nuevo que pueda llegar a la descripción normalizada puede aprovecharse de los algoritmos de análisis existentes y cualquier algoritmo nuevo que parta de una de estas descripciones puede aplicarse a todos los objetos existentes.

1ª Etapa (Adecuación)	2ª Etapa (Aplicación)
Punto de un espacio métrico	Agrupación
Punto de un espacio de características	Reconocimiento de patrones discriminantes
Grafo	Análisis de grafos

Tabla 2.1: Descripciones normalizadas para distintas técnicas de Análisis

Obviamente, existen gran cantidad de algoritmos y no todos requieren la misma adaptación. La **tabla 2.1** resume, en la primera columna, formas normalizadas requeridas para las familias de algoritmos de la segunda necesarios para estudiar los objetivos de esta tesis. Las adecuaciones persiguen adaptar las formas de las cadenas para su utilización en los algoritmos aplicados.

Como ejemplo de adecuación, la construcción de una métrica apropiada para un tipo de elemento permite medir la distancia entre los elementos de dicho tipo y esa medida de distancia es necesaria para poder usar algoritmos de agrupación (*Clustering*)[2] que buscan la distribución de los objetos en grupos en función de su similitud.

[2.2] (Xu & Tian, 2015)

No todos los objetos analizados, en un momento dado, han de compartir la misma forma o descripción. En el segundo ejemplo mostrado en la **tabla 2.1**, se busca una transformación de los objetos a una forma normalizada. Así, no todas las cadenas o muestras de audio tienen la misma longitud y, en un caso más extremo, no existen dos películas iguales (con el mismo guión, los mismos actores, misma música, duración...). Con el fín de poder trabajar con un grupo heterogeneo de objetos de un tipo, es necesario establecer una transformación entre cada instancia de dicho tipo a una forma común: un punto en un espacio de características. Para ello, se construye (para cada objeto analizado) un punto en el que cada coordenada de éste es una propiedad específica del objeto[3], que ya es procesable por los procedimientos de reconocimiento de patrones discriminantes (*Discriminant Pattern Recognition*) existentes.

[2.3] Un último ejemplo: es difícil a priori saber si el precio de una casa es justo o no. Antes de realizar un análisis de precios, es necesario adaptar el objeto «casa» en un punto de propiedades como: superficie construida, número de habitaciones, número de plantas, precio pedido, localización, presencia de piscina…

Una vez situados los objetos como puntos del espacio de características, es posible dividir el mismo en subespacios que identifiquen los objetos como miembros de una categoría u otra. El objetivo del reconocimiento de patrones discriminantes no es otro que encontrar la división adecuada de dicho espacio de características. La calidad en la identificación de los objetos (o lo que es lo mismo, la capacidad de discriminación de los subespacios encontrados) dependerá de los objetos de los que se partan; pero, en gran medida, también dependerá de la elección de las características empleadas[4].

La conversión de cadenas a grafos es el último caso de adecuación de cadenas con el que se ha trabajado en esta tesis. Al igual que la conversión a puntos del espacio de características, ésta es una proyección de la información original a una forma útil para el análisis que se propone.

Tanto las cadenas como los grafos dirigidos comparten la existencia de una noción de orden (en las cadenas impuesto por su propia definición de secuencia ordenada de símbolos y en los grafos es explicitado por la direccionalidad de los arcos presentes). Por lo que la conversión cadena-grafo que se propone, sólo requiere establecer reglas de selección de las subcadenas que formarán los nodos del grafo (ya que el orden entre los nodos lo proporcionará el orden natural que ocurre en las cadenas).

Para la selección de las subcadenas-nodos se ha recurrido a herramientas en el campo del reconocimiento de patrones estructurales (*Structural Pattern Recognition*) que son una sistematización moderna (a partir de los años 50 (Chomsky, 1959)) de estudios del análisis de lenguaje natural mucho más antiguos[5]. En él, se busca definir las reglas de formación que construyan cadenas que pertenezcan a un cierto lenguaje[6]. Como efecto secundario, y deseable, de la construcción de las reglas está la identificación de subcadenas[7] de interés que usaremos como nodos del grafo.

El principio *No free lunch* (Wolpert, 1996) establece que el rendimiento general[8] de los algoritmos de aprendizaje automático es similar. Las mejoras posibles de rendimiento vienen, por tanto, de dos caminos posibles: una mejora en la calidad de los datos suministrados a estos algoritmos (entre ellos gracias a una mejor adecuación de los datos) y en la búsqueda del mejor algoritmo específico para el problema concreto analizado. Dado que en la presente memoria se presentan herramientas generales de análisis la segunda vía no es un camino viable. Es por ello que en las descripciones que siguen en el capítulo, y aunque se resuman algunas de las técnicas análisis, se hará un especial hincapié

2.4 Para determinar si una película es de acción, usar la primera letra del título o lo que cobra el actor protagonista son peores indicativos que analizar el *tempo* de la banda sonora o medir la duración promedio del plano (ASL, *Average Shot Length*).

2.5 Un ejemplo de ello es la gramática de Antonio de Nebrija (de Nebrija, 1492).

2.6 Parejo a la construcción de las reglas, el *Syntactic Pattern Matching* estudia los procedimientos para verificar si cierta cadena cumple con una reglas dadas

2.7 Los «símbolos no terminales».

2.8 El teorema NFL plantea que el rendimiento de los algoritmos ante los problemas es de suma cero. Algoritmos con buen desempeño en un tipo de problemas tendrá, en compensación, un mal desempeño en los demás.

en la adecuación de los datos. No obstante, existen gran cantidad de monografías sobre análisis (como (Hastie, Tibshirani, & Friedman, 2016) o (Provost & Fawcett, 2013)) en los que se trata el tema con una mayor extensión y profundidad de la que se le puede destinar aquí.

El resto de los apartados del capítulo describirá, con las limitaciones ya expuestas, tanto las técnicas de adecuación como alguna técnica de análisis usada a lo largo de la investigación desarrollada. Previamente a ello, es necesario describir, con cierta formalidad, los términos que se usarán durante el resto de la memoria para evitar posibles ambigüedades en el texto.

2.1.1 *Definiciones útiles*

Cadena en *A* (o sobre *A*) Una cadena en (o sobre) A es una secuencia finita de ocurrencias de elementos de A (Partee, ter Meulen, & Wall, 1990).

El tipo cadena puede representarse[9] como:

$$Cadena :: [A] \qquad \text{(TIPO 2.1)}$$

[9] En el **anexo A** se detalla la sintaxis empleada para la descripción de tipos.

Alfabeto (o Vocabulario) El conjunto finito A sobre el que se construye una cadena.

Símbolo (o *token*) Cada uno de los elementos del alfabeto. En el caso de una cadena de texto, se puede usar también «caracter» o incluso «letra».

Una cadena de texto ASCII[10], es un tipo de cadena en el que el alfabeto es un conjunto de 256 elementos que representan letras y símbolos para ser impresos. Aunque son un tipo de cadenas, no todas las cadenas son para registrar textos.

[10] En el caso de una cadena Unicode (the Unicode Consortium, 2017) el alfabeto está compuesto por un conjunto de 1 114 112 elementos.

Es importante destacar que una cadena es un concepto que no impone cómo ha debe almacenarse en un sistema informático. Donde la librería estándar del lenguaje C (Kernighan & Ritchie, 2006) almacena las cadenas en un *array* (posiciones contiguas de memoria), la de Haskell (Marlow, 2010) las almacena como una lista enlazada de caracteres y en MoarVM[11] se almacenan como *strands* (hebras: una lista de arrays de memoria). Aun cuando conceptualmente una cadena no sea equivalente a vector, array o lista de caracteres, es posible implementar las cadenas en la estructura de datos que más nos convenga en cada ocasión, quedando a nuestra disposición todas las técnicas de análisis existentes para aquellas.

[11] Una máquina virtual sobre la que se ejecuta Perl 6

Otra forma de expresar la cadena es como una función que ante una posición devuelve el token que ocupa dicha posición en la secuencia.

$$Cadena' :: Int \rightarrow Maybe\ A \qquad \text{(TIPO 2.2)}$$

Esta definición alternativa proporciona mecanismos para, posteriormente construir subcadenas. El tipo de salida (*Maybe A*) indica que la cadena puede no devolver valores en caso de que la posición indicada no esté en los límites válidos de la cadena.

Longitud de la cadena número de tokens de la secuencia. Se representa como $|a|$ (siendo a una cadena).

Conjunto de cadenas Denotamos como A^* el conjunto de todas las posibles cadenas en A. Este conjunto es enumerable y es isomorfo al de los números naturales.

Si se fija una longitud máxima de las cadenas (a N), la cardinalidad (número de elementos) de este conjunto es:

$$|A^*| = |A|^N \qquad (2.3)$$

Concatenación El operador concatenación «$\frown$» crea una cadena como combinación de dos de forma que la secuencia de simbolos de la segunda cadena es añadida, en el mismo orden, al final de la primera.

El conjunto de todas las cadenas sobre A y el operador concatenación ($< A^*, \frown >$) forman un monoide. Y por tanto presenta las siguiente propiedades: cualquier concatenación de cadenas es una cadena, satisface la asociatividad y presenta un elemento neutro (e, la cadena vacía). En cambio no satisface la propiedad del elemento inverso ni la conmutatividad (por lo que ni es grupo, ni Monoide Abeliano).

Subcadena Dada una cadena x, una subcadena de x es cualquier cadena formada de ocurrencias consecutivas de símbolos de x tomados en el mismo orden en los que aparecen en x. Más formalmente:

y es una subcadena de x si y solo si $\exists z,w \in A^* \big/ x = z \frown y \frown w$.

Donde z se llama «prefijo» y w «sufijo». Tanto z como w pueden ser igual a la cadena vacía (e), por lo que toda cadena es subcadena de si misma y e es subcadena de todas las cadenas.

Gramática Sea $\Sigma = V_T \cup V_N$. Una gramática G se define como una 4-tupla $< V_T, V_N, S, R >$, donde

- $V_T \subset A$ es el conjunto «Alfabeto terminal».
- V_N es el conjunto «Alfabeto no terminal». Los conjuntos V_T y V_N son conjuntos finitos disjuntos.
- $S \in V_N$ es un elemento distinguido de V_N: el «Símbolo inicial» y
- R es un conjunto de pares ordenados en $\Sigma^* V_N \Sigma^* \times \Sigma^*$ que funcionan como reglas de sustitución. Si en una cadena se encuentra una subcadena que coincide con el primer elemento de un par, esta puede sustituirse por el segundo elemento del mismo par.

El proceso de derivación de una gramática consiste en empezar con una cadena con el símbolo S e ir aplicando reglas sucesivas de sustitución hasta que no haya ningún elemento no terminal. En cada paso, es posible que existan más de una regla de sustitución aplicable y la elección de una u otra regla propiciará terminar con una cadena u otra.

Lenguaje (sobre el vocabulario *A*) Subconjunto de A^*. Todas las cadenas que se pueden derivar de una gramática dada.

2.1.2 *Métricas*

Las métricas son funciones que calculan similitud entre elementos. Esta similitud nos permite estimar propiedades de los elementos en función de las propiedades de los elementos con los que se comparan. Aun cuando ningún elemento usado esté previamente caracterizado, las métricas identifican agrupaciones de las que se pueden extraer información.

La existencia de una métrica para comparar cadenas, hace de A^* un Espacio Métrico del que compartirá todas sus propiedades. Formalmente, definimos un Espacio Métrico como:

Espacio Métrico Un espacio métrico (Fréchet, 1906) es un par ordenado (M,d) donde M es un conjunto y d es una métrica en M.

Métrica Sea $x, y, z \in M$. d es una métrica si cumple los siguientes requisitos:

— **No negatividad** $\mathrm{d}(x,y) \geq 0$

— **Identidad de indiscernibles** $\mathrm{d}(x,y) = 0 \Leftrightarrow x = y$

— **Simetria** $\mathrm{d}(x,y) = \mathrm{d}(y,x)$

— **Desigualdad Triangular** $\mathrm{d}(x,z) + \mathrm{d}(z,y) \geq \mathrm{d}(x,y)$

Por lo que el tipo que define la métrica es:

$$\mathrm{d} :: M \to M \to \mathit{Real} \qquad \text{(TIPO 2.4)}$$

Las métricas definen la similitud a partir de la distancia o diferencia entre los puntos del espacio comparados (cuanto mayor sea dicha distancia, menor parecido hay entre dichos puntos). Existen funciones de comparación que directamente devuelven la similitud entre puntos (esto es, mayor valor implica mayor similitud). Aunque estas funciones de comparación no son métrica, no es difícil construir nuevas funciones a partir de estas. Por ejemplo:

- Si la función similitud (sim) tiene un máximo (m) definido,

$$\mathrm{d}(x,y) = m - \mathrm{sim}(x,y) \qquad (2.5)$$

- Si la función similitud es siempre positiva,

$$\mathrm{d}(x,y) = \frac{1}{\mathrm{sim}(x,y)} \qquad (2.6)$$

A lo largo del tiempo se han desarrollado muchas métricas para comparar cadenas. En la presente relación se ha optado por presentarlo en grandes bloques o familias en las que destacar el principio que unifica a cada una. Estas familias son:

- distancia de edición,
- alineamiento,
- basado en vectores,
- basados en conjuntos y
- metamétricas

Para completar la revisión, se ha añadido un último apartado de comparación que, aunque ya no tratan de métricas, determinan si dos cadenas son o no equivalentes.

Edit Distance

Edit distance calcula el coste mínimo de transformación de una cadena a otra. Para ello, establece un conjunto de operaciones de transformación, asigna un coste a cada una y busca el conjunto de operaciones que minimicen el coste total de la transformación entre cadenas. Las operaciones más comunes son:

— **Adición** Añadir un símbolo entre dos símbolos existentes.

— **Eliminación** Elimina un símbolo de la cadena.

— **Sustitución** Sustituye un símbolo por otro en la cadena.

— **Intercambio** Intercambia la posición de dos símbolos adyacentes.

Estas operaciones no son las únicas posibles y existen otras de propósito más específico como:

— **Prefijo/sufijo** Usado en biología, permite añadir o quitar prefijos o sufijos complejos a un coste muy inferior de hacerlo símbolo a símbolo.

La distancia Levenshtein es la más popular en esta familia. Tanto que toda mención a una «*Edit Distance*» que no especifique pesos u operadores hace referencia a la distancia Levenshtein.

En la **figura 2.1** se muestra una posible transformación de la cadena «graso» a «carro». Las operaciones efectuadas han sido una sustitución, una eliminación, conservar un símbolo, sustitución, adición y conservación. En total 4 transformaciones. Dado que no existe una combinación que efectúe la transformación en menos operaciones (si ha otras combinaciones en el mismo número de operaciones), la distancia de edición «graso»-«carro» es igual a 4.

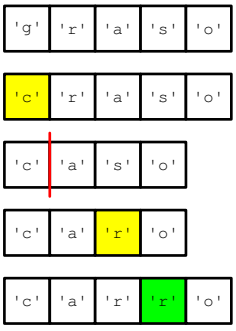


Figura 2.1: Transformación de «graso» en «carro»

En general, la distancia de edición de dos cadenas puede expresarse como una función de coste acumulativa ($H(i,j)$) que tiene la forma:

$$H(i,j) = \text{mín} \begin{cases} H(i-1,j) + c_1 & \text{(inserción)} \\ H(i-1,j-1) + c_2 & \text{(sustitución/coincidencia)} \\ H(i,j-1) + c_3 & \text{(borrado)} \\ \cdots & \text{(otras operaciones)} \end{cases} \tag{2.7}$$

donde $H(i,j)$ es el coste de transformación de subcadenas de las cadenas s_1 y s_2 comparadas (hasta la longitud i y j respectivamente). Siendo, por tanto, $H(|s_1|,|s_2|)$ el valor de la distancia final que se está calculando. Los elementos c_1, c_2 y c_3 (que no tienen por qué se constantes) son los costes asociados a cada operación.

Dado que cada operación de transformación tiene su antagónica, si los costes de estas operaciones son iguales, la distancia será una función simétrica.

El caracter recursivo de estas expresiones de coste hace que el cálculo de la distancia, entendiendo H como una función, sea prohibitivo. De ahí que se hayan desarrollado distintas técnicas de optimización de espacio de memoria y operaciones. La más famosa de ellas es el algoritmo Wagner–Fischer (Wagner & Fischer, 1974) que sustituye la función H por una matriz y establece un orden de cálculo que evita las llamadas recursivas.

Las distintas variantes existentes en la distancia de edición se basan en la asignación de distintos valores en los costes o en expresiones de H más complejas.

Levenshtein

La distancia Levenshtein (Levenshtein, 1966) entre dos cadenas se define como el número mínimo de transformaciones necesarias para transformar una en otra. Las operaciones de transformación permitidas son inserción, borrado o sustitución de un simple símbolo de la cadena.

$$H(i,j) = \text{mín} \begin{cases} H(i-1,j)+1 \\ H(i-1,j-1)+c_{sc}(s_{1,i},s_{2,j}) \\ H(i,j-1)+1 \end{cases} \tag{2.8}$$

Donde los costes de sustitución/coincidencia dependen de los símbolos en las posiciones i y j ($s_{1,i}$ y $s_{2,j}$) y son:

$$c_{sc}(x,y) = \begin{cases} 1 & \text{si } x \neq y \\ 0 & \text{si } x = y \end{cases} \tag{2.9}$$

Los valores posibles de la distancia están acotados: valiendo 0 exclusivamente si las dos cadenas son iguales y como máximo la longitud de la cadena más larga. Si las dos cadenas miden distinto, el valor mínimo de la distancia será la diferencia de longitudes de las dos cadenas.

Damerau-Levenshtein

Extensión de Levenshtein inspirada por el trabajo de Damerau (Damerau, 1964) sobre la construcción de correctores ortográficos. Damerau presentó las cuatro causas más comunes de errores mecanográficos Su aportación consistió en añadir un cuarto operador de transposición en el que intercambiaba la posición de dos caracteres contiguos.

La fórmula recursiva de cálculo de la nueva distancia es:

$$\mathrm{d}(i,j) = \text{mín} \begin{cases} \mathrm{d}(i-1,j)+1 \\ \mathrm{d}(i,j-1)+1 \\ \mathrm{d}(i-1,j-1)+c_{sc} \\ \mathrm{d}(i-2,j-2)+1 \end{cases} \quad \text{solo si } s_{1,i} = s_{2,j-1} \text{ y } s_{1,i-1} = s_{2,j} \tag{2.10}$$

Hamming

De la familia *Edit Distance*, solo incorpora la operación de sustitución. Por ello, la distancia Hamming (Hamming, 1950) solo es aplicable en subespacios de cadenas de igual longitud.

$$H(i,j) = H(i-1,j-1) + c_{sc}(s_{1,i},s_{2,j}) \tag{2.11}$$

Con la función de coste c_{sc} es la misma que en Levenshtein.

El valor de la distancia que proporciona es equivalente al número de símbolos que coinciden en posición pero no en valor. Esta distancia es aplicada en muchos campos para detectar errores de transmisión. En el ámbito de las telecomunicaciones también es conocida como *signal distance*.

Basados en alineamientos

El comportamiento de estas métricas se basa en una matriz de similitud que asigna a cada par posible de tokens una energía de contacto (que puede ser positiva si aumenta la estabilidad[12] o negativa si provoca inestabilidad en la unión) y establece un coste energético por un hueco.

[12] El convenio empleado es por tanto contrario al de los sistemas físicos.

La similitud entre las dos cadenas es la energía de la configuración de alineación más estable entre ambas cadenas y proporcionan una medida de similitud en vez de la distancia ya que valores más elevados indican mayor similitud.

Esta familia de métricas imitan el comportamiento de las secuencias de proteinas y ácidos nucleicos y han sido diseñados especialmente para trabajos en bioinformática.

Las cadenas «GCATGCU» y «GATTACA» pueden alinearse como:

```
GCATG-CU      GCA-TGCU      GCAT-GCU
G-ATTACA      G-ATTACA      G-ATTACA
```

en función de la matriz de similitud y el algoritmo usado, se puede determinar qué alineamiento es más estable y cuánto vale la energía (similitud) entre las cadenas.

Aunque existen diferencias conceptuales entre esta familia y las distancias de edición, (Sellers, 1974) demuestra que el problema de alineamiento es equivalente al problema de las distancias de edición. De ahí que también se utilice Wagner–Fischer como procedimiento de cálculo.

Needleman-Wunch

También conocido como Needleman-Wunch-Sellers (Needleman & Wunsch, 1970), Sellers y el algoritmo de mejora de Sellers.

La innovación de Needleman-Wunch-Sellers frente a Levenshtein es que los costes por sustitución dependen de los elementos que se están comparando.

$$H(i,j) = \text{máx} \begin{cases} H(i-1,j) + gap & \text{(hueco)} \\ H(i-1,j-1) + e_c(s_{1,i},s_{2,j}) & \text{(energía contacto)} \\ H(i,j-1) + gap & \text{(hueco)} \end{cases} \tag{2.12}$$

donde *gap* es el coste por la creación de un hueco y e_c es la matriz de similitudes entre tókens que es la que determina la energía entre cada par de elementos del alfabeto.

Smith-Waterman

Es una variación de Needleman-Wunch-Sellers en el que no se permite que los valores de energía que se van calculando sean negativos (saturándolos en cero).

En este caso, el valor de la semejanza entre las cadenas ($\text{sim}(s_1,s_2)$) no coincide con $H(i_{max},j_{max})$ (el valor de la energía cuando se han procesado las dos cadenas) sino que es el máximo de todas las energías calculadas en el proceso.

$$\text{sim}(s_1,s_2) = \max_{i,j} H(i,j) \tag{2.13}$$

El valor de la similitud proporcionada está relacionado con el tamaño de las mayores subcadenas que encajan entre si.

Gotoh

También llamado Smith-Waterman-Gotoh (Gotoh, 1982). Es una variación de Smith-Waterman que establece un coste distinto para el inicio de un hueco (identificado previamente como *gap*) y la continuación del mismo.

Subcadena común más larga (LCS, Longest Common Substring*)*

Medida de similaridad entre cadenas consistente en la longitud de la subcadena en común más larga. Esta medida está acotada entre 0 (no hay ninguna coincidencia de contenidos) y la longitud de la cadena más corta por lo que si fuera necesario, el resultado se puede normalizar al rango [0,1].

Para su cálculo se puede utilizar un procedimiento similar la de las distancias de edición y las técnicas de alineamiento:

$$H(i,j) = \begin{cases} H(i-1,j-1)+1 & \text{si } s_{1,i} = s_{2,j} \\ 0 & \text{resto} \end{cases} \tag{2.14}$$

Siendo la subcadena común más larga:

$$LCS(s_1,s_2) = \max_{i,j} H(i,j). \tag{2.15}$$

Basados en vectores numéricos

Las distancias de esta categoría son realmente normas de $\mathbb{R}^n$ reconvertidas en distancias de cadenas. Todas estas distancias sufren de dos limitaciones:

- Es necesario una función de conversión entre cadenas y vectores en $\mathbb{R}^n$. Para ello hay que definir una función de conversión de los símbolos del alfabeto:

$$Símbolo2Número :: A \to \mathbb{R} \qquad \text{(TIPO 2.16)}$$

- Todas las cadenas comparadas han de tener el mismo tamaño.

p-normas o distancias Minkowski

Los espacios de Lebesgue son espacios métricos, de dimensión finita n, en los que la norma que los define tiene una forma específica:

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p} \tag{2.17}$$

donde se ha de cumplir $p \geq 1$. Esta norma se denomina p-norma (aunque a veces se usa ℓ_p o L_p para relacionarla con su espacio de Lebesgue asociado).

A partir de una norma, que determina el tamaño de un vector, es posible construir un operador distancia:

$$\mathrm{d}(x,y) = \|x-y\|_p = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p} \tag{2.18}$$

A continuación una rápida descripción de las p-normas más usadas.

Distancia Manhattan

También llamada distancia L_1, distancia taxi o distancia en bloques. Representa la suma de las diferencias en cada coordenada del vector que representa la cadena.

$$\mathrm{d}(x,y) = \sum_{i}^{n} |x_i - y_i| \tag{2.19}$$

Distancia Camberra

Variación de la distancia Manhattan (Lance & Williams, 1966; 1967) consistente en una suma ponderada de las coordenadas de los vectores.

$$\mathrm{d}(x,y) = \sum_{i}^{n} \frac{|x_i - y_i|}{|x_i| + |y_i|} \tag{2.20}$$

Distancia Euclídea

Distancia del espacio euclídeo que coincide con L_2.

$$\mathrm{d}(x,y) = \sqrt{\sum_{i}^{n} (x_i - y_i)^2} \tag{2.21}$$

Distancia Chebyshev

Distancia L_∞. La distancia entre dos vectores es igual a la distancia mayor entre dos coordenadas.

$$\mathrm{d}(x,y) = \mathrm{máx}(|x_i - y_i|) \tag{2.22}$$

*Coeficiente de coincidencia (*Matching Coefficient*)*

Es una medida de similitud que cuenta el número de coordenadas que son distintas de cero en ambos vectores.

$$\mathrm{sim}(x,y) = |x \,\&\, y| \tag{2.23}$$

Distancia coseno

Medida de similitud usando el coseno del ángulo que forman los dos vectores que se comparan. Para ello, utiliza la definición del producto escalar en el espacio euclídeo:

$$\text{sim}(x,y) = \frac{\sum_i^n x_i y_i}{\sqrt{(\sum_i^n x_i^2)(\sum_i^n y_i^2)}} \tag{2.24}$$

Conjuntos y probabilidad

Esta categoría se basa en transformar las cadenas en conjuntos de subcadenas de esta y calcular la similitud entre las cadenas comparando los conjuntos.

En general, los algoritmos no hacen una imposición en cómo se construyen estos conjuntos siendo lo más habitual escoger un parámetro n y construir los conjuntos como todos los n-gramas presentes en cada cadena.

Así la cadena s_1 =«tren» puede generar el conjunto

S_1 = [«t», «r», «e», «n»]

si se usan 1-gramas, o puede generar el conjunto:

S_1' = [«tr», «re», «en»]

si usa 2-gramas.

Una extensión aplicable a todos los algoritmos consiste en añadir al alfabeto dos símbolo especiales uno de inicio de cadena y otro de final de cadena. De forma que se pueden marcar n-gramas de inicio y final. Con esta extensión, los 3-gramas de la cadena s_1 serían:

S_1'' = [«^^t», «^tr», «tre», «ren», «en$», «n$$»]

Cuando se habla de una distancia n-grama o q-grama, se suele hablar del coeficiente Dice (que se describe a continuación) con esta extensión de marca de inicio y final.

Coeficiente Dice

Medida de similitud (Van Rijsbergen, 1979) que sólo se preocupa por los términos comunes que aparecen y no por su posición. En él, se calcula para cada cadena (s_i) el conjunto (S_i) de todas las subcadenas de longitud n que contengan. La similitud se define como:

$$\text{sim}(S_1,S_2) = \frac{2|S_1 \cap S_2|}{|S_1| + |S_2|} \tag{2.25}$$

Este planteamiento es similar al de Jaccard aunque usado sobre conjuntos y no sobre vectores. El valor de similitud también está acotado entre 0 (cadenas ortogonales) y 1 (cadenas equivalentes). Como se discutirá en la **sección 3.1.2**, no se cumple la identidad de los indiscernibles ya que existen cadenas que dan una similitud 1 cuando son diferentes.

Coeficiente Jaccard

Esta es la versión original de la similitud de Jaccard (Jaccard, 1912), también llamada *intersection over union* (intersección sobre unión).

$$\text{sim}(S_1,S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} = \frac{|S_1 \cap S_2|}{|S_1| + |S_2| - |S_1 \cap S_2|} \tag{2.26}$$

Este índice está acotado entre 0 y 1, siendo este último valor el de máxima similitud.

Coeficiente de solapamiento

Métrica destinada a indicar si una cadena es un subconjunto de otra (Gomaa & Fahmy, 2013).

$$\text{sim}(S_1,S_2) = \frac{|S_1 \cap S_2|}{\text{mín}(|S_1|,|S_2|)} \tag{2.27}$$

Si la similitud resultante es igual a 1, indica que una cadena solapa completamente con la otra siendo una subcadena. En caso de que devuelva 0, indica que no hay ninguna coincidencia entre los elementos de S_1 y S_2

Jaro

Desarrollada (Jaro, 1989) para buscar registros repetidos[13] en el censo, en el que una misma persona puede haber sido inscrita más de una vez con pequeñas variaciones de ortografía.

Se basa en dos conceptos:

- Un *token* de una cadena corresponde con otro de otra cadena si el símbolo coincide y la diferencia entre sus posiciones no es superior a

[2.13] Eliminar registros duplicados en una base de datos se conoce en la literatura como *record linkage*.

$$\left\lfloor \frac{\text{máx}(|s_1|,|s_2|)}{2} \right\rfloor - 1 \tag{2.28}$$

– Sea la cadena s'_1 aquella que solo contiene los tokens de s_1 (en el mismo orden que aparecen en s_1) que corresponden con otra cadena s_2.

  se denomina número de transposiciones ($T_{s'_1,s'_2}$) al menor número de tokens de s'_1 que hay que mover para que estén en el mismo orden que s'_2.

La similitud Jaro se define como:

$$\text{sim}(s_1,s_2) = \frac{1}{3}\left(\frac{|s'_1|}{|s_1|} + \frac{|s'_2|}{|s_2|} + \frac{|s'_1| - T_{s'_1,s'_2}}{|s'_1|}\right) \tag{2.29}$$

esta función está acotada entre [0,1]. Cuando son idénticas vale 1 y cuando son completamente distintas 0.

Existe la extensión de Winkler (Winkler, 1990) que le añade un peso extra a coincidencias al inicio de la cadena:

$$\text{sim}_\text{w} = \text{sim}_\text{j} + \ell\, p(1 - \text{sim}_\text{j}) \tag{2.30}$$

con

– sim_w y sim_j las semejanzas de Winkler y Jaro,
– ℓ la longitud del prefijo común (saturado en 4) y
– p es un factor de escala cuyo valor típico es 0.1.

Lamentablemente, aunque es utilizada esta extensión, no es una métrica ya que no cumple con la identidad de los indiscernibles ni con la desigualdad triangular.

Metamétricas

Por último, es posible construir métricas dependientes de otras métricas. Ya sea para combinar resultados de varias métricas como para refinar el resultado que proporcionado por una en concreto.

Sumas ponderadas

Sea un conjunto de métricas (d_i) y pesos (w_i) y sean x e y dos cadenas a comparar. Se puede definir una metamétrica como la suma ponderada de las métricas disponibles.

$$\text{d}_\text{meta}(s_1,s_2) = \frac{\sum_i w_i \text{d}_\text{i}(s_1,s_2)}{\sum_i w_i} \tag{2.31}$$

Si las métricas constituyentes (d_i) están acotadas, la metamétrica también lo estará. Si no importa el rango de valores de salida de la métrica, es posible eliminar el factor de normalización ya que el cambio de escala que produce no altera los requisitos de métrica.

Monge Elkan

El procedimiento Monge Elkan (Monge & Elkan, 1996), añade un nivel de complejidad a una distancia.

Se basa en dividir las cadenas s_1 y s_2 en listas de campos[14] o subcadenas (S_1 y S_2), y comparar cada campo de uno con el campo más similar del otro.

[14] Originalmente eran palabras: subcadenas divididas por un símbolo separador.

$$d_{monge}(S_1,S_2) = \frac{1}{|S_1|}\sum_{i}^{|S_1|}\min_{j}^{|S_2|} d(S_{1,i},S_{2,j}) \tag{2.32}$$

donde la función mín representa la máxima similaridad que en la práctica será un máximo o un mínimo numérico en función de que la métrica de la que se parte esté en la forma de distancia o de similaridad.

Originalmente, la propuesta inicial Monge Elkan era si dos descripciones referenciaban un mismo objeto (por ejemplo, dos direcciones postales) y usaba como métrica base Gotoh.

Lamentablemente, el cálculo de esta distancia implica comparar todas los campos de S_1 con los de S_2 por lo que no es muy eficiente.

Otras comparaciones entre cadenas que no son métricas

Existen otros algoritmos de comparación de cadenas que no devuelven un valor numérico de distancia o similitud sino que simplemente devuelven un valor biestado («similares» o «distintas») que no entra a determinar un grado de similitud entre cadenas. Este tipo de comparadores se emplean en el ya mencionado problema de enlace de registros (*record linkage*).

Este tipo de comparadores pueden construirse a partir de una métrica sin más que asignar un valor umbral y determinar si la similitud es o no superior a dicho umbral. En cualquier caso, existen métodos específicos de comparación como son Soundex (Pinto, et al., 2012) o Phonix (Gadd, 1990) centrados en determinar si dos apellidos suenan o no parecido.

Ambos sistemas funcionan gracias a un preprocesado antes de efectuar la comparación. El preprocesado consiste en la aplicación de distintas reglas sobre cada texto hasta llegar a una forma normalizada o firma (en el caso de Soundex es una letra seguida de 3 cifras). Si la forma normalizada de ambas cadenas coincide, ambas cadenas se consideran equivalentes o similares, si no coinciden se consideran distintas.

Dada la poca granularidad de las comparaciones, estas técnicas se utilizan en casos muy concretos como son similitud fonética de nombres o apellidos.

2.1.3 *Técnicas de Agrupación*

El *clustering* es un proceso no supervisado de etiquetado de puntos en un espacio métrico. Los puntos muy cercanos (o relacionados) en dicho espacio compartirán más etiquetas que puntos menos relacionados entre sí. Esta operación presenta dos finalidades (Tan, Steinbach, & Kumar, 2013): ayuda a comprender mejor los datos que se disponen asociando aquellos elementos con propiedades similares, y facilita posteriores etapas de procesado al proporcionar una reducción de información (de datos en crudo a etiquetas). Dada la naturaleza genérica de los algoritmos de *clustering*, las propiedades asignables a cada etiqueta dependerá fundamentalmente de la métrica escogida.

Dentro de las técnicas de agrupación, existen dos enfoques clásicos (Berkhin, 2006; Xu & Tian, 2015) de etiquetar los elementos.

Partición El proceso de agrupación por partición reparte el espacio en politopos y considera que dos puntos del espacio pertenecen al mismo grupo si están contenidos en el mismo subespacio (ejemplo en la **figura 2.2**).

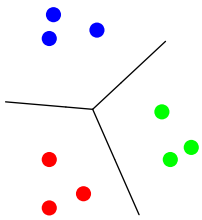


Figura 2.2: Agrupación por partición

Este procedimiento asigna una y solo una etiqueta a cada cadena.

$$\text{clustering}_{partición} :: [Cadena]_n \rightarrow [Tag]_n \qquad \text{(TIPO 2.33)}$$

Jerárquico (Ejemplo en la **figura 2.3**) El procedimiento jerárquico construye una estructura recursiva en árbol en el que cada grupo puede estar compuesto por puntos del espacio métrico y por subgrupos. A diferencia de la partición, cada punto recibe tantas etiquetas como a grupos pertenezca.

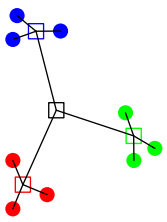


Figura 2.3: Agrupación jerárquica

$$\text{clustering}_{jerárquico} :: [Cadena]_n \rightarrow [[Tag]]_n$$

(TIPO 2.34)

El etiquetado de las distintas cadenas nos permite establecer, tras la agrupación, una función de relación o similitud entre cadenas que tenga menos sensibilidad que una métrica aplicada directamente. Dado que la función de etiquetado devuelve para cada cadena un conjunto de etiquetas, es posible determinar la similitud entre cadenas empleando cualquier métrica basada en conjuntos.

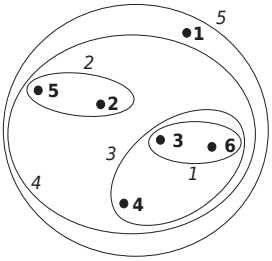
La mencionada función de relación tendría el tipo

$$\text{relacion} :: [[Tag]] \rightarrow [[Tag]] \rightarrow Float \in [0,1]$$

(TIPO 2.35)

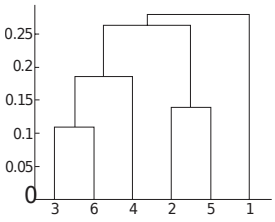
y una posible implementación sería usar el coeficiente Dice (**ecuación 2.25**):

$$\text{relacion}(x,y) = \frac{2 \cdot |\text{tags}(x) \cap \text{tags}(y)|}{|\text{tags}(x)| + |\text{tags}(y)|} \tag{2.36}$$



(a) Group average clustering.

Figura 2.4: Figura 8.18(a) tomada de (Tan, Steinbach, & Kumar, 2013)



(b) Group average dendrogram.

Figura 2.5: Figura 8.18(b) tomada de (Tan, Steinbach, & Kumar, 2013)

Los valores posibles en la imagen de la función dependen del tipo de agrupación efectuada. En el caso de una partición, dos cadenas pueden compartir la etiqueta (dado que cada cadena solo se le asigna una etiqueta) o no. En cuyo caso, la relación entre ellas será total (1) o nula (0) sin posibilidad de valores intermedios.

Distinto es el caso de una agrupación jerárquica en el que la relación determinada entre dos cadenas puede contener distintos valores entre 0 y 1 en función del número de etiquetas comunes y etiquetas asignadas.

La relación entre cadenas es una estimación cualitativa de grupos en común que no entra en consideración de diferencias o similitudes entre los grupos. La agrupación jerárquica provee, además, de una medida cuantitativa que expresa cómo de parecidos son todos los elementos con un ancestro común.

En la **figura 2.4** se muestra una serie de puntos que han sido agrupados de forma jerárquica. Esta agrupación puede verse, de forma simplificada, en la **figura 2.5** en un gráfico denominado «dendograma». En el dendograma se organizan los puntos en el eje de las abscisas. Las líneas verticales representan distancias y las líneas horizontales identifican los «clados» (elementos que

tienen un ancestro común). La altura de la línea de clado indica la diferencia entre los miembros de este. Un clado muy bajo, indica elementos muy parecidos entre sí y un clado alto, indica elementos poco relacionados.

En las figuras empleadas como ejemplo, los puntos 3 y 6 forman un grupo cuya diferencia entre sus elementos es de 0,1 y los puntos 3, 6 y 4 presentan una diferencia de casi 0,2. De todos los puntos mostrados, el punto 1 es el más alejado del resto.

En cualquier caso, tanto la relación entre elementos como la diferencia del clado hay que interpretarlas como una medida de distancia en un orden superior a la mera distancia entre elementos.

Clustering por partición

Existen diversos criterios de división del espacio basados en distribuciones de probabilidad (McLachlan & Basford, 1988), cercanía a puntos singulares (Hartigan & Wong, 1979; Kaufman & Rousseeuw, 2009), densidad de puntos (Miller & Han, 2001) o por partición del espacio (Schikuta & Erhart, 1997). En general funcionan por refinamiento progresivo en el que se parte de una solución tomada al azar y en sucesivas iteraciones va convergiendo a una solución estable.

Salvo el enfoque de partición del espacio, los otros procedimientos enunciados se basan en etiquetar los puntos considerados. Si se deseara tener una partición real y completa del espacio es necesario definir una distancia al *cluster* (veremos distintas técnicas cuando se hable de las agrupaciones jerárquicas) y a partir de ella hacer un mapa de Voronoi del espacio.

A continuación, se describirán brevemente los principios en los que se basan.

Modelos Probabilísticos

Este enfoque considera que cada categoría tiene asociada una distribución de probabilidad en el espacio y considera todo el espacio como la suma de las distribuciones de todas las categorías. Se impone que cada punto analizado pertenece a una y solo una categoría y se considera como un muestreo sobre la mezcla de distribuciones.

El agrupamiento probabilístico busca encontrar las distribuciones que mejor ajusten con la distribución de puntos que se analizan, por lo que convierte el problema de agrupamiento en

un problema de optimización cuya función objetivo es el logaritmo de la función verosimilitud cuyo valor es:

$$\mathrm{L}(X|C) = \prod_{i=1:N} \sum_{j=1:k} \tau_j \Pr(x_i|C_j) \tag{2.37}$$

donde N es el número de puntos analizados, k es el número de grupos que construir, τ_j la probabilidad asociada a que un punto se muestree del modelo j y $\Pr(x_i|C_j)$ la probabilidad de que el punto x_i pertenezca a la categoría j.

Basados en un punto singular: k-mean, k-medoid

El procedimiento *k*-mean es uno de los más usados en la industria (Berkhin, 2006) (dada su simplicidad) y el principio del que parte es la representación de cada grupo por un único punto.

El **algoritmo 2.1** muestra los pasos generales de cálculo del *k*-mean según el algoritmo de Lloyd (Faber, 1994).

Seleccionar k puntos como centroides iniciales.
repeat
  Asignar cada punto al centroide más cercano.
  Recalcular el centroide de cada cluster.
until los centroides calculados no varíen.

Algoritmo 2.1: k-mean. Formulación de Lloyd

El algoritmo de *k*-means presenta algunos inconvenientes que dificulta su uso directo. Problemas que se agravan, sobre todo, en espacios de altas dimensiones:

— **Cálculo del centroide** el cálculo del centroide no es trivial[15] para todo tipo de estructuras. En esos problemas, el algoritmo empleado se sustituye por *k*-medoids (Ng & Han, 2002) en el que el punto singular ya no es la media de los puntos del cluster sino que se fuerza que el punto singular sea un punto del cluster (cercano a la media).

— **Forma de los grupos** *k*-mean funciona bien cuando los grupos naturales a identificar tienen forma hiper-esférica. Pero fallan al intentar discriminar grupos en los que existe cierto solapamiento en las envolventes.

La **figura 2.6** muestra algunos ejemplos sintéticos de agrupación usando *k*-mean, en el que se muestra la posición del centroide con un círculo con borde negro. Se puede apreciar

[15] Una de las aportaciones de esta tesis es la construcción de centroides para cadenas.

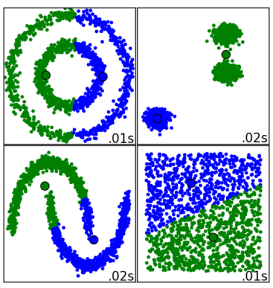


Figura 2.6: Ejemplos de agrupamiento *k*-mean tomado de (Pedregosa, et al., 2011)

el fallo en superficies con concavidades y solapamiento de envolventes.

— **Calidad de los *clusters*** El algoritmo identifica el número de grupos indicados, tengan sentido o no. Los ejemplos de la derecha de la **figura 2.6** sufren de dicho problema en uno tendría, a priori, más sentido tener tres grupos en vez de dos y en el otro tendría más sentido tener un único grupo.

— **Repetibilidad** El resultado obtenido depende, en gran medida, de las posiciones de los centroides iniciales. No hay garantía de convergencia, y cuando esta se produce no hay garantía de partición óptima.

Con intención de minimizar los inconvenientes expuestos, numerosas extensiones se han propuesto (Jain, 2010) aunque no pueden batir la sencillez de uso que hacen de *k*-mean de una herramienta sin parangón para el análisis y visualización inicial de los datos.

Basado en densidad

La idea principal de estas técnicas se basa en considerar que cada punto está conectado con los puntos de su vecindad inmedidata (puntos cuya distancia al primero no supera un umbral determinado) y que el etiquetado de ese punto está fuertemente relacionado con el etiquetado de los puntos de vecindad. Todos los puntos del espacio en un área de densa de conexiones pertenecerán, por tanto, al mismo *cluster*.

El proceso de general de particionado comienza en un punto al que se le asigna una etiqueta y dicha etiqueta se va extendiendo a los puntos del vecindario siempre que se mantenga una densidad de conexión mínima. Una vez que la región no puede crecer más, se escoge un nuevo punto no etiquetado y se repite el proceso hasta que no queden puntos sin identificar.

Estos algoritmos son capaces de marcar regiones de formas arbitrarias y, por su construcción, son inmunes a *outliers*. Además, presentan una buena relación tiempo de cálculo tamaño de la nube de puntos, por lo que presentan buena escalabilidad.

Lamentablemente, también presentan una serie de inconvenientes a considerar.

— **No es sensible a variaciones de densidad** Dos regiones adyacentes con densidades significativamente distintas (pero

superiores al umbral escogido) se identificarán como un único *cluster*.

— **Dificultad de interpretación de resultados** Cuando aumenta el número de dimensiones presentes en los objetos, más difícil de interpretar los resultados obtenidos.

— **El número de grupos no es controlable** Las variables de control de los algoritmos están relacionadas con umbrales de densidad y definición de vecindad.

El algoritmo DBSCAN (*Density Based Spatial Clustering of Applications with Noise*) (Ester, Kriegel, Sander, Xu, et al., 1996) es el más conocido de esta categoría (estando orientado a espacios de pocas dimensiones). Se basa en dos parámetros ε y *MinPts*, y las siguientes definiciones:

– Un ε-vecindario de x es el conjunto:

$$N_\varepsilon = \{y \in X | \mathrm{d}(x,y) \leq \varepsilon\}$$

– Un núcleo es un punto con un vecindario que contiene más de *MinPts*.

– y es denso-alcanzable (*density-reachable*) desde el núcleo x si existe una secuencia finita de núcleos entre x e y tal que cada núcleo pertenezca al vecindario del anterior

– La denso-conectividad (*density-connectivity*) de dos puntos x e y es cuando ambos puntos son denso-alcanzables desde un mismo núcleo. La denso-conectividad es simétrica.

Los pasos básicos para efectuar una agrupación DBSCAN se describe en el **algoritmo 2.2**

Todos los puntos denso-conectados forman un *cluster* en los que los núcleos representan los puntos interiores, los puntos que no son núcleos están situados en el borde y todo punto que no pertenece al vecindario de un núcleo es un *outlier*. La **figura 2.7** muestra el desempeño en las mismas distribuciones que se emplearon con el algoritmo *k*-mean.

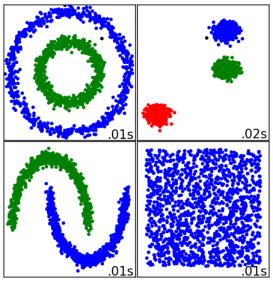


Figura 2.7: Ejemplos de agrupamiento DBSCAN tomado de (Pedregosa, et al., 2011)

Una dificultad de aplicación de este algoritmo es la selección de los parámetros ε y *MinPts*. No existe una regla o norma que asocie los datos a analizar con los parámetros que implica que parámetros con valores generosos (entornos grandes y conectividad baja) tienen a construir un único grupo con todos los datos y parámetros restrictivos a considerar que todos los puntos son *outliers*.

```
idcluster ← 0
while hay puntos sin etiquetar do
  punto_1 ← punto no etiquetado al azar
  if punto_1 no es nucleo then
    tag(punto_1) ← outlier
  else
    Añadir punto_1 a cola.
    while cola no está vacía do
      Extraer primero de la cola en punto_2
      tag(punto_2) ← idcluster
      Añadir a cola todos los puntos del vecindario de punto_2
      no etiquetados.
    end while
    idcluster ← idcluster + 1
  end if
end while
```

Algoritmo 2.2: Agrupación DBSCAN

Con el ánimo de minimizar este problema se han buscado extensiones como OPTICS (*Ordering Points To Identify the Clustering Structure*) (Ankerst, Breunig, Kriegel, & Sander, 1999) en la que se consideran que ε es una variable y estudia el comportamiento del agrupamiento para todos los valores $\varepsilon' \leq \varepsilon$, o DBCLASD (*Distribution Based Clustering of Large Spatial Databases*) (Xu, Ester, Kriegel, & Sander, 1998) que considera que cada *cluster* tiene una densidad de puntos y utiliza una distribución de probabilidad para estimar la densidad y los grupos.

Clustering jerárquico

Los métodos de construcción de una agrupación jerárquica se clasifican en agregativos (enfoque *bottom-up*) y divisivos (enfoque *top-down*)[16]. En los primeros se parte de los puntos originales que en sucesivas pasadas se van fusionando con otros (creando grupos) y estos entre sí hasta llegar a uno único. En los segundos métodos, se parte de un único grupo que engloba todos los puntos y se van subdividiendo hasta llegar al nivel de granularidad deseado.

[2.16] (Jain & Dubes, 1988)

Una gran ventaja de estas técnicas de agrupamiento es que funcionan a partir de una matriz de distancias por lo que pueden operar con cualquier tipo de objetos que tengan definida una métrica. Igualmente y dado que el árbol de *clusters* (que como se ha indicado anteriormente también se llama dendograma) es una

instantánea de todos los grupos y subgrupos generados, permite decidir a posteriori particiones excluyentes de los puntos.

Al ser un proceso determinista, los árboles generados siempre serán idénticos siempre que los puntos analizados y el método escogido de construcción también lo sean.

Agrupación aditiva

Para realizar una agrupación jerárquica agregativa, es necesario obtener previamente una matriz de distancias en el que cada elemento $D_{i,j}$ sea la distancia entre el elemento i y el elemento j del conjunto de datos. Esta matriz nos permite determinar cómo hacer la agrupación tal y cómo se muestra en el **algoritmo 2.3**.

Establecer matriz D de distancias entre elementos.
repeat
  Seleccionar elemento $D_{i,j}$ menor.
  Crear el agregado ij
  Eliminar en D las distancias de los elementos i y j
  Añadir las distancias de los elementos restantes de D con el agregado ij
until D tenga un solo elemento.

Algoritmo 2.3: Agrupación jerárquica aditiva

Para completar la descripción de la agrupación jerárquica, es necesario especificar cómo se van a construir las distancias del nuevo agregado ij con cada elemento (k) o agregado (kl) restante del conjunto. Este criterio de distancias entre clados se denomina *linkage* (enlaces). Entre otras estrategias[17] de enlace, están:

[17] (Murtagh, 1983; Olson, 1995)

— ***single link*** La distancia entre dos clados se define como la mínima existente entre los miembros de estos.

$$\text{single}(C_1,C_2) = \min \mathrm{d}(x,y) \ \forall x \in C_1, y \in C_2 \quad (2.38)$$

— ***average link*** La distancia entre clados se define como la media de las distancias entre cada elemento de un clado con cada elemento del otro.

$$\text{average}(C_1,C_2) = \frac{\sum_{x \in C_1} \sum_{y \in C_2} \mathrm{d}(x,y)}{|C_1| \cdot |C_2|} \quad (2.39)$$

— ***complete link*** La distancia entre clados es la máxima distancia entre los puntos de uno y otro.

$$\text{complete}(C_1,C_2) = \max \mathrm{d}(x,y)\ \forall x \in C_1, y \in C_2 \quad (2.40)$$

El cálculo de las distancias a los nuevos agregados sería computacionalmente gravoso si no fuera por la fórmula de actualización Lance-Williams (Lance & Williams, 1967) que permite calcular las distancias de los agregados[18] en función de las distancias previamente calculadas antes de la unión:

$$\mathrm{d}(C_i \cup C_j, C_k) = \alpha_1 \mathrm{d}(C_i,C_k) + \alpha_2 \mathrm{d}(C_i,C_k) + \beta \mathrm{d}(C_i,C_j) + \\ + \gamma |\mathrm{d}(C_i,C_k) - \mathrm{d}(C_j,C_k)| \quad (2.41)$$

Donde las constantes α_1, α_2, β y γ están definidas con valores distintos para cada tipo de enlace.

[18] En el diagrama del dendograma, la altura del clado o agrupación ij corresponde con la distancia $\mathrm{d}(C_i,C_j)$.

Agrupación divisiva

Empleada en lingüística y agrupación de documentos, las divisiones binarias han demostrado ser muy útiles (Steinbach, Ertöz, & Kumar, 2004; Steinbach, Karypis, Kumar, et al., 2000).

El procedimiento PDDP (*Principal Direction Divisive Partitioning*)[19] presenta un procedimiento de agrupación divisiva respondiendo a dos preguntas: cómo dividir el conjunto, y qué conjunto dividir.

[19] (Littau & Boley, 2006)

Respecto a la primera cuestión, en el caso de agrupación por división binaria de documentos, se parte de N documentos representados por un vector x en el que el componente l es la frecuencia de aparición de la palabra en la posición l de una lista de términos. La matriz de frecuencias X es entonces normalizada para que cada coordenada tenga su media en 0. Esto es, la nueva matriz C se construye como

$$C = (X - e\bar{x}),\ \bar{x} = \frac{1}{N} \sum_{i=1:N} x_i,\ e = (1,\cdots,1)^T \in \mathbb{R}^d \quad (2.42)$$

La bisección de los documentos se hace con un hiperplano que pasa por el centroide de X y con dirección normal el autovector asociado al valor singular mayor de la matriz de valores singulares (SVD, *Singular Value Descomposition*) de C.

Sobre la cuestión de qué conjunto dividir, no hay una opinión tan clara y las opciones más comunes son:

– Dividir cada subgrupo un mismo número de niveles.

– Dividir el subgrupo con mayor cardinalidad (y repetir un número determinado de veces).

– Dividir el subgrupo que presente la mayor varianza inter grupo.

Todas las propuestas tienen problemas y se remite a (Savaresi, Boley, Bittanti, & Gazzaniga, 2002) para un análisis más profundo.

2.1.4 *Descriptores*

Llamamos descriptor o característica (en inglés *feature*) a toda función que aplicada a una cadena, devuelve un valor[20].

$$Descriptor :: Cadena \rightarrow Valor \qquad \text{(TIPO 2.43)}$$

[2.20] Por simplicidad y cuando no haya confusión, también llamaremos descriptor (de la cadena s) al valor devuelto por dicha función al aplicarla a la cadena s.

En principio, un *Valor* es un escalar; pero los sistemas de reconocimiento de patrones aceptan otras formas de datos que, de forma transparente para el usuario, transforman en una serie de descriptores escalares. Así un descriptor que devuelva un par (una 2-tupla) se puede descomponer en 2 sub-descriptores que devuelvan cada término de la tupla. Análogamente, una variable nominal (que caracteriza la cadena con una palabra de entre una lista de N) puede sustituirse por N descriptores booleanos que indiquen si la palabra escogida es la palabra i de la lista.

Es posible definir un espacio de características (*feature space*) a partir de una lista $F = \{f_1, f_2, \cdots, f_n\}$ de descriptores por el que una cadena s se convierte en una n-tupla $x = (f_1(s), f_2(s), \cdots, f_n(s))$ que es procesable por los algoritmos de reconocimiento de patrones discriminantes.

Como ya se ha comentado en la introducción del capítulo, la transformación al espacio de características convierte objetos de formas variables en formas normalizadas de tamaño definido en el que cada descriptor funciona como una proyección de la cadena u objeto en el espacio definido.

A la hora de los datos en el espacio características aparece un conflicto de intereses. Un espacio multidimensional rico aporta más información sobre los objetos que representa que uno con menos dimensiones. Desarrollar un listado elevado de descriptores facilita una mejor identificación de elementos diferenciadores o discriminantes en el corpus analizado. Por otro lado, los

algoritmos de análisis discriminantes son computacionalmente complejos.

Un ejemplo de ello: el entrenamiento perfecto de una red neuronal de 3 neuronas (**figura 2.8**) es un problema NP-completo (Blum & Rivest, 1993). Aplicando ciertas optimizaciones de cálculo (Livni, Shalev-Shwartz, & Shamir, 2014) que sacrifican precisión, entrenar esta red tiene una complejidad $O(n^2)$ (con n el número de entradas de la red que coincide con las dimensiones del espacio de características).

Como puede verse, existe un conflicto de intereses a la hora de escoger el número de descriptores a usar. Un número elevado de estas permiten tener descripciones ricas de los objetos y más posibilidades de encontrar características discriminantes; pero también aumenta el tiempo de cómputo del análisis hasta límites en los que no es práctico su uso.

En el proceso de crear un listado adecuado de descriptores se identifican tres tareas complementarias.

— **Ingeniería de características (*Feature Engineering*)** Es el proceso de construcción de los descriptores que proyectan los objetos en el espacio de características. El objetivo de esta tarea consiste en determinar qué características de los objetos hacen que sean más fácilmente identificables por los algoritmos de reconocimiento de patrones.

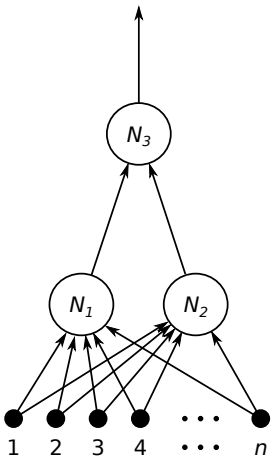


Figura 2.8: Red neuronal de 3 neuronas (Blum & Rivest, 1993)

Las características extraídas están en el ámbito de magnitudes físicas (como pueden ser el color, peso, material...), estructurales (relacionadas con la forma, relación de elementos entre sí...) o matemáticos (valores medios, varianza, máximos y mínimos...)[21].

[21] (Friedman & Kandel, 1999)

Esta tarea es compleja y requiere de expertos en el campo de estudio para fijar las características consideradas más relevantes. Adicionalmente, hay un auge en sistemas de aprendizaje que intentan identificar estas características (como (Severyn & Moschitti, 2013) o (Pezeshk & Tutwiler, 2011)). Aunque son sistemas tan centrados en un campo del conocimiento que son los desarrolladores los que han incorporado su *know-how* de ingeniería al sistema, por lo que no pueden considerarse una solución general al problema de los descriptores.

En general, esta tarea está muy castigada en la literatura debido a su gran especificidad del problema concreto que se analiza. No es extraño encontrarse en artículos sobre gestión de descriptores con citas como «Do you have domain knowledge? If yes, construct a better set of “ad hoc” features.» (Guyon & Elisseeff,

2003) o «However, it is impossible to develop general guidelines how to select physical or structural features.» (Friedman & Kandel, 1999) y no volver a hablar del tema pasando a otros menesteres. En el **capítulo 3** se ha hecho un esfuerzo importante en describir descriptores para cadenas de propósito general.

— **Extracción de características (*Feature extraction*)** Procedimientos de cálculos aplicados a descriptores para construir nuevos descriptores como combinación de los anteriores. La extracción de características proporciona nuevos descriptores (aumentando por tanto la dimensionalidad del problema) que expliquen mejor los datos[22].

[2.22] La extracción de características es una herramienta de incremento de la dimensionalidad del problema. Es usual, tras la creación de una nueva característica, eliminar los descriptores originales que se emplearon en la construcción. En cuyo caso, el balance neto es de reducción dimensional.

Podemos decir que el proceso de extracción de características es un mapeo de un espacio de características original a un nuevo espacio de características de menor tamaño.

Tal y como se están enunciando esta clasificación de generadores de descriptores, existe una sutil diferencia entre la ingeniería de características y la extracción de características. Los primeros construyen descriptores a partir de los datos disponibles, mientras que los segundos actúan exclusivamente sobre descriptores. Esta diferencia permite separar, conceptualmente, la construcción inicial de descriptores (que a todas luces es imprescindible) de la generación de nuevos descriptores derivados que algunos investigadores rechazan.

Los detractores de estas técnicas, como (Guyon & Elisseeff, 2003; Tang, Alelyani, & Liu, 2014), achacan que el remapeo entre espacios provoca una desconexión del proceso físico analizado con los resultados obtenidos y esto reduce la capacidad de interpretación de los resultados y su relación con los fenómenos originales.

La diferencia entre la ingeniería y la extracción de características es, como decimos, reducida y no es extraño ver investigadores (Friedman & Kandel, 1999) que no distinguen entre ambas categorías de técnicas.

— **Selección de características (*Feature Selection*)** La selección de características es otra técnica de reducción de la dimensionalidad que pretende escoger un subconjunto adecuado de características que elimine redundancias presentes.

Si la extracción de características funcionaba como un cambio de espacio, la selección de características funciona como la proyección de las características en un subespacio de dimensiones reducidas.

A continuación se ven estas técnicas con un poco más de detalle.

Ingeniería de Características

Como se ha comentado anteriormente, este campo es muy específico del problema que se está analizando. En un intento de organizar los descriptores obtenidos en esta fase, se propone la siguiente clasificación[23]:

[2.23] Modificación sobre la efectuada por (Friedman & Kandel, 1999)

– Físicas o magnitudes directamente tomadas de los datos disponibles. Estos datos pueden representar datos capturados con sensores o directamente introducidos en el sistema. Suelen representarse por valores escalares o cualitativos.

  Las características registradas sobre las magnitudes directas, pueden sufrir un proceso de preprocesado en el que se le dé una forma más apropiada para su posterior análisis.

– Estructurales. Estos valores representan relaciones entre elementos del objeto analizado como puede ser las propiedades geométricas (la forma de un polígono no depende de la posición de un punto sino de la posición relativa de dicho punto respecto a los demás).

– Matemático. Valores aglutinantes tomados sobre series de datos. A diferencia de las características estructurales que consideramos específicas del ámbito de estudio, los descriptores matemáticos son de aplicación genérica en distintos campos.

– *Kernels*. Un *kernel* es una función que es capaz de calcular el producto escalar (en el espacio de descriptores) de dos objetos sin necesidad de calcular previamente los vectores en el espacio de características.

Preprocesado

Una vez escogido las magnitudes físicas y estructurales, es útil efectuar un preprocesado que ayude a unificar la forma de las características. Existen distintas técnicas de preparación de datos que se pueden aplicar dependiendo de la naturaleza de los datos que reflejan. Algunos de estos procesos[24] son:

[2.24] (Guyon & Elisseeff, 2006)

— **Estandarización** Las características pueden tener distintas escalas (aunque puedan referirse a magnitudes comparables) por lo que puede ser recomendable cambiar la escala antes de

trabajar con ellas. La fórmula de centrado y cambio de escala más clásica es:

$$x'_i = \frac{x_i - \mu_j}{\sigma_j} \tag{2.44}$$

donde x_i son las distintas observaciones registradas de la característica j del vector de características. μ_j y σ_j son, consecuentemente, la media y la desviación estándar de la característica j a lo largo de los datos de trabajo.

— **Normalización** Reducir el rango de valores de la característica a un rango fijo (normalmente [0,1]).

$$x'_i = \frac{x_i - \text{mín}_j}{\text{máx}_j - \text{mín}_j} \tag{2.45}$$

Empleando la misma nomenclatura que en la estandarización, mín_j (máx_j) es el valor mínimo (máximo) registrado en la característica j.

— **Aumento relación señal-ruido** Obtenible al aplicar un filtro sobre los datos. Entre otras operaciones se pueden citar[25] borrado del fondo, *de-noising*, *smoothing* y *sharpening* (Antoniou, 2016). Para ello se emplean transformadas de Fourier o *Wavelet* (Mastriani, 2016; Walker, 2008).

[25] Algunas de estas técnicas son nombradas en inglés al ser los términos usados habitualmente en el campo del tratamiento de señales.

— **Discretización de las características** Algunos algoritmos no manejan bien datos con valores continuos. Trabajar con esta reducción de información no solo facilita el uso de ciertos algoritmos, sino que puede simplificar la descripción de los datos y la comprensión de los datos. (Liu & Motoda, 1998)

— **Emborronado de datos** Aunque no se trabaje específicamente con lógica borrosa (Kahraman, Kaymak, & Yazici, 2016; Zadeh, 1965), las funciones de pertenencia son válidas como descriptores.

— **Filtrado de *outliers*** Tras el escalado, es posible detectar puntos fuera del rango de valores normales. Estos puntos pueden ser sustituidos empleando algún descriptor estadístico de la característica (como el valor medio o la moda) o descartar completamente el vector que tenga presente *outliers* por no ser fiables.

— **Coerción de longitud** Cuando las características escogidas provienen directamente de un proceso de muestreado, es posible que distintos sucesos registrados tengan distintas duraciones. Dado que el espacio de características es un espacio n

dimensional (con n fijo), es necesario que todas las observaciones tengan exactamente el mismo tamaño.

Algunas técnicas de coerción de longitud son:

- Trucamiento inicial o final. Tomando los primeros (o últimos) n elementos registrados.
- Enventanado. Sustitución de una muestra larga por una serie de ventanas de tamaño n con cierto solapamiento.
- *Zero-padding*. Completar con ceros aquellas muestras más cortas del tamaño fijado.

Estructurales

Los descriptores estructurales son muy dependientes del problema con el que se trabaje y su principal característica es que dependen de un conjunto de datos relacionados entre si. Así, dentro del campo de procesado de imágenes, los descriptores SIFT (Lowe, 2004), SURF (Bay, Ess, Tuytelaars, & Van Gool, 2008) u ORB (Rublee, Rabaud, Konolige, & Bradski, 2011) determinan puntos singulares en una imagen a partir de examinar ciertos puntos y su entorno.

Solo por citar otro campo, dentro del área de análisis de señales de audio podemos hablar (por ejemplo) de descriptores MPEG-7 (Luque, Larios, Personal, Barbancho, & León, 2016).

Matemáticos

Es posible determinar ciertas descriptores estadísticos directamente calculados a partir de los datos. El uso de unos u otros dependerá de la naturaleza de los datos analizados[26]. Basándonos en la clasificación que hace (Esmael, Arnaout, Fruhwirth, & Thonhauser, 2015), establecemos las siguientes categorías de descriptores matemáticos:

- Medidas de tendencia Central: media, mediana y moda.
- Medidas de variabilidad: varianza, desviación típica, Rango.
- Medidas de forma: *skewness*, *kurtosis*, segundo momento.
- Medidas de frecuencia: primer orden, segundo orden.
- Medidas de posición: percentil.
- Medidas de Impureza: entropía.

[2.26] El valor medio de una característica tiene sentido en valores escalares. Descriptores relacionados con la forma dependen de cómo se ordenen los datos previamente (por ejemplo usando el orden de captura de los datos o si se reordenan en función del valor numéricos).

A continuación, describimos algunos de estos descriptores. Para ello, vamos a considerar que tenemos una serie de capturas temporales y que trabajamos con ventanas de m elementos. Esto es, el objeto que estamos considerando es una captura de m elementos muestreados de forma sucesiva $x = [x_1, x_2, \cdots, x_m]$.

Media

$$\mu = \frac{1}{m} \sum_{i=1}^{m} x_i \tag{2.46}$$

Desviación estándar

$$\sigma = \sqrt{\frac{1}{m} \sum_{i=1}^{m} (x_i - \mu)^2} \tag{2.47}$$

Momento estandarizado El momento k estandarizado es la expresión

$$\frac{\mu_k}{\sigma^k} \tag{2.48}$$

donde

$$\mu_k = \frac{1}{m} \sum_{i=1}^{m} (x_i - \mu)^k, \qquad \sigma^k = \left(\sqrt{\frac{1}{m} \sum_{i=1}^{m} (x_i - \mu)^2} \right)^k \tag{2.49}$$

y permiten calcular el *Skewness* y la *Kurtosis*.

Kurtosis Cuarto momento estandarizado y está relacionado con distribuciones con colas cortas y alta concentración de valores alrededor de la media.

$$Ku = \frac{\mu_4}{\sigma^4} \tag{2.50}$$

Skewness Tercer momento estandarizado que está relacionado con la simetría de la señal o, visto de otro modo, con la comparación del tamaño de las colas de las señal.

$$Sk = \frac{\mu_3}{\sigma^3} \tag{2.51}$$

Frecuencia 1$^{\text{er}}$ orden Frecuencia de aparición de un valor α en la muestra. Solo es aplicable en valores discretos por lo que puede ser necesario cuantizar previamente las variables continuas.

$$\mathrm{h}(\alpha) = \sum_{i=1}^{m} \begin{cases} 1 & \text{si } x_i = \alpha \\ 0 & \text{resto} \end{cases} \tag{2.52}$$

Frecuencia orden *n* Frecuencia de aparición de una secuencia contigua de n valores discretos en la muestra.

$$\mathrm{h}(\alpha_1,\alpha_2,\cdots,\alpha_n) = \sum_{i=1}^{m-(n-1)} \begin{cases} 1 & \text{si } x_i = \alpha_1, x_{i+1} = \alpha_2, \ldots x_{i+n-1} = \alpha_n \\ 0 & \text{resto} \end{cases} \tag{2.53}$$

Posición Un percentil es el valor de una variable que tiene un cierto porcentaje de observaciones con valor inferior al dado. Así, el percentil 10 ($p10$) es el valor que es superior al 10% del total de observaciones consideradas.

El cálculo de varios valores de percentil (típicamente $p25$, $p50$ y $p75$ pudiéndose complementar con $p10$, $p90$ y con $p5$ y $p95$) permite hacer un esbozo de la función de distribución acumulada (CDF, *Cumulative Distribution Function*).

Entropía Si se tiene una distribución de probabilidad de aparición de los valores considerados ($\mathrm{P}(X)$), es posible determinar la entropía o impureza asociada a la muestra tomada. Para variables discretas la entropía vale:

$$\mathrm{H}(x_1,x_2,\cdots,x_m) = -\sum_{i=1}^{m} \mathrm{P}(x_i) \log_2 \mathrm{P}(x_i) \tag{2.54}$$

Otros estadísticos Otros estadísticos son usables como son los valores máximos, los mínimos, la longitud, el primer elemento, el último, etc.

Kernels

El enfoque proporcionado por los *kernels* es distinto al de los anteriores técnicas sobre descriptores[27].

Existen una serie de algoritmos de aprendizaje que pueden adaptarse para que no requieran conocer el vector de características de cada objeto u observación. Esta adaptaciones se basan en trabajar sobre el producto escalar de los vectores de características (que funciona como una comparación entre objetos) en vez de trabajar directamente sobre los vectores. Entre estos algoritmos, quizás el más famoso es la máquina de vector soporte de vectores (SVM, *Support Vector Machine*[28]) aunque existen otros algoritmos que soportan esta adaptación como son el perceptrón, el análisis de componentes principales (PCA, *Principal Component Analysis*), *Nearest Neighbour* y otros (Lodhi, Saunders, Shawe-Taylor, Cristianini, & Watkins, 2002b).

[27] De hecho, no es estrictamente hablando una herramienta de gestión de descriptores. Más bien, es una herramienta para ahorrarse el uso de estos.

[28] (Arredondo Arteaga, Gil González, Flórez, & José, 2017)

Un *kernel* no es más que una función que es capaz de calcular el producto escalar de dos vectores de características partiendo de los objetos originales y sin necesidad de calcular dichos vectores. Un *kernel* eficiente permite, por tanto, aplicar los algoritmos anteriormente citados ahorrándose los pasos de ingeniería, extracción y selección de características (al coste de encontrar un *kernel* apropiado al problema que se está trabajando).

La definición de producto escalar de dos vectores a y b es:

$$a \cdot b = \sum_i a_i b_i \tag{2.55}$$

Si estamos en $\mathbb{R}^2$ y tenemos dos objetos expresados en forma exponencial $a = \rho_a e^{j\varphi_a}$ y $b = \rho_b e^{j\varphi_b}$ es posible construir un kernel que devuelva el producto escalar de a y b sin necesidad de calcular los componentes a_i y b_i:

$$\text{kernel}(a,b) = \rho_a \rho_b \cos(\varphi_a - \varphi_b) = a \cdot b \tag{2.56}$$

Extracción de Características

La extracción de características es una herramienta de construcción de nuevos descriptores a partir de descriptores antiguos. Generalmente se emplea como método de reducción dimensional (ya que los nuevos descriptores sustituyen a los empleados en su construcción), aunque en sistemas complejos pueden usarse para aumentar la capacidad descriptiva de los vectores de características aumentando las dimensiones del espacio y así mejorar sus resultados.

La reducción dimensional que proporciona la extracción de características permite, aparte de una simplificación en el análisis de datos, generar visualizaciones de los datos que resuman todas las características consideradas con nada más que forzar una reducción a 2 o 3 dimensiones. Esta ventaja no tiene equivalente en la selección de características cuyas reducciones a 2 dimensiones sólo mostrarían las dos características más relevantes identificadas.

Algunas de las técnicas más usadas de extracción de características son:

— **Expansión polinomial** (*Polynomial Expansion*) es una técnica de expansión dimensional que permite a algoritmos de análisis lineales ser sensibles a efectos no lineales (Gergonne, 1974; Nystrom & Hughes, 2016) o de interacción entre características.

En función de la naturaleza del problema, se puede añadir al conjunto de características derivados de un descriptor (x) como son x^n (con $n = 2,3,4,\cdots$), $1/x$, $log(x)$, etc.

Así, en un sistema de regresión lineal en un espacio de una característica x, la salida del sistema sería

$$y(x) = \alpha_0 + \alpha_1 x \tag{2.57}$$

efectuando una expansión se pueden construir los descriptores x^2, x^3, etc. determinando (con el mismo algoritmo de regresión lineal) una expresión

$$y(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \alpha_3 x^3 + \cdots \tag{2.58}$$

que ya no es lineal.

Análogamente, es posible construir nuevos descriptores que muestren la interacción entre dos o más elementos como efectos cruzados ($x \cdot y$ o x/y).

— **Análisis de componentes principales (PCA)** El análisis por componentes principales (Bro & Smilde, 2014) identifica una base ortogonal de n dimensiones (las mismas que existen en el espacio de características, $B = \{b_1,b_2,\cdots,b_n\}$). Si llamamos σ_i^2 a la varianza de los datos en la dirección b_i, se ha de cumplir la siguiente condición:

$$\max_{b_i} \sigma_i^2 \quad / \quad b_i \perp b_j \quad \forall j < i \tag{2.59}$$

que impone la condición de que las varianzas σ_i^2 son máximas en el subespacio ortogonal que se considere.

A partir de la base B es posible generar una transformación de cambio de ejes desde la base canónica (la del espacio de características) hasta la base B (el espacio de componentes principales). Las figuras **2.9** y **2.10** muestran el antes y el después del cambio de coordenadas.

La reducción dimensional se efectúa proyectando las coordenadas sobre el subconjunto de direcciones $B' = \{b_1,b_2,\cdots,b_m\}$[29] que no es más que tomar las primeras m coordenadas de los puntos referidas a la base B.

Este procedimiento es muy sensible a las escalas de las características, por lo que se debe reescalar las coordenadas como se comentó en la sección de preprocesado de datos.

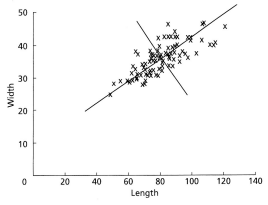


Figura 2.9: PCA. Espacio de características (Swan & Sandilands, 1995)

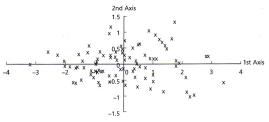


Figura 2.10: PCA. Espacio de componentes principales (Swan & Sandilands, 1995)

— **Escalado Multidimensional (MDS)** (*Multidimensional Scaled*). El objetivo del MDS (Young, 2013) es realizar una reducción dimensional en puntos de un espacio de forma que las

[29] Con $m < n$.

distancias antes y después de la transformación sean las más parecidas posibles.

Sea un conjunto de puntos $X = \{x_1, x_2, \cdots, x_I\}$ con $X \subset \mathbb{R}^N$. Definimos $\delta_{i,j}$ como la distancia entre los puntos x_i y x_j.

$$\Delta = \begin{pmatrix} \delta_{1,1} & \delta_{1,2} & \cdots & \delta_{1,I} \\ \delta_{2,1} & \delta_{2,2} & \cdots & \delta_{2,I} \\ \vdots & \vdots & \ddots & \vdots \\ \delta_{I,1} & \delta_{I,2} & \cdots & \delta_{I,I} \end{pmatrix} \tag{2.60}$$

y la matriz Δ representa a la matriz de distancias entre puntos. El objetivo de MDS es encontrar el conjunto de puntos $X = \{y_1, y_2, \cdots, y_I\}$ con $Y \subset \mathbb{R}^M$ en el que $\|y_i - y_j\| \approx \delta_{i,j}$ (para que el algoritmo funcione como reducción dimensional, $M < N$).

En la práctica, MDS es un problema de optimización en la que se intenta encontrar el mínimo:

$$\min_{y_1, \cdots, y_I} \sum_{i<j} (\|y_i - y_j\| - \delta_{i,j})^2 \tag{2.61}$$

que se resuelve con las herramientas específicas de optimización.

Algunas de las técnicas que vamos a discutir se basan en el concepto de variedad (*Manifold*). Una variedad es un espacio topológico que localmente se comporta como un espacio euclídeo (esto es, que las distancias entre puntos muy cercanos se pueden medir con la distancia euclídea); pero que globalmente no tiene porqué serlo[30].

Las técnicas basadas en variedades consideran que las observaciones efectuadas en el espacio de características están sobre una variedad y la representan (de forma discreta) como un grafo en el que sólo hay arcos entre puntos que comparten un mismo entorno local y cuya longitud es la distancia entre dichos puntos. Dependiendo de la técnica empleada, es posible que el grafo se deforme; pero siempre manteniendo los arcos entre los nodos y sus longitudes[31].

— **Isomapas** Los isomapas (Tenenbaum, De Silva, & Langford, 2000) se apoyan en el concepto de variedad y de su grafo asociado para generan una matriz de distancias entre puntos. Para ello, definen la distancia entre puntos como la longitud de

[2.30] En un pañuelo de tela (ejemplo de una variedad de dimensión 2) podemos considerar que la distancia entre un cruce de hilos y el siguiente es constante y que se puede medir usando la distancia euclídea entre dichos puntos. En cambio, la distancia física entre los extremos del pañuelo dependerán de cómo esté éste doblado.

[2.31] Volviendo al ejemplo del pañuelo, cualquier manipulación del pañuelo como un doblado o arrugado es válido siempre que no se deformen o rompan los hilos del mismo.

la geodésica en el grafo que une dichos puntos. Así, para cada par considerado, su distancia será la longitud del camino más corto de entre todos los posibles en el grafo.

Dado que el grafo refleja una variedad, la distancia entre puntos suficientemente cercanos coincidirá con su distancia euclídea.

La matriz de distancias es analizada con el algoritmo MDS que genera el nuevo espacio de dimensiones reducidas.

— **Desplegado de máxima varianza MVU** (*Maximum Variance Unfolding*) La técnica (Weinberger & Saul, 2006) consiste en manipular el grafo asociado a la variedad (sin alterar las longitudes de los arcos) para aumentar la distancia total entre los puntos.

Esta transformación de «estirado» del grafo, es un problema de optimización cuadrática con las siguientes características:

- Cada observación $x_i \in \mathbb{R}^N$ se pretende sustituir por un vector $y_i \in \mathbb{R}^M$ con $M < N$.
- Se pretende maximizar $\sum_{i,j} \|y_i - y_j\|^2$ sujeto a:
  - $\|y_i - y_j\|^2 = \|x_i - x_j\|^2$ si los puntos i y j comparten vecindad. Esto es, conservar las propiedades locales.
  - y $\sum_i y_i = 0$ que garantiza que las coordenadas están centradas en el origen.

Esta técnica funciona como una generalización no lineal del análisis de componentes principales ya expuesto. Aunque este solo se encarga de buscar la dirección más privilegiada (ya que no puede deformar la nube de puntos) y el MVU deforma la nube para forzar varianzas máximas.

— **Referencia localmente lineal LLE** (*Locally-Linear Embedding*) Estrategia (Wang, 2012) que también se basa en el grafo asociado de la variedad. Y se desarrolla en dos etapas:

Referir cada punto como combinación lineal de los puntos de su entorno. Para ello se minimiza la función

$$\varepsilon(w) = \sum_i \left| x_i - \sum_j w_{ij} x_j \right|^2 \tag{2.62}$$

en la que $w_{ij} = 0$ si los puntos x_i y x_j no están conectados directamente y $\sum_j w_{ij} = 1$.

Una vez calculados w_{ij}, se buscan los vectores $y_i \in \mathbb{R}^M$ que minimicen la función de coste:

$$\Phi(y) = \sum_i \left| y_i - \sum_j w_{ij} y_j \right|^2 \tag{2.63}$$

en el que ahora los pesos w_ij son constantes.

— **Autoencoder** Un autoencoder es una red neuronal multicapa (Hinton & Salakhutdinov, 2006) que tiene una capa central de tamaño reducido y que se entrena para que la salida sea una reconstrucción de los valores proporcionados a la entrada.

La arquitectura de las redes neuronales impone que si la entrada y salida presentan (idealmente) la misma información, cada capa intermedia ha de contener igualmente dicha misma información. El estrechamiento provocado en la capa central del autoencoder reduce el número datos (los valores de salida de dicha capa central) necesarios para expresar la misma información que en los datos originales (los valores de la entrada en la red).

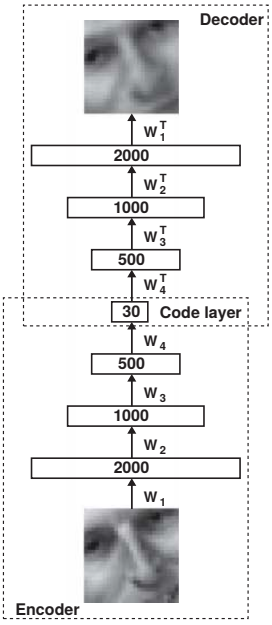


Figura 2.11: Autoencoder (Hinton & Salakhutdinov, 2006)

Si observamos la **figura 2.11**, en la parte inferior está la imagen original codificado en un espacio de características de 2000 descriptores (y en la parte superior la reconstrucción de la misma tras el procesado por la red neuronal). La capa central tiene solamente 30 neuronas por lo que, tras el entrenamiento, se ha producido una reducción de 2000 a 30 características.

— **Reducción de dimensionalidad multifactor (MDR)** (*Multifactor Dimensionality Reduction*) Diseñado para identificar genes deslocalizados que causan alto riesgo de ciertas enfermedades (Gola, Mahachie John, Van Steen, & König, 2015; Ritchie, et al., 2001). Fuera ámbito original, es aplicable a aquellos sistemas en los que existe una variable binaria que estimar.

En su formulación original se parte de una serie de factores (genéticos o ambientales) que están codificados como presentes o no presentes[32]. El **algoritmo 2.4** muestra de forma simplificada el cálculo de la reducción dimensional: sobre un espacio de N características se prueban todos los subespacios existentes sobre el que se construye un índice que se genera para cada observación: la razón de factores presentes en el

[2.32] Existen ampliaciones que permiten otros tipos de valores en las características.

for all $d \leftarrow 1,\cdots,N$ **do**
  for all $f_d \leftarrow$combinación de d descriptores (l_k) **do**
    for all observación $j \leftarrow 1,\cdots$ **do**
      $r_j \leftarrow \frac{\sum_k l_k = presente}{\sum_k l_k = no\,presente}$
      if $r_j > T$ **then**
        $nl_j \leftarrow presente$
      else
        $nl_j \leftarrow no\,presente$
      end if
    end for
    $error_{f_d} \leftarrow |nl - objetivo|$
  end for
end for
$reducción \leftarrow \min_{f_d} error_{f_d}$

Algoritmo 2.4: *Multifactor Dimensionality Reduction*

subespacio entre los que no se encuentran presentes y a partir de un umbral dado T se construye una predicción nl que se comparará con la variable *objetivo*.

La elección del mejor subespacio f_d se efectúa aplicando un criterio de validación que evite los efectos del sobreentrenamiento. Produciéndose una reducción de descriptores de N a d en este caso.

Selección de Características

La selección de características pretende identificar un subconjunto de características que reduzcan la redundancia de información y maximicen el rendimiento de las herramientas de análisis. Estas técnicas (cuando el objetivo es una clasificación) suelen ser supervisadas por lo que requieren de una etiqueta que identifique la categoría a la que pertenece cada observación[33] realizada.

[2.33] Punto en el espacio de características.

La ventaja que aporta frente a la extracción de características es la inmediata interpretación de los resultados en función de características originales del sistema.

La **figura 2.12** muestra un esquema en el que se clasifican los distintos enfoques para realizar la selección de características. Se distinguen tres grandes grupos en función de la relación que tengan los descriptores entre si.

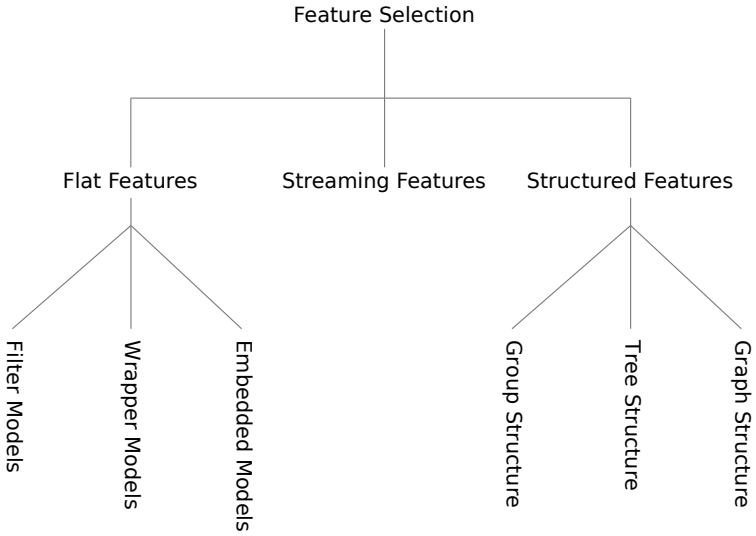


Figura 2.12: Clasificación de técnicas de selección de características (Tang, Alelyani, & Liu, 2014)

- *Flat Features*. Considera que los descriptores son independientes entre si. Esta categoría está dividida, a su vez en tres subcategorías en función de cómo se calcula la selección de las características.
  - *Filters* que hacen una extracción autónoma del análisis posterior a aplicar.
  - *Wrappers* que usan el análisis que se aplicará a continuación para seleccionar las características más idóneas.
  - *Embedded* que son los métodos de análisis que incorporan, en su procedimiento, la selección de las características más relevantes.
- *Structured Features*. Parte de que los descriptores presentan una relación o estructura entre si. La subdivisión presente en las técnicas sobre características estructuradas depende del tipo de relación determinada entre los factores habiéndose considerado estructuras como grafos, árboles o grupos.

  La consideración de estas relaciones permiten mejorar la selección de estas características. Por ejemplo, en el estudio de biología computacional sobre el arrayCGH (Tibshirani, Saunders, Rosset, Zhu, & Knight, 2005), las características empleadas (los números de copias de porciones de ADN a lo largo

del genoma) tienen el orden espacial natural. El uso de esta ordenación (que deriva de una cierta estructura) mejora el rendimiento de las posteriores operaciones de análisis.

– *Streaming Features*. Los métodos anteriores asumen que todas las características son conocidas de antemano. La característica fundamental de estos métodos es su capacidad de proporcionar una salida e ir actualizándola con el tiempo, antes de tener toda la información disponible.

  Estás técnicas están pensadas para sistemas en los que los datos se van produciendo de forma continua e ilimitada en el tiempo, como pueden ser las plataformas Twitter o Facebook en los que los descriptores pueden ser la frecuencia de aparición de cada palabra registrada. En estos sistemas hay que evaluar si la aparición de una nueva palabra debe ser considerada o ignorada, no siendo práctico tener que esperar *toda* la información recogida antes de procesar.

A continuación, se describen con un poco más de detalle algunas técnicas empleadas de selección de características.

Filter Models

Efectúan su selección exclusivamente por características de los datos sin emplear ningún algoritmo de clasificación (Liu & Motoda, 2007). En general el filtrado se efectúa en dos etapas: una evaluación de las características por cierto criterio y una selección de los mejores candidatos.

En función de la técnica empleada, la evaluación puede ser univariable (en la que se evalúa de forma independiente característica a característica) o multivariable (en la que se consideran la idoneidad de conjuntos de características). Este segundo enfoque, aunque computacionalmente más complejo, es capaz de detectar grupos de descriptores que aportan información redundante.

— ***Fisher Score*** (Zhao, et al., 2010) Se basa en que las características de gran calidad deberían asignar valores similares a instancias de la misma clase y valores distintos a instancias de distintas clases.

  La puntuación del descriptor i (S_i) se calcula como

$$S_i = \frac{\sum_j^c n_j(\mu_{ij} - \mu_i)^2}{\sum_j^c n_j \rho_{ij}^2} \tag{2.64}$$

donde μ_{ij} y ρ_{ij} son la media y varianza de la característica i en la clase j, n_j es el número de instancias en de la clase j y μ_i es la media de la característica i.

Esta puntuación evalúa las características individualmente por lo que no es capaz de detectar redundancia de características, aunque existen desarrollos como (Gu, Li, & Han, 2012) que generalizan la puntuación *Fisher* para grupos de descriptores.

— **Métodos basados en información mutua** (Peng, Long, & Ding, 2005) Es un sistema de alta eficiencia computacional y de simple interpretación muy usado. Se basa en medir la dependencia entre características y etiquetas calculando la ganancia de información entre la característica número i (denotado como f_i) y las etiquetas C.

La ganancia de información (IG) se determina como

$$\mathrm{IG}(f_i,C) = \mathrm{H}(f_i) - \mathrm{H}(f_i|C) \tag{2.65}$$

donde $\mathrm{H}(f_i)$ es la entropía de f_i y $\mathrm{H}(f_i|C)$ es la entropía de f_i después de observar las etiquetas C:

$$\mathrm{H}(f_i) = -\sum_j p(x_j)\log_2(p(x_j)) \tag{2.66}$$

$$\mathrm{H}(f_i|C) = -\sum_k p(c_k)\sum_j p(x_j|c_k)\log_2(p(x_j|c_k)) \tag{2.67}$$

Cuanto mayor sea la ganancia de la información, más relevante es la característica evaluada. Este método también es univariable aunque, nuevamente, existen extensiones para evaluar redundancias como (Yu & Liu, 2003).

— **ReliefF** (Kira & Rendell, 1992; Robnik-Šikonja & Kononenko, 2003) Los algoritmos Relief y ReliefF seleccionan las características que mejor diferencian entre clases (siendo el primero exclusivamente para clasificaciones binarias y el segundo para sistemas multiclase).

En su formulación más sencilla, se establece la puntuación de cada descriptor en función de la cercanía de los valores de este que comparten categoría y la distancia de los que no la comparten.

Para ello, es necesario normalizar los valores de las características al intervalo [0,1] y muestrear l observaciones de los datos disponibles. La puntuación del descriptor i es

$$S_i = \frac{1}{l}\sum_{k}^{l} \mathrm{d}(x_{ik},\mathrm{nearmiss}(x_{ik})) - \mathrm{d}(x_{ik},\mathrm{nearhit}(x_{ik})) \tag{2.68}$$

donde

- x_{ik} es el valor del descriptor i de la observación k,
- nearmiss(x_{ik}) es el valor del descriptor i más cercano a x_{ik}; pero de una observación catalogada en una categoría distinta a la asociada a la observación k y
- nearhit(x_ik)) es el valor más cercano; pero en esta ocasión, de observaciones que sí tengan la misma categoría.

La distancia entre los valores (d) se puede medir con cualquier métrica aunque (Robnik-Šikonja & Kononenko, 2003) recomiendan la distancia *Manhattan* por su eficiencia y eficacia.

Wrappers Models

El mayor inconveniente de los filtros es que la independencia respecto al sistema posterior de aprendizaje no permite seleccionar las propiedades que éste pueda aprovechar en mayor medida (Hall & Smith, 1999). Las técnicas *wrapper* (de envoltorio) postulan que el subconjunto óptimo de características dependen de las peculiaridades específicas de los algoritmos que posteriormente procesarán los datos.

El algoritmo de aprendizaje funciona como una caja negra con la que interactúa el *wrapper*. Dado el tamaño del espacio de selección de m características es $O(2^m)$ lo que, salvo que m sea pequeño, hace inviable una búsqueda exhaustiva de todas las posibilidades, se requiere aplicar una estrategia de búsqueda de soluciones.

A partir del algoritmo de *machine learning* escogido, el modelo *wrapper* de selección se aplicaría en los siguientes pasos:

1. selección de un subconjunto de descriptores,
2. evaluación de los descriptores empleando como indicador el rendimiento del algoritmo a usar y
3. repetir los pasos 1 y 2 hasta llegar a la calidad deseada.

La construcción de un *wrapper* que debe responder a tres preguntas: cómo seleccionar el subconjunto de descriptores, cómo evaluar el desempeño del subconjunto y cómo evolucionar a partir de los resultados obtenidos.

Respecto a la selección inicial de descriptores, se presentan dos enfoques: el enfoque directo parte de un subconjunto vacío y progresivamente se van añadiendo nuevos descriptores (si es que mejoran el rendimiento del sistema) y el enfoque inverso que parte del espacio de descriptores completos del que se van eliminando aquellos que se consideren redundantes.

Algunas de las estrategias de búsquedas más usadas son:

— ***Hill Climbing*** Consistente en la expansión del conjunto actual con el descriptor (no usado) que proporcione la máxima precisión (Russell & Norvig, 2005). El sistema termina cuando ninguna incorporación de un descriptor puede mejorar el subconjunto actual.

— ***Best First*** Un heurístico evalúa cuales de los descriptores que no han sido usados es el más prometedor y lo incorpora al subset.

  El heurístico le proporciona una mayor robustez que el caso de la ascensión de la montaña que solo usa la precisión alcanzada como mecanismo de control (Kohavi & John, 1997).

— **Ramificación y poda** (*branch-and-bound*) Construye un árbol de búsqueda en la que la calidad de las estimaciones de etapas anteriores establecen unos umbrales que permiten podar las ramas menos favorables en el árbol de búsqueda (Land & Doig, 1960).

El desconocimiento del rendimiento real de los algoritmos de aprendizaje para cada problema, hace que las estimaciones efectuadas con los datos muestreados (nuestras observaciones) puedan sufrir de problemas de sesgo. Ello obliga a que las valoraciones de eficacia sean usualmente realizadas sobre conjuntos de validación o empleando cross-validación.

Los modelos de envoltorio suelen tener una mayor efectividad que los filtros (Liu & Motoda, 2012) a costa de ser mucho más costosos computacionalmente.

Embedded Models

Los filtros son eficientes y no requieren cross-validación; pero no tienen en cuenta las características del algoritmo que se aplicará

a continuación. Los modelos de envoltorio son más precisos, pero computacionalmente son costosos ya que obligan a ejecutar el algoritmo de aprendizaje escogido muchas veces para asegurar la calidad del subconjunto de descriptores utilizados.

Los modelos incorporados (*Embedded Models*) presentan las ventajas de ambos enfoques: incluyen interacción con los algoritmos de aprendizaje siendo menos intensivos computacionalmente que los métodos de envoltorios (Liu & Yu, 2005).

Existen tres tipos de modelos incorporados.

— **Métodos de poda** Entrenan el modelo con todos los descriptores y una vez generado intentan eliminar los menos útiles anulando la contribución de estos sobre el modelo.

  Un ejemplo de ello es la eliminación de características recursiva usando máquinas de vector soporte (SVM) (Guyon, Weston, Barnhill, & Vapnik, 2002).

— **Algoritmos con mecanismos incorporados** Son técnicas que en su proceso tienen incorporado un mecanismo de selección de descriptores como ID3 (Quinlan, 1986) o C4.5 (Quinlan, 2014).

— **Modelos de regulación** Son modelos que incorporan una función objetivo que se optimiza para reducir los errores de entrenamiento y, simultáneamente, para que los efectos de los descriptores (w) sean pequeños. Una vez calculado los coeficientes aplicados a cada descriptor, se eliminan aquellos que sean 0 o cercanos (Ma & Huang, 2008).

  El problema de optimización con regulación puede expresarse como

$$\hat{w} = \min_{w} \mathrm{c}(w,X) + \alpha\,\mathrm{penalty}(w) \tag{2.69}$$

  donde $\mathrm{c}(w,X)$ es la función objetivo original, $\mathrm{penalty}(w)$ es la función de regulación y el coeficiente α es el balance entre el primer y segundo término.

  Algunas de las funciones de regulación más conocidas son *Lasso*[34] (Tibshirani, 2011), *Adaptive Lasso* (Zou, 2006), regularización *Bridge* (Huang, Horowitz, & Ma, 2008) y la regulación de red elástica (Zou & Hastie, 2005). El excelente rendimiento de estos modelos hace que tengan gran aceptación en la actualidad (Tang, Alelyani, & Liu, 2014).

[34] $penalty(w) = \sum_i^m |w_i|$, donde w_i representa al coeficiente asociados al descriptor i.

Structured Features

Todas las estrategias presentes en la categoría de *Flat Features* presumen la independencia entre los distintos descriptores[35]. Incorporar el conocimiento sobre las estructuras presentes en los descriptores pueden mejorar significativamente el rendimiento y ayudar a la identificación de características significativos.

Existen muchas aplicaciones en las que los descriptores van asociados en grupos[36]. En el momento de la selección de características, estos grupos tienden a considerarse como un todo en el que se incluye el grupo entero o no se incluye (Yuan & Lin, 2006).

Existen aplicaciones en los que una característica puede pertenecer simultáneamente a dos o más grupos[37] para los que se han diseñado extensiones como en (Jacob, Obozinski, & Vert, 2009) que se aplican a la función de regulación *Lasso*.

Análogamente y en función de la aplicación sujeto de estudio, es posible aplicar modificaciones similares en las funciones de regulación para reflejar efectos de estructuras en árbol (Liu & Ye, 2010) o grafos en los que hay relaciones estructurales de los descriptores dos a dos (Sandler, Blitzer, Talukdar, & Ungar, 2009).

Streaming Features

La minería de textos utiliza, entre otros, un descriptor denominado tf-idf (*Term Frecuency - Inverse Document Frecuency*) que expresa la importancia de una palabra en un documento dentro de una colección de documentos[38]. Esto implica que cada palabra analizada tiene su descriptor tf-idf asociado. Cuando el corpus de textos es conocido de antemano, es posible emplear las técnicas de selección de descriptores comentados en los apartados de modelos planos y estructurados.

La plataformas de mensajería Twitter genera más de 670 millones de mensajes al día (Internet live stats, 2017). En ese volumen de mensajes, nuevas palabras aparecen constantemente y no es posible ni establecer un vocabulario a priori ni esperar a recoger todas las palabras que puedan surgir para empezar un análisis con las herramientas de selección ya comentadas.

Ante problemas de selección de características en los que la información es un flujo de datos sin fin[39], es necesario establecer estrategias de análisis adaptadas (Wang, Zhao, Hoi, & Jin, 2014).

En general, la estructura típica de un sistema de selección de características *online* tiene los siguientes pasos:

[35] Aunque estos pudieran presentar alguna redundancia de información.

[36] Por ejemplo, en procesamiento de señales cada banda de frecuencia representa un grupo del que se extraen distintos descriptores (McAuley, Ming, Stewart, & Hanna, 2005).

[37] Por ejemplo en el estudio de efectos de una proteína o gen que puede afectar a distintos fenómenos biológicos (Kim & Xing, 2009).

[38] Cómo de importante es una palabra en una novela dentro de un conjunto de novelas o un término en un artículo dentro de una colección de artículos.

[39] También llamados *online*.

1. Generar un nuevo descriptor.
2. Determinar si el nuevo descriptor se incorpora a la selección actual de descriptores.
3. Determinar si es necesario eliminar algún descriptor de la selección actual de descriptores.
4. Volver al paso 1.

Cada algoritmo de selección tiene distintas implementaciones de los pasos 2 y 3, determinando la estrategia de selección. Así, el algoritmo *grafting* (Perkins, Lacker, & Theiler, 2003) parte de la regularización (**ecuación 2.69**) usando *Lasso*

$$\hat{w} = \min_{w} \mathrm{c}(w,X) + \alpha \sum_{i}^{m} |w_i| \tag{2.70}$$

Cuando se genera un nuevo descriptor, se acepta si la reducción de la función de coste c (por incluir un nuevo w_j) es superior al incremento que produce la penalización. Matemáticamente:

$$-\frac{\partial c}{\partial w_j} > \alpha \tag{2.71}$$

2.1.5 *Reconocimiento de patrones discriminantes*

La técnicas de reconocimiento de patrones suelen agruparse en dos grandes familias dependiendo del enfoque que se aplique (Fu, 1982): la primera está basada en teorías de decisión en la que se pretende determinar las características que determinan un patrón. Estas generalmente funcionan efectuando una partición del espacio de características y etiquetando cada subespacio generado. El segundo grupo son las técnicas de análisis sintáctico de las que se hablará en la siguiente sección (**sección 2.1.6**).

El reconocimiento de patrones por teorías de decisión es el enfoque ideal para la consecución del segundo objetivo planteado en esta tesis (en la **página 6**) que busca las características que definen que una observación pertenezca a una categoría dada frente a otras posibles. La identificación de qué descriptores son relevantes y para qué valores de estos se identifica una categoría se conoce como reconocimiento de patrones discriminantes.

El proceso de reconocimiento de patrones discriminantes se apoya sobre sistemas de clasificación que expongan el cuáles son los factores que sobre los que se hace la clasificación. De los sistemas de clasificación más comunes (Kotsiantis, Zaharakis, & Pintelas, 2007; León, López, Monedero, & Montaño, 2001)[40], vamos

[2.40] Los sistemas básicos citados son:

- Árbol de decisión y Reglas de aprendizaje,
- Redes neuronales,
- Red Bayesiana,
- *k-Nearest Neighbour*,
- Máquinas de vector soporte (SVM) y
- combinaciones.

a centrarnos en dos de ellos que aportan una mayor facilidad de conversión del conocimiento adquirido de los patrones para su uso por las personas: los árboles de decisión y clasificadores Naive-Bayes.

Árboles de decisión

Un árbol de decisión es una estructura de modelado de datos útil para clasificación o regresión de datos(Breiman, Friedman, Olshen, & Stone, 1984). En un árbol de decisión, el espacio de datos es dividido sucesivamente en dos partes hasta llegar a un nivel de detalle deseado, en el que se le asigna a cada punto del subespacio un valor del modelo. Dada la naturaleza del particionado, el árbol de decisión es representado como un árbol (un grafo acíclico y direccional) en el que cada nodo representa un valor, un conjunto de observaciones, y una condición de particionado.

La **figura 2.13** muestra los elementos asociados a un nodo. El nodo t, tiene asociado una serie de observaciones ($S(t)$) y una función denominada de impureza $\phi(t)$. Un nodo puede tener conjunto de nodos hijos (t_1,t_2,$\cdots$,t_n) determinados por una regla de construcción. Cada nodo hijo tendrá, a su vez, un conjunto de observaciones (determinados por la regla) y una impureza (calculado a partir de su conjunto de observaciones.

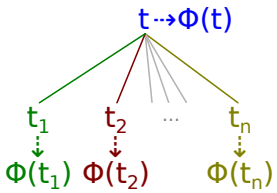


Figura 2.13: Anatomía del nodo de un árbol de decisión

La construcción del árbol se efectúa siguiento un proceso recursivo (Therneau, Atkinson, Ripley, et al., 2010): se escoge la variable que mejor[41] divide los datos de t. El proceso se repite por cada subnodo creado hasta llegar a una condición de detención o se llega a un tamaño mínimo de observaciones por nodo.

Si la variable analizada (X) para dividir un nodo es de tipo numérica[42], las reglas que construyen las divisiones tendrán la forma $b < X \leq c$; siendo el número de posibles particiones igual a $n-1$ (con n el número de valores distintos encontrados en X. En cambio, si la variable X es de tipo categórica, la regla de selección hacia un hijo se expresará como $X \in A$ (con A un subconjunto de valores de X) y el número de posibles particiones quedaría como $2^{m-1}-1$ (siendo m el número de valores existentes en X).

La elección de la mejor partición para el árbol suele hacerse escogiendo como función objetivo a minimizar la suma de las impurezas de los nodos hijos que se producirían en dicha partición ($\sum_i \phi(t_i)$). Así, el árbol de decisión original, THAID (*THeta Automatic Interaction Detection*, de (Messenger & Mandell, 1972)), propone tres estrategias de partición:

[2.41] La elección de la variable y el punto de división es dependiente del algoritmo de construcción seleccionado.

[2.42] (Morgan & Sonquist, 1963)

– Maximizar el número de observaciones que aparecen en un subnodo que pertenecen a la categoría más frecuente.

– Minimizar la función de entropia $\phi(t) = -\sum_j p(j|t) \log p(j|t)$

– Minimizar el índice Gini $\phi(t) = 1 - \sum_j p^2(j|t)$

donde $p(j|t)$ es la probabilidad de que se dé la observación j en el nodo t.

Una vez aplicado el proceso de partición hasta su final, existen algoritmos que efectúan una segunda fase de poda del árbol en la que se pueden eliminar ramas que no se consideren relevantes. Dicha poda pretende buscar una simplificación del grafo que proporcione una mayor robustez resistencia al sobreajuste.

El modelo de clasificación en árbol presenta la gran ventaja de ser fácil de interpretar, proporcionando una rápida visión de la complejidad de los datos analizados en función del número de nodos generados. Además, permite generar reglas de clasificación fácilmente reproducibles por personas.

Los tipos de árboles de decisión más usados según (Loh, 2014) son:

— **CART** (*Classification And Regression Trees*) de (Breiman, Friedman, Olshen, & Stone, 1984). Sigue los mismos principios de construcción voraz de THAID; pero construye árboles más grandes (al no tener reglas de detención de la construcción del árbol) y termina haciendo una poda usando cross-validación sobre las ramas.

  La construcción sin detención y la posterior poda permite evitar los problemas presentes en THAID de sobre ajuste e infra ajuste de los datos.

— **CHAID** (*CHi-squared Automatic Interaction Detector*) de (Kass, 1980). Se basa en dividir cada nodo en función del tipo de variable existente. Si esta es numérica se divide el nodo en 10 intervalos iguales y si es categórica se divide en tantos hijos como valores categoricos tenga la variable considerada.

  Tras la división se comparan los hijos por pares y se fusionan los que cumplan con el criterio Bonferroni ajustado de significación estadística (Bland & Altman, 1995).

— **C4.5** de (Quinlan, 1993). Emplea como medida de la impureza la ganancia de ratio (una medida basada en la entropía

de los nodos). Posteriormente emplea un podado que, a diferencia de CART no se basa en cross-validación sino en una heurística.

— **QUEST** (*Quick, Unbiased and Efficient Statistical Tree*) de (Loh & Shih, 1997). Los algoritmos mencionados anteriormente sufren de un sesgo a la hora de construirse. Si la variable X_1 tiene n_1 posibles particiones y la variable X_2 tiene n_2 con $n_1 < n_2$, en estos algoritmos la variable X_2 tiene mayor probabilidad de quedar como la regla que divide el nodo.

QUEST ha sido construido para no sufrir de dichos efectos de sesgo. Para ello, utiliza *F*-tests y análisis X^2 a la hora de seleccionar la variable de partición.

Tras la construcción del árbol, emplea un podado equivalente a CART.

Naive-Bayes

El teorema de Bayes (Russell & Norvig, 2005) plantea, en una de sus formulaciones, cómo se actualiza la probabilidad de que un evento A ocurra cuando se observa un evento B.

Aplicado al problema que nos ocupa, el teorema de Bayes podría enunciarse como:

$$P(c_A|d_B) = \frac{P(c_A)P(d_B|c_A)}{P(c_A)P(d_B|c_A) + P(\neg c_A)P(d_B|\neg c_A)} \quad (2.72)$$

Donde,

$\boldsymbol{P(c_A)}$ Es la probabilidad a priori de que una pieza corresponda a una categoría A.

Esta probabilidad se conoce como la probabilidad a priori o previa ya que representa el punto del que se parte antes de considerar nuevas evidencias. La estimación de esta probabilidad puede hacerse partiendo de una situación no informada (en la que se puede suponer una distribución equiprobable entre todas las categorías consideradas) o establecer una probabilidad proporcional al número de elementos registrados de cada categoría.

$\boldsymbol{P(c_A|d_B)}$ Probabilidad a posteriori de que una pieza pertenezca a la categoría A cuando se ha observado la evidencia d_B. En nuestro planteamiento, una evidencia es la observación de que un descriptor tenga un valor dado.

Podemos entender esta probabilidad como un refinamiento de $P(c_A)$ cuando se ha aportado nueva información al sistema.

$P(d_B|c_A)$ Es la probabilidad de que el descriptor tenga un valor dado considerando solo las piezas que pertenecen a la categoría A.

$P(\neg c_A)$ Probabilidad de que la pieza no sea de la categoría A

$P(d_B|\neg c_A)$ Es la probabilidad de que el descriptor tenga un valor dado considerando solo las piezas que no pertenecen a la categoría A.

El teorema de Bayes permite el cálculo de probabilidades condicionadas al facilitar una fórmula que relaciona una probabilidad condicionada (como $P(A|B)$ en función de su recíproca $P(B|A)$) ya que una de estas expresiones suele ser más compleja de calcular que la otra.

Es posible aplicar el teorema de Bayes en el caso de dos o más evidencias; pero esto requeriría calcular probabilidades de co-ocurrencia entre cada n-tupla de evidencias a considerar. Si bien es posible, es poco práctico este enfoque. El modelo Naive-Bayes considera, ingenuamente[43], que las evidencias estudiadas son independientes entre sí[44] y que, por tanto, la probabilidad de co-ocurrencia entre ambas evidencias es el producto de las probabilidades de cada evidencia por separado.

Este modelo proporciona una expresión asequible para el cálculo de las probabilidades condicionadas:

$$P(c_A|\wedge_i^n d_i) = \frac{P(c_A)\prod P(d_i|c_A)}{P(c_A)\prod P(d_i|c_A) + P(\neg c_A)\prod P(d_i|\neg c_A)} \tag{2.73}$$

A partir de la **expresión 2.73** es posible construir un clasificador naive-bayes siguiendo el **Algoritmo 2.5**.

[43] *Naive* significa justamente eso: ingenuidad

[44] Curiosamente, aunque en la mayoría de las aplicaciones reales los factores considerados no son independientes, el método funciona y no faltan investigadores que intentan explicar el éxito de un sistema que no debería (Rish, 2001; Zhang, 2004).

2.1.6 *Reconocimiento de patrones estructurales*

El segundo gran grupo de técnicas de reconocimiento de patrones es el análisis estructural o sintáctico (D'Ulizia, Ferri, & Grifoni, 2011; Horning, 1969). En algunos sistemas de reconocimientos de patrones la estructura en el que están organizados

Fase de entrenamiento (solo se realiza una vez)

1. Determinar las probabilidades a priori ($P(c_j)$) de cada categoría.
2. Calcular las probabilidades condicionadas ($P(d_i|c_j)$) de cada observación.

Fase de clasificación (se realiza por cada pieza a clasificar)

3. Determinar las evidencias de la pieza analizada. Si un descriptor incluye un valor no visto anteriormente, eliminar dicha evidencia de las disponibles.
4. Empleando la **ec. 2.73** calcular $P(c_j|\wedge_i^n d_i)$ para cada categoría disponible.
5. Asignar a la pieza la categoría j que tenga el $P(c_j|\wedge_i^n d_i)$ máximo.

Algoritmo 2.5: Clasificador Naive-Bayes

los datos es un aspecto importante en el que no solamente interesa saber la presencia de cierto elemento sino su relación con otros elementos existentes.

Diferencia entre características estructuradas y patrones estructurales

No hay que confundir las características estructuradas (comentado brevemente en la **página 56**) con patrones estructurados. Los primeros hablan de la relación existente entre descriptores de un elemento determinado, mientras que el reconocimiento de patrones estructurales no trabaja sobre descriptores sino sobre una codificación apropiada del objeto.

Así, en el caso de una cadena compuestas por campos delimitados por un símbolo separador, podríamos calcular descriptores como la longitud o la moda de los símbolos para cada parte. Los descriptores obtenidos presentarían una estructura a dos niveles: el indicador de campo y los descriptores de dicho campo.

Codificación

En el mismo ejemplo, el análisis sintáctico trabaja directamente sobre los símbolos tal y cómo se hayan codificado (sin pasar previamente por el proceso de la generación de características). La codificación juega un papel importante en el proceso y el elemento mínimo de la codificación de un objeto se llama primitivas del patrón. En el ejemplo dado de la cadena con campos, la primitiva más habitual sería el uso de caracteres; pero existen otras posibles representaciones (si los datos almacenados son nominales, se puede sustituir cada término por un identificador numérico tomado de un diccionario)[45].

[45] En la descripción de formas en una imagen binaria, una opción sería una serialización de unos y ceros y otra el empleo de *chain code* (Bribiesca, 2016; León, Molina, Fragoso, & Algarín, 1997). En una se destaca el área mientras en la otra codificación la forma del contorno.

El segundo aspecto a considerar en el reconocimiento de patrones es la construcción de reglas que permitan definir la gramática[46]. Una regla de producción identifica una secuencia de símbolos (que debe incorporar, al menos, un símbolo no terminal) y especifica un posible forma de sustitución de dicha secuencia por una nueva secuencia.

El proceso de derivación de una cadena parte siempre del símbolo inicial (S) sobre el que se efectúan sucesivas aplicaciones de reglas de sustitución (R) hasta que todos los símbolos presentes en la cadena sean terminales (los que definen las primitivas).

Para que esté bien definida una gramática, se necesita que todas las cadenas que pueda derivar pertenezcan a un lenguaje dado y que todas las cadenas del lenguaje se puedan producir con dicha gramática. Este proceso de generación de reglas (de forma óptima) esta identificado como un problema NP-completo (Flasiński & Jurek, 2014), por lo que en lenguajes con un número de cadenas elevadas, la única forma abordable de atacarlo es empleando heurísticas y estrategias de inducción de gramáticas que proporcionan resultados sub-óptimos.

Las gramáticas plantean dos estrategias de análisis contrapuestas:

- Generar cadenas pertenecientes a un lenguaje. Aunque el número de cadenas de un lenguaje sea infinito, el conjunto es contable, por lo que es posible establecer una enumeración que construya todas las cadenas.

  La generación de cadenas permite producir nuevas cadenas que no hayan sido observadas previamente en el proceso de análisis de los datos.

- Analizar una cadena y determinar si puede ser generada a partir de una gramática. Este proceso se denomina búsqueda de patrones estructurales (*Structural Pattern Matching*) y, a diferencia de la búsqueda de patrones discriminantes, la aplicación de estas técnicas no es inmediata (Gonzalez & Thomason, 1978).

  Este análisis presenta dos objetivos:

  - Estructural, que identifica que la cadena pertenece (o no) a un lenguaje por lo que funciona como clasificador.
  - Semántico, donde el proceso de análisis pasa por determinar las reglas que se han aplicado y la identificación de subcadenas con ciertos símbolos no terminales. La asociación de estos conjuntos de primitivas con los símbolos

[2.46] Recordamos que una gramática se define como una 4-tupla $< V_T, V_N, S, R >$. En el que se definían unos símbolos y las reglas de transformación de dichos símbolos. Más detalles en la **página 14**.

permite extraer información sobre características de la cadena.

El análisis de una frase como «Los niños corren» empleando la Gramática Española nos permite decir que la frase pertenece al lenguaje Español y, durante el proceso de generación, el símbolo `<sujeto>` se asocia con la subcadena «Los niños». A partir de esta asociación, podemos extraer información sobre quién o qué realiza la acción descrita.

Un lenguaje puede estar definido por múltiples gramáticas

Un conjunto de cadenas no definen una única gramática. Sea el lenguaje compuesto por una única cadena: $\{L_1$=«Los niños corren»$\}$. Los símbolos terminales corresponden a las letras de la oración: $\Sigma = \{'\mathtt{L}','\mathtt{o}','\mathtt{s}',' ','\mathtt{n}','\mathtt{i}',\cdots\}$ y las tablas **2.2a**, **2.2b** y **2.2c** muestran tres conjuntos de reglas distintas que permiten generar el mismo lenguaje. No existe a priori de un criterio que permita seleccionar cuál es la mejor gramática de un lenguaje dado, por lo que la elección de esta dependerá del objetivo buscado al analizar el lenguaje. Así, existen gramáticas diseñadas para comprimir información cuyo criterios de elección son el número y tamaño de las reglas generadoras y, por otro lado, existen otras gramáticas centradas en reflejar el pensamiento humano (como una gramática que describa las formas o la estructura de un idioma).

`<S>` → `<s><p>`	`<S>` → `<m1><m2>`	`<S>` → «Los niños corren»
`<s>` → «Los niños»	`<m1>` → «Los niño»	
`<p>` → «corren»	`<m2>` → «s corren»	
a. «sujeto-predicado»	b. «mitad-mitad»	c. «una regla»

Tabla 2.2: Distintas gramáticas para el lenguaje L_1

Clasificación de Chomsky

(Chomsky, 1959) referencia 4 tipos de gramáticas formales en función de las restricciones que se le imponen a las reglas de producción de las gramáticas. La **tabla 2.3** muestra los tipos, el tipo de lenguaje que generan y las restricciones aplicadas sobre las reglas de producción. En esta tabla (y en los posteriores comentarios), se seguirá el convenio siguiente: mayúsculas representan símbolos no terminales, minúsculas símbolos terminales y griegas subcadena compuesta por símbolos terminales y no terminales.

Gramática	Lenguaje	Restricciones
Tipo-0	Recursivamente enumerable	$\alpha \rightarrow \beta$ (sin restrictiones)
Tipo-1	Sensible al contexto	$\alpha A \beta \rightarrow \alpha \gamma \beta$
Tipo-2	Sin contexto	$A \rightarrow \gamma$
Tipo-3	Regular	$A \rightarrow a$ o $A \rightarrow aB$

Tabla 2.3: Gramáticas de Chomsky

Como puede observarse, las condiciones se vuelven más restrictivas (es posible construir menos reglas de producción) conforme aumenta el tipo de gramática además de que cada restricción es compatible con todas las anteriores. Esto nos lleva a todas las gramáticas tipo-n son a su vez gramáticas tipo-$n-1$ (no siendo cierto lo contrario). La **figura 2.14** muestra las relaciones entre distintas gramáticas.

Gramáticas tipo 0 Esta gramática no impone ninguna restricción a las reglas de producción. Cualquier símbolo (sea terminal o no) puede sustituirse por cualquier otro símbolo.

Las gramáticas tipo 0 incluyen todas la gramáticas formales y requieren para su verificación de una máquina de Turing.

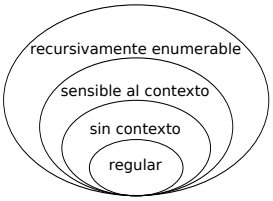


Figura 2.14: Gramáticas de Chomsky

Gramáticas tipo 1 Las gramáticas sujetas a contexto requieren que el símbolo no terminal que se sustituya esté entre ciertos delimitadores que no se modifican.

Gramáticas tipo 2 Las gramáticas libres de contexto obligan a que las reglas de sustitución deben poder aplicarse de igual forma ante todos los delimitadores posibles. Siendo, por tanto más restrictiva que el tipo 1 (cuyas reglas pueden aplicarse o no según el contexto dado).

Gramáticas tipo 3 Son las más restrictivas en las que en cada sustitución se ha de producir un símbolo terminal opcionalmente acompañado por uno no terminal[47].

Estas gramáticas pueden verificar cadenas empleando una máquina de estados.

[47] En una gramática solo se permitirán reglas tipo $A \rightarrow aB$ o $A \rightarrow Ba$ pero no ambos simultáneamente.

Inducción de gramáticas

Gramáticas supervisadas y no supervisadas

Entre los distintos tipos de generación automática de gramáticas (de la Higuera, 2005), vamos a distinguirlos en función del nivel de supervisión que requieren:

– Los algoritmos no supervisados (Bisk & Hockenmaier, 2015) construyen la gramática siguiendo una serie de heurísticas

establecidas de antemano. Estas heurísticas establecen cómo buscar qué símbolos se combinarán.

Estos pueden ser deterministas, en cuyo caso, las gramáticas obtenidas siempre serán idénticas ante la misma entrada o no deterministas (Li & Chen, 2014) si no se puede garantizar dicha igualdad.

– Los algoritmos supervisados (Hwa, 1999) ajustan las reglas de la gramática a partir de una una función externa de evaluación (que es la que aporta el conocimiento). Como ejemplo de este tipo están los generadores por algoritmos genéticos. La elaboración de una función de evaluación de gramáticas requiere de un experto en el área y puede ser tan complejo como plantear una gramática directamente.

Aunque los algoritmos supervisados puede generar gramáticas ajustadas a los objetivos deseados, requieren de una función externa personalizada a cada problema y objetivos (función, que por otro lado es difícil de obtener). Esta dependencia dificulta su utilidad en soluciones de carácter genérico como se busca en esta tesis.

A continuación se describirán tres algoritmos de inducción de gramáticas no supervisados y deterministas que han sido escogidos a partir de algoritmos de compresión que se basan en detectar subcadenas frecuentes.

Sequitur

El algoritmo Sequitur(Nevill-Manning, 1996; Nevill-Manning & Witten, 1997) va construyendo, conforme se procesan nuevos símbolos de la cadena, un diccionario de pares de símbolos consecutivos (llamado digrama). Cuando un digrama aparece más de una vez, se sustituye todas sus apariciones (tanto en la cadena recibida como en el diccionario) creando un nuevo símbolo que se incorpora al diccionario.

La generación de símbolos se realiza aplicando 2 reglas:

Unicidad del digrama Cada digrama sólo puede aparecer una única vez. Si aparece más de una vez se crea un nuevo símbolo y se sustituyen los digrama por el símbolo.

Utilidad del símbolo Todo símbolo creado debe ser útil (aparecer más de una vez). Si solo aparece una vez, el símbolo se elimina sustituyéndolo por su contenido.

Posición símbolo	Cadena leída	Gramática generada	Notas
1	a	S → a	
2	ab	S → ab	
3	abc	S → abc	
4	abcd	S → abcd	
5	abcdb	S → abcdb	
6	abcdbc	S → abcdbc	«bc» Aparece dos veces
		S → aAdA	Aplica regla de unicidad de digramas
		A → bc	
7	abcdbca	S → aAdAa	
		A → bc	
8	abcdbcab	S → aAdAab	
		A → bc	
9	abcdbcabc	S → aAdAabc	«bc» Aparece dos veces
		A → bc	
		S → aAdAaA	Aplica regla de unicidad de digramas
		A → bc	«aA» Aparece dos veces
		S → BdAB	Aplica regla de Unicidad de digramas
		A → bc	
		B → aA	
10	abcdbcabcd	S → BdABd	«Bd» Aparece dos veces
		A → bc	
		B → aA	
		S → CAC	Aplica regla de unicidad de digramas
		A → bc	
		B → aA	«B» solo se usa una vez
		C → Bd	
		S → CAC	Aplica regla de utilidad
		A → bc	
		C → aAd	

Tabla 2.4: Generación de gramática Sequitur. Tomado de (Nevill-Manning & Witten, 1997)

La **tabla 2.4** ilustra la generación de una gramática Sequitur. Cada símbolo del diccionario generado $'\texttt{A}' = «bc»$ y $'\texttt{C}' = «abcd»$ representan una subcadena interesante. El símbolo inicial `'S'` no es considerado ya que representa la cadena entera y no una subcadena.

Re-Pair

Re-Pair(Larsson & Moffat, 2000) funciona de forma similar a Sequitur con la diferencia de que empieza a generar la gramática cuando se ha leido toda la cadena. Esta diferencia permite una mayor optimización en la generación de la gramática ya que se tiene toda la información disponible.

Con Re-Pair, los digrama escogidos para constituir nuevos símbolos se hacen en función del número de ocurrencias de los mismos (siendo usados primero los que aparezcan en más ocasiones) y el proceso se repite hasta que todos los digrama aparezcan una sola vez. La similitud de Re-Pair con Sequitur permite aplicar la regla de utilidad de Sequitur para reducir el número de símbolos no terminales generados.

La **tabla 2.5** se analiza la misma cadena que en el ejemplo de sequitur empleando ahora el algoritmo Re-Pair. En este caso, las reglas obtenidas son idénticas que en Sequitur (a falta de aplicar la regla de utilidad); pero este resultado es fortuito y no debe considerarse como regla general.

Gramática generada
S → abcdbcabcd
S → aAdAaAd
A → bc
S → BdABd
A → bc
B → aA
S → CAC
A → bc
B → aA
C → Bd

Tabla 2.5: Generación de gramática Re-Pair

Lempel-Ziv-Welch (LZW)

El último procedimiento de generación de gramáticas empleado está basado en el algoritmo de compresión Lempel-Ziv-Welch (Welch, 1984; Ziv & Lempel, 1978) (a partir de ahora LZW). En él y como ya se ha comentado, la fase de compresión del algoritmo consiste en la construcción de un diccionario de reglas de producción de una gramática. El **algoritmo 2.6** muestra el desarrollo básico.

A diferencia de los algoritmos Sequitur y Re-Pair, LZW no trabaja a partir de digramas; sino que trabaja intentando corregir ausencias en un diccionario que va generando. Cuando una expresión que aparece en la secuencia no está en el diccionario, añade la misma por si vuelve a salir. La **tabla 2.6** muestra la gramática generada para la cadena de ejemplo «abcdbcabcd». Si se observa la gramática resultante, se puede apreciar en las reglas de producción el efecto de este enfoque sin digramas. Por un lado, cada símbolo del alfabeto aparece como una regla de producción y, por otro lado, se construyen reglas de producción que no se usan nunca (debido a que se recogen subcadenas que sólo se han detectado en una ocasión).

Para aumentar la utilidad de análisis de la generación de gramáticas LZW, se proponen efectuar una limpieza de reglas de

```
w ←( )
añadir alfabeto al diccionario
loop
   k ←leer carácter
   wk ←concatenación w y k
   if wk existe en el diccionario  then
      w ←wk
   else
      escribir el código de w
      añadir wk al diccionario
      w ←k
   end if
end loop
```

Algoritmo 2.6: Algoritmo de compresión LZW

producción una vez que se ha generado completamente el diccionario con las siguientes modificaciones sobre el enfoque original.

- Eliminar de las reglas aquellas cuya producción sea un único símbolo terminal.
- Eliminar las reglas que no se hayan usado en la descripción de la cadena.

De esta forma, obtendremos un conjunto de subcadenas que hayan aparecido, al menos, dos veces en la cadena y de un tamaño mínimo de 2 elementos haciendo el diccionario comparable con los dos mecanismos descritos anteriormente.

#	Cadena considerada	w	k	Gramatica/Diccionario
0		( )		S←( )
				A←a, B←b, C←c, D←d
1	a	( )	a	S←( )
				A←a, B←b, C←c, D←d
2	ab	a	b	S←A
				A←a, B←b, C←c, D←d, E←ab
3	abc	b	c	S←AB
				A←a, B←b, C←c, D←d, E←ab, F←bc
4	abcd	c	d	S←ABC
				A←a, B←b, C←c, D←d, E←ab, F←bc, G←cd
5	abcdb	d	b	S←ABCD
				A←a, B←b, C←c, D←d, E←ab, F←bc, G←cd, H←db
6	abcdbc	b	c	S←ABCD
				A←a, B←b, C←c, D←d, E←ab, F←bc, G←cd, H←db
7	abcdbca	bc	a	S←ABCDF
				A←a, B←b, C←c, D←d, E←ab, F←bc, G←cd, H←db, I←bca
8	abcdbcab	a	b	S←ABCDF
				A←a, B←b, C←c, D←d, E←ab, F←bc, G←cd, H←db, I←bca
9	abcdbcabc	ab	c	S←ABCDFE
				A←a, B←b, C←c, D←d, E←ab, F←bc, G←cd, H←db, I←bca, J←abc
10	abcdbcabcd	c	d	S←ABCDFE
				A←a, B←b, C←c, D←d, E←ab, F←bc, G←cd, H←db, I←bca, J←abc
	abcdbcabcd	cd	( )	S←ABCDFEG
				A←a, B←b, C←c, D←d, E←ab, F←bc, G←cd, H←db, I←bca, J←abc

Tabla 2.6: Generación de gramática LZW

2.2 *Problemática del cante flamenco*

2.2.1 *Características del cante flamenco*

El cante flamenco es una manifestación musical artística de difícil clasificación. El soporte oral, que casi exclusivamente se emplea en su transmisión, hace que sea un sistema vivo en constante evolución.

FLAMENCO COMO MÚSICA DE AUTOR

Si bien, cada vez más, hay estudios de música flamenca y flamencología en los conservatorios, no puede considerarse una música académica al no compartir los intereses de esta. Quizás uno de los más importantes sea la conservación de las obras que permite establecer una forma canónica de interpretación de las mismas.

Tampoco puede ser considerado como folclore. El folclore es una música popular interpretada en acontecimientos específicos (como fiestas como la feria de abril de Sevilla, la navidad o actos sociales como un cumpleaños) y de interpretación por personas sin especiales habilidades vocales. El cante flamenco, en cambio, requiere de cantaores con cierto entrenamiento vocal y su cante no está no está circunscrito a condiciones espacio-temporales [48].

[2.48] Considérese la diferencia entre el cante de una sevillana y de una soleá, por ejemplo.

Es más, la forma actual del espectáculo flamenco, que inició su aparición en el siglo XIX (Castro Buendía, 2014; Núñez, 1998), muestra un arte profesionalizado aunque de fuerte arraigo en la vida social de Andalucía (Cruces Roldán, 2003).

Es por ello que es más correcto considerar el cante flamenco como música de autor en la que la creación y la interpretación de las piezas está fuertemente ligada. De este modo, es habitual identificar los cantes por su tipo y autor (como Soleá de Charamusco, Soleá de Camarón, o Soleá de Antonio Mairena, por citar tres tipos de soleares).

Aun así, dentro de la comunidad científica está considerada como etnomúsica (Rice, 2013) un área de aglutinación donde la musicología incluye a músicas no occidentales[49].

[2.49] Clásica y populares (pop, rock, folk, …).

Características Musicológicas

Según (Osuna Lucena, 1995), la personalidad musical del flamenco se puede resumir en:

> «la microtonalidad interválica, las escalas modales, la riqueza y acentuación rítmica, todo en torno a una filosofía improvisatoria y ajena a toda rigidez metódica»

A continuación describiremos brevemente cada una de estas características. En los casos que sea útil, se comparará con la música occidental (ya sea clásica o popular) a la que estamos más acostumbrados.

Microtonalidad interválica En flamenco, la menor razón entre las frecuencias de las notas es inferior al semitono[50] por lo que, además de las notas musicales tradicionales, pueden aparecer notas con afinaciones intermedias a estas dando la sensación de desafino.

[2.50] El semitono es la razón mínima entre dos frecuencias en la música occidental, cuyo valor es de $2^{\frac{1}{12}}$.

La afinación al cantar (desde el punto de vista académico) no se considera una virtud, es más, se considera una voz «poco flamenca» (Molina & Mairena, 1963).

En la música occidental, todas las frecuencias se definen en función de una de referencia siguiendo la expresión:

$$f_i = 2^{\frac{i}{12}} \cdot 440 \text{ Hz} \tag{2.74}$$

con $i \in \mathbb{Z}$ por lo que la razón entre dos notas musicales siempre será un múltiplo exacto del semitono.

Escalas modales De las 12 notas disponibles en una octava, las melodías no usan todas; siendo el conjunto de notas usables conocido como la escala. Para determinar qué notas pertenecen a una escala, existen reglas o modos que establecen la construcción de la escala escogida.

Los modos más usados en la música occidental son los modos «mayor» (o «jónico») y «menor» (o «eólico»). En contra, el flamenco usa fundamentalmente el «modo de mi» (también llamado «dórico griego» y «frigio medieval o moderno»[51]).

[2.51] Una explicación de porqué se utilizan nombres de modos aparentemente contradictorios se puede encontrar en (Romero Jiménez, 1996).

El empleo de un modo u otro, también limita las armonías que pueden usarse para acompañar a la melodía.

Cadencia andaluza A parte del modo hay otra característica, melódicamente hablando, del cante flamenco: la cadencia andaluza. La cadencia andaluza es un motivo musical compuesta por las notas La-Sol-Fa-Mi (que en el caso de tocarse como acompañamiento, los acordes serían La m - Sol M - Fa M y Mi M).

Ritmo Cuando se clasifica el cante flamenco, en función del ritmo, se puede dividir en dos grandes grupos (Moore, 2015): cantes «libres» y cantes «al compás».

En los cantes libres no hay un ritmo externo y la duración de las notas y motivos es voluntad exclusiva del cantaor siendo los tiempos un recurso expresivo más de la interpretación.

En los cantes a compás si existe un ritmo externo percibido impuesto por el cantaor que es acompañado por los demás elementos intervinientes (como pueden ser los guitarristas, palmeros y otros percusionistas). Aunque, en ningún caso, se puede hablar de ritmo con la precisión de un metrónomo. El ritmo sigue siendo un recurso expresivo (en menor medida que en los cantes libres) (Berlanga, 2017).

Los ritmos de la música occidental se caracterizan por métricas sencillas cuyas medidas más populares son: $\frac{4}{4}$ (en **figura 2.15**[52]) o $\frac{3}{4}$ ([53]). Estos ritmos se perciben de forma homogénea en todos los músicos y cantantes.

[2.52] cuatro golpes por compás, 1 fuerte y 3 débiles

[2.53] tres golpes por compás, 1 fuerte y 2 débiles

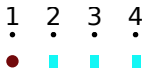


Figura 2.15: Compás $\frac{4}{4}$

Por contra, el flamenco presenta ritmos compuestos denominados «compás de amalgama» o más concretamente «hemiolia» (Tenzer, 2006) (sucesión de ritmos ternarios y binarios). El compás más usado es el $\frac{12}{8}$ (**figura 2.16**), que se descompone en una sucesión de 2 compases $\frac{3}{8}$ seguidos de 3 compases $\frac{2}{8}$. Esta amalgama produce una gran riqueza rítmica ya que los tiempos fuertes no están igualmente espaciados en el tiempo de forma regular.

Figura 2.16: Compás $\frac{12}{8}$

Además de la heterogeneidad temporal que proporciona el compás de amalgama, también existe una heterogeneidad vertical en el que los ritmos seguidos por el cantaor y los acompañantes (guitarristas, palmeros y percusionistas) pueden diverger[54].

[2.54] Quizás el ejemplo paradigmático de esta divergencia se encuentra en las seguiriyas en el que el cantaor canta a compás libre (sin compás) y el guitarrista va a compás (Berlanga, 2014).

Calixto Sánchez (cantaor y conferenciante de temas de flamenco) menciona (Teleprensa, 2014) como las características que principalmente definen el flamenco justo estas dos últimas características: la cadencia andaluza y el compás de amalgama.

Improvisación Los distintos estilos (llamados palos) en el flamenco tienen un número limitado de melodías que todos los cantantes siguen[55]. No se puede decir, por tanto, que el cante flamenco sea un cante (estrictamente hablando) improvisado.

[2.55] Aunque es difícil establecer un censo exacto de melodías, se puede afirmar que hay hasta 50 estilos distintos con hasta 85 variantes según el caso.

Aun así, el cante flamenco está abierto a improvisación e innovación personal (que es considerado un valor positivo de

la interpretación). Es por ello que los cantantes suelen añadir melismas (adornos melódicos de estructura muy barroca) sobre la melodía base que hace de su interpretación una pieza única.

La **figura 2.17** muestra la transcripción (tomada de (Mora, Gómez, Gómez, Escobar-Borrego, & Díaz-Bañez, 2010)) de dos interpretaciones de la debla En el barrio de Triana según Antonio Mairena y Chano Lobato. Es apreciable, sin la necesidad de conocimientos musicales, la gran diferencia entre ambas transcripciones.

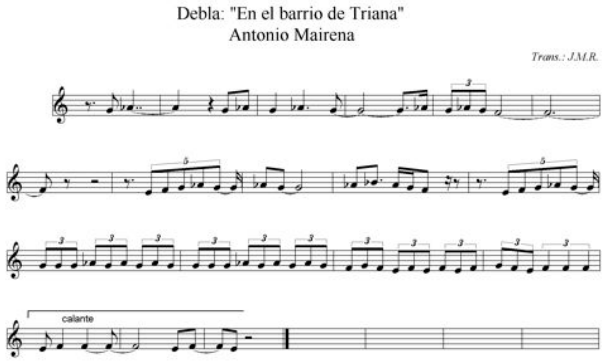


Figure 1: Manual transcription of a *cante* (debla)

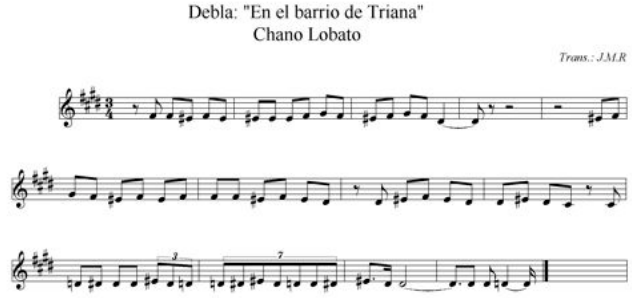


Figure 2: Transcription of another version of the previous *cante*

Figura 2.17: Transcripción dos interpretaciones de la misma pieza (Mora, Gómez, Gómez, Escobar-Borrego, & Díaz-Bañez, 2010)

Otras características de estilo (Berlanga, 2017; Guerrero, 2013; Mora, Gómez, Gómez, & Díaz-Báñez, 2016; Preciado, 1969)

Las melodías se suelen desarrollar en grados conjuntos. Esto significa que notas consecutivas en la melodía son, asimismo, notas consecutivas en la escala musical.

La ornamentación es uno de los recursos más expresivos del cante que marcan diferencias entre interpretaciones. El melisma es, quizás, el más importante y consiste en introducir más de una nota musical por sílaba de la letra.

Debido a que los intervalos (distancia entre notas consecutivas) entre notas consecutivas son pequeños, los cantaores pueden intercalar, de forma ornamental, *portamentos* (en vez de atacar una nota y posteriormente salta a la siguiente nota, el cantaor da la primera nota y luego va variando la frecuencia de la misma hasta llegar a la frecuencia de la segunda pasando por todas las intermedias). En la música occidental, el portamento suele asociarse a una mala técnica vocal del cantante. Esto no ocurre en el flamenco donde éste es un recurso estilístico usado habitualmente.

El flamenco tampoco requiere tener una tesitura muy elevada. Los cantes suelen estar en el rango de una sexta (cuatro tonos y medio) que permite ser cantado cómodamente sin necesidad de poseer una voz entrenada.

No hay que pensar que el cante flamenco es pobre en recursos expresivos; vista las limitaciones en melodías usables o las tesituras reducidas. Otros recursos disponibles en la interpretación, además del uso de los melismas y adornos ya mencionados, comprenden el uso del volumen de la voz, del expresar con el cuerpo o la capacidad de modificar su timbre.

Otras características

El flamenco no solamente es una interpretación musical. Si bien no es objeto de este trabajo analizar en profundidad las características sociales del flamenco (para ello se pueden consultar obras especializadas como (Blas Vega & Ríos Ruiz, 1988), (Navarro García & Ropero Núñez, 1995) o (Gamboa, 2005)), si se ve necesario describir algunas otras características que permitan entender la magnitud del problema que nos ocupa.

Figura 2.18: Bailarines flamencos de Picasso

El flamenco posee una personalidad y estética única e inconfundible que permite identificarlo aun sin ser un aficionado. Tan fuerte es esta estética que ha inspirado a artistas de distintas artes como la pintura (Picasso en la **figura 2.18** o Dalí (Bond, 1965)), literatura como el «Libro del Cante Jondo» de Lorca que escribe poemas a la seguiriya gitana, a la soleá, a la saeta entre otros o las conferencias que éste mismo dio sobre el cante jondo (García Lorca, 1922) y (García Lorca, 1931).

Ha sido capaz de saltar a otros medios inspirando películas como «Tarantos»[56] de Rovira-Beleta (1963) que fue nominada a los Oscars o documentales como «Flamenco» (1995) de Carlos Saura.

2.56 Posiblemente la mejor película de flamenco de todos los tiempos, en el que el baile flamenco aparece como un personaje más integrado en la narrativa.

Marca Andalucía y marca España

Pero el flamenco no es un fenómeno exclusivamente local. Éste forma parte de la marca Andalucía y la marca España a nivel internacional. Ha inspirado obras a pintores como Picabia, Matisse, Degas, Sonia y Robert Delaunay, Giorgio de Chirico o George W. Apperley. Se ha usado como banda sonora de películas de proyección internacional (como las de Pedro Almodovar) y en películas de producción foránea como las de Tony Gatlif o Mike Figgis. Ha sido usada como música de fondo en competiciones deportivas, en desfiles de moda como los de Ralph Lauren (El Nuevo Herald, 2012), usada como inspiración para músicos de otros estilos como el guitarrista Al Di Meola o la virtuosa del sitar Anoushka Shankar.

Interés económico

Además de ser fuente de inspiración de múltiples artistas, también hay un interés internacional en el consumo de este tipo de música. Los artistas flamencos cada vez salen más de gira por el extranjero y por todo el mundo hay tablaos y academias de baile, toque y cante. Como último ejemplo del creciente interés del flamenco en el extranjero citaremos el ensayo *Flamenco with a Foreign Accent* de (Jost, 1997) que destaca la penetración de la música flamenca en los Estados Unidos: desde como empezaron a llegar artistas de España hasta la aparición de artistas americanos formados específicamente allí.

El flamenco se ha incorporado a la industria de contenidos (Cerezo, 2016); pero aún requiere del desarrollo de herramientas automáticas que permitan hacerlo más accesible a un mayor conjunto de consumidores. Entre otras necesidades presentes, está la recopilación automática de metadatos o sistemas de recomendación musical por similitud que no requieran de la participación de expertos (Müllensiefen & Frieler, 2004).

Valoración UNESCO

La característica cultural y social tan identitaria del flamenco, hacen que la UNESCO afirme: «El flamenco está fuertemente enraizado en su comunidad, reforzando su identidad cultural y continúa pasandose de una generación a otra» y «Existe interés demostrado en asegurar la permanencia del flamenco». Y por ello, la UNESCO declaró al flamenco como «Obra maestra del patrimonio oral e intangible de la humanidad» inscribiéndolo en 2010 (en la reunión 5.COM) en la «Lista Representativa del Patrimonio Cultural Inmaterial de la Humanidad». Esta inscripción sirve como recordatorio de la necesidad de protección y del debido reconocimiento.

2.2.2 *Investigación del flamenco*

Historia de la evolución de la investigación sobre el flamenco.

La riqueza presente en el flamenco ha motivado su estudio desde el ángulo de diferentes disciplinas. Los primeros estudios musicológicos, efectuados a finales del siglo XIX, se llevan a cabo considerándolos como una música popular. Algunos de estos pioneros (como (Ocón y Rivas, 1874) o (Pedrell, 1891)) incluyen piezas en cancioneros de música popular en los que se intuyen el origen de los estilos actuales del flamenco.

Entrada ya la segunda mitad del siglo XX, los estudios posteriores, que ya consideran al flamenco como una entidad distinta a los cantes populares, se centran más en estudios antropológicos de los que podemos destacar a (García Matos, 1950) y (Rossy, 1966). Lamentablemente, el aspecto musical del flamenco sigue quedando de lado. Así lo menciona, referente a esta época, Guillermo Castro en el prólogo de (Castro Buendía, 2014):

> «...Sobre el origen del flamenco como género artístico existe ya mucha documentación que trata entre otros aspectos los relacionados con los artistas que tuvieron que ver con la formación del género, la transmisión de los cantes, las letras, influencia de otras artes etc., pero sobre lo más importante que es su musicalidad – características rítmicas, armónicas, aspectos melódicos, estructura evolución, etc.- no hay aún suficientes estudios...»

Sin entrar en la crudeza expresada por (Berlanga, 2017), gran parte de la bibliografía publicada sobre el flamenco hasta los años 80 (más de un siglo), ha sido escrita por aficionados bienintencionados que mezclan investigación con relatos, evidencias basadas en verdades a medias y percepciones subjetivas. El resultado es un conjunto de sobresimplificaciones sobre realidades complejas o, paradójicamente, la complicación de realidades más sencillas. La repetición indiscriminada de estas *verdades* ha terminado por calar en un conjunto de conocimiento sobre el flamenco bastante inconsistente (Caro Baroja, 1981).

Investigación del flamenco en Andalucía

Políticamente hablando, la Comunidad Autónoma de Andalucía reconoce, en el Estatuto de Autonomía (reforma de 2007), el flamenco como elemento singular del patrimonio cultural andaluz al que hay que proteger y, entre otras cosas, expresa un interés expreso en fomentar la investigación en él.

Etnomusicología computacional

Los antecedentes de los estudios computacionales de la música flamenca se remontan a 1950 con la aparición del *Computer*

Music: una serie de experimentos de los laboratorios Bell de síntesis de sonido usando computadoras. La posibilidad de realizar síntesis de sonido, planteó la posibilidad de efectuar el proceso inverso y transcribir sonido en registros en papel para su posterior análisis (Seeger, 1958). Esta herramienta de transcripción electro-mecánica llamada melógrafo, se vio de interés especial en músicas de tradición oral.

Este interés en el empleo de dispositivos electrónicos para ayudar al análisis de músicas no occidentales creció hasta la creación del término «Etnomusicología Computacional» (CE, *Computational Ethnomusicology*). Este es un término que, aunque fue acuñado en 1978 por (Halmos, Köszegi, & Mandler, 1978), es un área de investigación no del todo establecida (Tzanetakis, Kapur, Schloss, & Wright, 2007) por lo que no es raro encontrar trabajos de este área diseminadas en revistas y congresos de otras disciplinas involucradas (como matemáticas, ingeniería, musicología, etc.).

Cuando (Halmos, Köszegi, & Mandler, 1978) reflexionan sobre la CE, plantean históricamente cinco áreas principales de investigación:

- recopilación de datos,
- administración de información,
- notación,
- selección y sistematización, y
- tratamiento científico.

(Tzanetakis, Kapur, Schloss, & Wright, 2007) redefine el término como «*the design, development and usage of computer tools that have the potential to assist in ethnomusicological research*», en el que relega al análisis computacional a ser un mero instrumento al servicio de la Etnomusicología (Gómez, Herrera, & Gómez-Martin, 2013).

En la actualidad, la mayoría de los trabajos en etnomusicología computacional están centrados análisis, transcripción y procesamiento de corpus a partir de grabaciones de audio (Kroher & Gómez, 2016) y el análisis de patrones de ejecución y gestuales del intérprete (Giraldo, et al., 2016; Zinemanas, Arias, Haro, & Gómez, In Press) con animo de preservación de la música, ayudar en la enseñanza de estas piezas o instrumentos musicales o identificación del intérprete.

MIR

Recientemente (menos de 20 años), ha surgido una nueva área de investigación en música denominada MIR (*Music Information Retrieval*) centrada en el estudio de grandes corpus musicales[57]

[2.57] Desde el punto de las tecnologías de la información, el MIR es equivalente a *Data Mining* sobre colecciones de piezas musicales.

(Orio, et al., 2006), propiciada por la alta disponibilidad de muestras sonoras y de equipos con capacidad de procesarlos. La propia naturaleza de la música hace que el MIR sea una disciplina heterogénea atacada desde distintos puntos de vista como las tecnologías de la información, ingenierías, matemáticas, archivística, musicología o incluso psicología.

Si bien existe un conjunto de tecnologías MIR muy maduras, como pueden ser

- sistemas de recomendación musical,
- sistemas de radio personalizados,
- consulta por tarareo/ritmo (*query by humming/tapping*) (Ghias, Logan, Chamberlin, & Smith, 1995; Jang, Lee, & Yeh, 2001),
- seguimiento automático de partituras (*score following*) (Dannenberg & Raphael, 2006),
- identificación de audio o *fingerprinting* (Wang, 2006) y
- identificación de versiones (*covers*) (Kim, Unal, & Narayanan, 2008)

estas funcionan especialmente en música occidental, aunque no están tan maduras en el campo de la etnomusicología (Tzanetakis, 2014).

Dentro del campo de la etnomusicología, los sistemas existentes no están muy depurados requiriendo conocimientos técnicos tanto para su configuración y como su uso. Estas limitaciones lo descarta, de momento, para un uso general. Sin embargo, es indudable que proporciona a los investigadores ventajas al facilitar el procesado masivo de piezas que, aunque en casos puedan tener menor precisión que el realizado manualmente por un experto[58], permiten trabajar con gran cantidad de información.

[2.58] Como el caso de una transcripción.

Aun con el creciente interés existente en la etnomusicología, todavía hay mucho camino que recorrer. El estudio efectuado por (Cornelis, Lesaffre, Moelants, & Leman, 2010) mostró que, en el campo del MIR y el periodo desde el 2000 hasta el 2008, solo un 5.5% de los artículos publicados en el área eran trabajos realizados sobre etnomusicología.

Es por ello que (Tzanetakis, 2014) expresa que el futuro de la etnomusicología pasa por:

– La necesidad de colaboración entre expertos y técnicos para crear un campo de investigación interdisciplinar,

– la existencia de grandes colección de audio anotadas por expertos y

– Desarrollo de técnicas de dominio específico y de aplicaciones de acceso a usuarios no técnicos ni especialistas.

A continuación, se mencionarán las líneas más actuales abiertas en el campo MIR.

*Extracción automática de alturas (*pitch*) y entonación (*tunning*)*

Como ya comentamos, en la música occidental, los valores de las frecuencias están cuantizados siendo el semitono la distancia mínima entre notas. Esto no sucede necesariamente en otro tipo de tradiciones musicales como las situadas en el norte de la India o el Flamenco en el que se pueden usar intervalos inferiores al semitono.

El problema que plantean estas otras tradiciones es determinar si la aparición de un subsemitono es intencional o si el interprete simplemente desafinó. Para ello es necesario el desarrollo de una teoría musical que defina nuevas escalas[59], los intervalos que pueden aparecer y que marque las diferencias entre notas de la melodía y meros ornamentos.

[2.59] Una relación de notas y frecuencias usables en un fragmento.

Un ejemplo de trabajos en este sentido es la música Makam de Turquía que presenta diversas teorías musicales con hasta 79 notas distintas para describirla (Bozkurt, Yarman, Karaosmanoğlu, & Akkoç, 2009; Yöre, 2012).

Computacionalmente hablando, las herramientas existentes efectúan análisis de frecuencias mediante histogramas del *pitch* de las notas. Un ejemplo de este tipo de software es Tarsos (Six & Cornelis, 2011) que efectúa visualización de tónicas, histograma de alturas, seguimiento de partituras y otros en el contexto de músicas no occidentales.

Análisis del ritmo

Uno de los aspectos más estudiados en etnomusicología es el ritmo: la extracción de patrones rítmicos, la desviación de la interpretación de un tiempo patrón o criterios de similitud rítmica (este último es uno de los pilares para los sistemas de recomendación musical).

Distintos estudios se han realizado para músicas no occidentales como griega y africana (Antonopoulos, et al., 2007), turca e india (Holzapfel & Stylianou, 2009; Srinivasamurthy, Holzapfel, & Serra, 2014), flamenco (Guastavino, Gomez, Toussaint, Marandola, & Gomez, 2009) o música latina (Völkel, Abeßer, Dittmar, & Großmann, 2010).

A pesar de ser un problema muy estudiado, siguen existiendo casos de difícil análisis derivados de la complejidad rítmica de los sistemas orales. Uno de ellos es la música afrocubana en el que el patrón rítmico base (llamado clave) tiene variaciones de tiempo respecto a un patrón difíciles de estimar (Wright, Schloss, & Tzanetakis, 2008).

Métodos automáticos de transcripción

La forma principal de convervación y almacenamiento de piezas musicales (y más en las música de tradición oral) es la grabación de audio. En las grabaciones, los datos almacenados son de tipo acústico en los que, sin considerar la codificación final en el dispositivo físico, se consideran frecuencias, amplitudes de señales sonoras.

El proceso de transcripción engloba un proceso de abstracción en la que se descarta información no relevante[60] y la construcción de una representación simbólica útil que contenga la que si se considere.

[2.60] Como podrían ser consideraciones tímbricas.

El proceso de obtención la representación simbólica útil de una pieza, o transcripción automática, es un proceso complejo que engloba diferentes tareas[61] (Benetos, Dixon, Giannoulis, Kirchhoff, & Klapuri, 2013): estimación del *pitch*, determinación del inicio, frecuencia y duración de las notas, estimación del volumen, reconocimiento del instrumento, extracción de información rítmica y cuantización temporal.

[2.61] Obviamente, según los objetivos buscados no es necesario resolverlas todas.

Según (Benetos, Dixon, Giannoulis, Kirchhoff, & Klapuri, 2013), la estimación del *pitch* de grabaciones en las que aparece sonido monofónico (una única nota musical cada vez) es un problema resuelto siendo las corrientes actuales de investigación en el área la determinación de la melodía principal de una grabación (en las que hay presentes varias fuentes de sonido) como en (Gómez, et al., 2012) o la determinación simultánea *multi-pitch* en la que se determina el *pitch* de cada fuente por separado (Adalbjörnsson, Jakobsson, & Christensen, 2015).

Comparación de melodías

Muchos de los procedimiento de análisis se basan en un mecanismo de comparación o medida de similitud entre piezas. Inicialmente, la mayoría de medidas se han centrado en un aspecto concreto de las melodías (el ritmo, las notas, ...), pero en la actualidad se buscan funciones multiobjetivo que encuentren el equilibrio entre estos distintos aspectos.

Una de las características más empleadas es el contorno melódico (Hewlett & Selfridge-Field, 1998), aunque sigue siendo un tema candente que ha dado muchas publicaciones los últimos veinte años (Kroher, Gómez, Guastavino, Gómez, & Bonada, 2014; Müllensiefen & Frieler, 2004; Volk, et al., 2007).

Recuperación basada en contenido

La existencia de bases de datos de piezas cada vez mayores, impone la necesidad de proporcionar métodos de búsqueda de alto nivel más allá del empleo de búsqueda por metadatos (título, autor, intérprete, ...).

Algunos ejemplos de recuperación por contenido son el *Query-by-Humming* (Consulta por tarareo) de (Ghias, Logan, Chamberlin, & Smith, 1995) o la identificación de fragmentos musicales comerciales como Shazam (Wang, 2006).

Sistemas de contexto e interfaces de contenido

Un segundo problema asociado a la existencias de grandes bases de datos musicales está en la construcción de un sistema relacional en el que cada pieza pueda enmarcarse en un contexto de otras piezas del mismo estilo, época, autor o localización (Magas & Proutskova, 2013) (por citar algunas).

La construcción de interfaces de navegación adecuados para moverse por este hiperespacio de características ofrece una ayuda inestimable de contexto tanto a especialistas como a oyentes no familiarizados con el material (Mcdonald, 2017; Porter, Sordo, & Serra, 2013).

Herramientas de deformación de piezas musicales

El análisis y conservación de las piezas en la etnomusicología, ha pasado por distintas etapas. Inicialmente, hace más de 150 años, los modelos melódicos y ritmos se preservaban por medio

de la tradición oral. Era un método rico que en cada transmisión añadía o quitaba elementos en las piezas[62].

Una segunda etapa se desarrolló en el último tercio del siglo XIX cuando parecieron los primeros fonógrafos. En esta etapa se inició el registro sonoro de las piezas que nos proporcionó una muestra estática de las mismas.

Las herramientas de deformación de registros sonoros, propiciado por los sistemas informáticos como (Driedger & Müller, 2016), han permitido modificar algunas propiedades de los registros con objeto de comparar la similitud de unas piezas con otras que de otra forma no hubiera sido posible. Algunas de estas modificaciones son: alterar el *tempo*[63] de una pieza sin alterar la afinación, alterar la afinación sin alterar la velocidad, alinear patrones rítmicos entre piezas, distorsionar solo una parte de la misma, alterar el volumen relativo de un instrumento frente a otros, etc.

Análisis gestual de una interpretación

Existen otros mecanismos relacionados con la producción de música que no son acústicos. Uno de ellos es el trabajo corporal del interprete. Al igual que ocurre en el entrenamiento deportivo de élite (Dallinga, et al., 2017), la grabación en vídeo de intérpretes aporta información importante en el campo de la pedagogía y la investigación musical.

Ya entrados en el siglo XXI, se ha ampliado el repertorio de registros gestuales por medio de captura en 3D (Togootogtokh, et al., 2017), sensores en los instrumentos (Benning, Kapur, Till, & Tzanetakis, 2007), sensores sobre los intérpretes (Al Kork, et al., 2014) y otros.

2.2.3 *Lineas de estudios actuales en flamenco y MIR*

Si bien el flamenco presenta un calado importante en la sociedad española (tanto desde un punto de vista cultural como comercial) y una gran proyección internacional, no ha sido objeto de estudio computacional hasta hace relativamente poco tiempo.

El primer estudio de música flamenca computacional fue publicado en 2004 (Díaz-Bañez, Farigu, Gómez, Rappaport, & Toussaint, 2004)[64]. En este trabajo se aplican técnicas de análisis matemático sobre patrones rítmicos en el flamenco, abriendo todo un nuevo conjunto de herramientas de análisis nunca empleadas

[62] Fenómeno que no se daba en la música occidental ya que las partituras fijaban, en gran manera, la intención del autor.

[63] La velocidad

[64] Una versión española se publicó en (Díaz-Bañez, Farigu, Toussaint, Gómez, & Rappaport, 2005).

en este tipo de música. Además, propone un enfoque de trabajo multidisciplinar que servirá de germen del grupo COFLA de investigación (COFLA, 2017).

El proyecto COFLA (*Computational analysis of FLAmenco Music*) comienza en 2007 como el primer proyecto de investigación universitario, con subvención pública, relacionado con el flamenco. Entre sus objetivos, está la defensa del flamenco como campo de estudio académico universitario. Más concretamente, el estudio de sus estructuras musicales, sus orígenes, la evolución y relación entre estilos, las propiedades de estos y la detección de influencias a y desde otras manifestaciones musicales. Para ello, emplea herramientas tecnológicas de procesado de audio, modelado computacional y técnicas de aprendizaje automático que contrasta con información extraída de otras áreas de conocimiento como son la historia, archivística, literatura, musicología o psicología. El proyecto COFLA está en activo con financiación hasta 2019.

Se debe citar un segundo grupo de investigación que, aunque no trabaje exclusivamente el flamenco, si que trabaja en etnomusicología computacional desarrollando herramientas útiles para el estudio del flamenco. El grupo en cuestión es el *Music Technology Group* (Music Technology Group, 2017) de la Universidad Pompeu Fabra de Barcelona fundado por Xavier Serra y cuyo investigador principal en temas de MIR es Emilia Gómez.

Las líneas actuales de investigación MIR sobre flamenco son las promovidas, principalmente, por los grupos COFLA y MTG. Estas se pueden resumir en:

Transcripción automática, segmentación y separación de voces
Con objeto de poder analizar un elevado número de piezas, es necesario disponer de herramientas automáticas de transcripción (AMT, *Automatic Music Transcription*). La transcripción manual es costosa de realización y sus resultados son dependientes del transcriptor que las efectúe.

Un proceso de transcripción automático y óptimo debe proporcionar una descripción simbólica propia para cada interviniente en la grabación, separando cantantes y acompañamiento.

Dentro del mismo problema de análisis para la realización de las transcripciones, está el subproblema de la segmentación del audio en el que se etiquetan regiones del mismo en función de su contenido (regiones de silencio, falsetas de guitarra, cantante, identificadores de tercio, etc.) (Herrera, Serra, & Peeters, 1999).

En este campo de transcripción automática, existen una gran cantidad de enfoques y utilidades destinadas a la misma. El congreso MIREX (*Music Information Retrieval Evaluation eXchange*) (Downie, 2008; Downie, Ehmann, Bay, & Jones, 2010) es un encuentro anual de investigadores del MIR que presenta anualmente una lista de lineas abiertas de investigación (entre las que se encuentra la transcripción automática) y una evaluación de los distintos algoritmos y sistemas presentados.

Estos algoritmos no tienen un desempeño demasiado bueno en músicas que presentan el problema añadido de las microtonalidades como el flamenco o la música Makam turca (Benetos & Holzapfel, 2015). Es por ello que la línea actual de trabajo pasa por algoritmos específicos como el proporcionado por la librería `CANTE` (Kroher & Gómez, 2016) y sistemas de transcripción asistida[65] como el que proporciona el programa `TONY` (Mauch, et al., 2015).

[65] Si bien no es un objetivo principal de esta tesis, entre otros trabajos desarrollados está una aplicación completa de asistencia a la transcripción presentada en el **apéndice B**.

Creación de un corpus musical Ya sea en forma de audios (más o menos etiquetados) o como transcripciones, es necesario disponer de colecciones comunes de piezas sobre los que los distintos investigadores puedan trabajar.

Estas colecciones deben cumplir con unos criterios de representabilidad, en las que las piezas recogidas deben de ser un muestreo adecuado del campo examinado, y han de ser accesibles para todos los investigadores.

En este sentido, existen dos corpus de piezas flamencas publicadas: el corpus TONAS (Mora, Gómez, Gómez, & Díaz-Báñez, 2016) que incluye una selección de 72 audios de martinetes y deblas con transcripciones efectuadas de forma semiautomática y el corpus COFLA (Kroher, Díaz-Báñez, Mora, & Gómez, 2016) con una base de datos de más de 1800 fragmentos de audio tomados de distintas antologías comerciales del flamenco.

Detección automática de patrones distintivos [66]

La identificación por expertos de un estilo musical, y más en músicas de tradición oral, pasa por un proceso de reconocimiento de ciertos patrones característicos en cada estilo. Patrones que en ocasiones aparecen tal cual en las piezas (como puede ser la ya mencionada cadencia andaluza) y que, en otras ocasiones, pueden ser meras estructuras melódicas que al cantarse se entremezclan con otros motivos decorativos o adornos.

Entre las distintas metodologías existentes, los enfoques más comunes son el método inductivo (que analiza la música para

[66] En esta línea se engloba el **objetivo tercero** de la presente tesis.

descubrir dichos patrones) y el método deductivo (que parte de un conjunto de patrones previamente seleccionados por un experto y que se buscan dentro de un corpus).

En el campo del flamenco, la búsqueda de patrones para la caracterización de los estilos es un tema sobre el que se le ha prestado escasa atención. Destacamos los trabajos del investigador Pikrakis en el que usa el método deductivo para detectar frases melódicas en fandangos de Valverde (Pikrakis, et al., 2012) y su algoritmo de alineación de patrones canónicos sobre melodías automáticamente transcritas (siendo más o menos robustos a silencios y ornamentaciones) (Pikrakis, Kroher, & Díaz-Báñez, 2016).

Detección automática de ornamentos El solapamiento presente entre la melodía flamenca y los adornos plantean una linea de investigación centrada en estos últimos. ¿Qué patrón melódico es intrínseco a la melodía y cuál al adorno? ¿Qué adornos son específicos de cada cantaor? ¿Existen adornos específicos de un estilo?

El estudio completo de los adornos permiten separar el arquetipo melódico base de las alteraciones interpretativas.

Existen distintos modelos, no específicos al flamenco, que analizan los adornos en música de fagot (Puiggròs, Gómez, Ramírez, Serra, & Bresin, 2006), violín (Perez, Maestre, Kersten, & Ramirez, 2008), gaita (Menzies & McPherson, 2015) o guitarra (en jazz) (Giraldo & Ramírez, 2016); pero, nuevamente, no existe una gran línea de trabajo de adornos en el flamenco (la línea la abre y, casi la cierra, el artículo (Gómez, et al., 2011)).

Análisis de la voz flamenca La técnica vocal empleada en el cante flamenco es, aparentemente, distinta a otras técnicas vocales empleadas en música lírica o popular. Se impone una racionalización sobre estas técnicas intentando responder si realmente la voz en el cante flamenco es distinta a la de otros estilos musicales y qué características distintivas tiene esta voz (Mora, 2013).

El análisis de cadenas derivadas de modelos desconocidos propone trabajar sobre dos líneas de investigación dentro del campo de la etnomusicología computacional:

Lineas de trabajo de la presente tesis.

– La medida de la similitud entre piezas y

– la detección automática de patrones distintivos (discriminantes y estructurales).

entendiendo que el crecimiento del conocimiento de las estructuras profundas presentes en las piezas es un trabajo base sobre el que se pueden apoyar otras investigaciones posteriores.

2.3 Conclusión

El principio *No free lunch* expone que la mejora de rendimiento en una tarea de aprendizaje viene por dos caminos: encontrar un algoritmo específico para el problema que se está tratando o tener la capacidad de proporcionar al algoritmo usado datos más ricos en información. Uno de los principios rectores del presente trabajo de tesis es proporcionar herramientas genéricas, por lo que se ha optado por seguir la segunda vía expuesta.

En este capítulo se ha expuesto una división de tareas original en el que la calidad de los datos suministrados depende de una fase previa de diseño denominada «fase de adecuación» y se ha efectuado un extenso repaso a distintas técnicas que entrarían dentro de esta fase.

Posteriormente se ha presentado un problema específico, el análisis de cantes flamencos, que será usado como banco de pruebas de las herramientas que se desarrollan en el siguiente capítulo.

3 Herramientas de adecuación de cadenas

El principio *No Free Lunch* afirmaba que el rendimiento de los algoritmos de aprendizaje automático era de suma cero. Tal y como se expuso, esto implicaba que la mejora del rendimiento en los procesos de aprendizaje debía basarse en la elección de técnicas específicas de aprendizaje para cada problema o en la mejora de la calidad de los datos suministrados a dichos sistemas de aprendizaje. En este trabajo se ha decidido trabajar sobre la segunda opción con el ánimo de no sacrificar la generalidad de las opciones propuestas.

Estructuralmente, siguiendo la idea de dividir el análisis de los problemas en adecuación de los datos y aplicación de los algoritmos, se ha dividido las operaciones a efectuar siguiendo una sistema modular en dos niveles de abstracción (mostrados en la figura **figura 3.1**). El nivel más bajo, y abstracto, comprende un conjunto de técnicas de adecuación de cadenas que serán empleadas en la capa de aplicación (el nivel superior de la pila). Esta estructura modular aísla las operaciones de adecuación (las que se describen en este capítulo) de los posibles algoritmos de análisis aplicables y favorece la reutilización de las herramientas en futuras investigaciones.

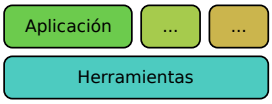


Figura 3.1: Pila de tareas de esta tesis

Adicionalmente, y dada la naturaleza abstracta de las herramientas presentadas, encontrar un problema que empleando estas herramientas desarrolladas dé resultados satisfactorios tendrá la doble utilidad de mostrar la validez de estas y aumentar el conocimiento en el problema planteado.

En el presente capítulo se presentan las técnicas genéricas propuestas en este trabajo, dejando la aplicación, en el campo de pruebas escogido, para el próximo capítulo.

Adecuación y Aplicación

Al inicio del estado del arte (descrito en el **capítulo 2**), se presentó una organización original de los algoritmos de aprendizaje automático que se resume en la **tabla 2.1** y que se repite aquí para su fácil acceso:

1ª Etapa (Adecuación)	2ª Etapa (Aplicación)
Punto de un espacio métrico	Agrupación
Punto de un espacio de características	Reconocimiento de patrones discriminantes
Grafo	Análisis de grafos

Tabla 3.1: (Repetición **tabla 2.1**) Descripciones normalizadas para distintas técnicas de Análisis

En esta tabla proponemos que la aplicación de sistemas de aprendizaje automático son realmente dos etapas conceptuales. La primera etapa se encarga del diseño de la adaptación de los datos a las necesidades del algoritmo concreto de aprendizaje (que corresponde a la segunda etapa). Aunque pudiera parecer que la primera etapa coincide con la finalidad del preprocesado (ya que parece que existe un cierto solapamiento entre sus tareas), sus objetivos y formas de actuación son diferentes.

El preprocesado es una fase ejecutiva de aplicación fundamentalmente local. Sus dos objetivos fundamentales son el limpiado y la conformación de los datos (Kotsiantis, Kanellopoulos, & Pintelas, 2006). En el primero se pretende identificar y solucionar los problemas ocurridos en la fase de captura de datos y en el segundo manipular la forma de los datos para que sean adecuados a la siguiente etapa del procesamiento. En ambos casos, el preprocesado se aplica fundamentalmente instancia a instancia.

En cambio, la adecuación es una fase de diseño de aplicación global. No se centra en cada instancia disponible sino que se preocupa de las operaciones requeridas para cada tipo de datos dato. De los ejemplos de adecuación ya dados, el más ilustrativo es la definición de una métrica para cadenas en el que las instancias no se modifican y simplemente se le dan nuevas habilidades[1] al tipo cadena. Igualmente, la adecuación se encarga de establecer la estrategia de conformado más adecuada que, posteriormente se aplicará durante el preprocesado.

[3.1] La idea no es nueva en otros ámbitos. En sistemas de programación orientada a objetos, el concepto se llama *traits* (Ducasse, Nierstrasz, Schärli, Wuyts, & Black, 2006)

Lo importante de esta división entre adecuación y aplicación estriba en la capacidad de reutilización de datos y algoritmos en nuevas situaciones. Un algoritmo como la agrupación se debe considerar como un par formado por una métrica (operación de adecuación) y el algoritmo en sí (la aplicación). Cualquier objeto sobre el que se defina una métrica puede usar cualquier algoritmo de agrupación que emplee métricas (por ejemplo *k-means*). Asimismo, cualquier nuevo algoritmo de agrupación, si se basa en métricas, puede aplicarse sobre cualquier objeto o campo

que tenga dichas métricas definidas. Esta capacidad de sustitución de objetos o algoritmos que se presenta con este enfoque no aparece cuando hablamos del par preprocesado-procesado que es más dependiente de los datos iniciales y el procesado posterior.

Presentación del resto del capítulo

El resto del capítulo se dedica a describir tres grupos de herramientas originales para el análisis de cadenas. La primera define una nueva métrica sobre cadenas. La nueva métrica tiene como producto derivado la construcción de una cadena inicial u origen (que llamaremos «centroide») que proporciona una hipótesis de la forma de una cadena a partir de la cual se derivaron las evaluadas.

El segundo conjunto herramientas está dentro de la ingeniería de características y es un conjunto de funciones generadoras de descriptores para cadenas. Este conjunto se describen en categorías en función de las características de las cadenas. Además se discuten ciertas estrategias meta-descriptor que permite incrementar el espacio dimensional de descripción de las cadenas.

El tercer conjunto de herramientas muestra un sistema de conformado de cadenas en grafos y utilizando las características de estos, presenta un sistema deductivo de extracción de patrones estructurales con hueco.

Estos tres conjuntos de técnicas se engloban dentro de las operaciones de adecuación y abren gran cantidad de posibilidades de análisis de cadenas.

3.1 Distancia media al centroide DMC

La «distancia media al centroide» DMC es una metamétrica que proporciona una medida de dispersión entre puntos de un espacio métrico. Esta construcción se comporta como una generalización de la métrica sobre la que se basa ya que no está limitada a medir exclusivamente distancias entre dos puntos.

En el resto de la sección, se efectuará una descripción formal de dicha generalización, su aplicación sobre cadenas, se proporcionará una interpretación de la información que proporciona y se finalizará con algunas consideraciones sobre su implementación.

3.1.1 Definición formal

Sea el conjunto $S = \{x_i : x_i \in E\}$ un subconjunto del espacio métrico E (con métrica d).

A partir de la métrica d es posible definir una medida de dispersión de un conjunto de puntos (respecto a otro punto) como la media de las distancias a dicho punto:

Distancia media de un punto p a un conjunto S como

$$\mathrm{d}_p(S) = \frac{\sum_i^{|S|} \mathrm{d}(p,x_i)}{|S|} \tag{3.1}$$

$\mathrm{d}_p(\{x,y\})$ es una métrica

Si el conjunto S está compuesto por sólo dos elementos, la distancia media es una métrica. Como todas las demás métricas, ésta ha de cumplir los requisitos que se enumeran a continuación:

1. $\mathrm{d}_p(\{x,y\}) \geq 0$ (no-negatividad).

   Dado que la distancia al centroide es la media de valores no negativos (obtenidos por medio de otra métrica), esta siempre será no negativa.

2. $\mathrm{d}_p(\{x,y\}) = 0 \rightarrow x = y$ (identidad de los indiscernibles).

   Para que la media de dos números no negativos sea 0, es necesario que ambos números sean 0. Por la propiedad de la métrica subyacente, se tiene que: $\mathrm{d}(p,x) = 0 \rightarrow p = x$ y $\mathrm{d}(p,y) = 0 \rightarrow p = y$, por lo tanto $x = y$.

3. $\mathrm{d}_p(\{x,y\}) = \mathrm{d}_p(\{y,x\})$ (simetría).

   La media de un conjunto de números es independiente del orden de los términos sumados por lo que $\mathrm{d}_p(S)$ es simétrico aunque d no lo sea[2].

[3.2] Una función que cumpla con todos los requisitos de una métrica salvo la simetría se denomina cuasimétrica (Xia, 2009) y representan espacios en los que el esfuerzo de ir de un punto a otro varía del sentido en el que se mide la distancia.

4. $\mathrm{d}_p(\{x,z\}) \leq \mathrm{d}_p(\{x,y\}) + \mathrm{d}_p(\{y,z\})$ (desigualdad triangular).
   Como se ilustra en la **figura 3.2**,

$$\mathrm{d}_p(\{x,z\}) = \frac{1}{2}(\mathrm{d}(p,x) + \mathrm{d}(p,z)) \tag{3.2}$$

   y

$$\begin{aligned}\mathrm{d}_p(\{x,y\}) + \mathrm{d}_p(\{y,z\}) &= \frac{1}{2}(\mathrm{d}(p,x) + \mathrm{d}(p,y)) + \frac{1}{2}(\mathrm{d}(p,y) + \mathrm{d}(p,z)) \\ &= \frac{1}{2}(\mathrm{d}(p,x) + \mathrm{d}(p,z)) + \mathrm{d}(p,y) \\ &= \mathrm{d}_p(\{x,z\}) + \mathrm{d}(p,y) \end{aligned} \tag{3.3}$$

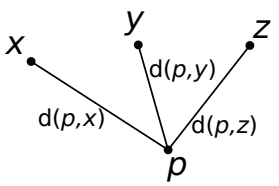


Figura 3.2: Desigualdad triangular

   Comparando las expresiones **3.2** y **3.3** se confirma el cumplimiento de la desigualdad triangular siendo los dos términos iguales cuando el punto y coincide con p.

Aunque se ha demostrado que $d_p(\{x,y\})$ es una métrica para cualquier valor de p, no todos los posibles valores poseen la misma utilidad. La distancia medida desde un punto p situado lejos de x e y, proporcionará fundamentalmente la distancia entre p y el par x e y. En cambio, si p está situado en el punto intermedio del segmento que une a x e y, se verifica que la distancia:

Distancia media al centroide

$$d_p(\{x,y\}) = \frac{1}{2} d(x,y) \tag{3.4}$$

que la distancia media es la mitad de la distancia de la métrica original. Siendo, por tanto, una buena medida de la dispersión entre los puntos.

Una generalización para más puntos del concepto del punto medio entre dos es el centro de masas o centroide que se define como

Centroide de S

$$c \in E \quad / \quad \sum_{i}^{|S|} d^2(c,i) \text{ es mínimo} \tag{3.5}$$

todo punto c del espacio métrico que minimiza la suma de distancias (al cuadrado) entre dicho punto c y cada elemento de S.

Cuando S está compuesto de dos elementos solamente, esta definición coincide con el punto intermedio del segmento que los une.

Una vez definido el concepto de centroide es posible definir la distancia media a este:

Distancia media al centroide de S (DMC) como la distancia media del centroide de S a S y lo denotamos:

$$d_c(S) \tag{3.6}$$

El cálculo de la distancia media al centroide de un conjunto proporciona simultáneamente dos informaciones sobre dicho conjunto. Proporciona una medida de dispersión de los puntos del conjunto analizado. Además, el cálculo de dicha distancia media requiere del cálculo del centroide de los puntos que, de alguna manera, representa a los miembros de dicho conjunto[3].

Las características del centroide y su relación con los puntos del conjunto S dependerán de la métrica base empleada.

[3.3] Al igual que el centro de masas puede representar a un cuerpo.

3.1.2 *Especialización en cadenas*

La aplicación de la distancia media al centroide al espacio de las cadenas (A^*) solo requiere escoger una métrica sobre la que basarse. El uso de una métrica u otra determinará, como se acaba de comentar, las características del centroide hallado.

En esta sección se discutirá el empleo de las dos métricas más populares usadas en el espacio de cadenas: la distancia de edición y los n-gramas, y se proporcionará una interpretación del centroide obtenido.

Dado que el cálculo del centroide, a partir de una métrica, no es una tarea trivial, se ha retrasado los detalles de la implementación de su calculo a una sección propia, a continuación de esta.

Distancia de edición

La distancia proporcionada por las métricas de distancia de edición depende de los costes asociados a cada operación de transformación. Asignar un coste de uso muy elevado a una operación es equivalente a prohibir el uso dicha operación. Análogamente, un coste reducido fomentará su uso cuando sea posible.

Calibración de la distancia de edición

Con el fin de determinar los costes más adecuados para la obtención de centroides, se ha formulado un problema tipo (cuyo resultado deseado es conocido) y se ha usado como evaluador de la bondad de la métrica propuesta. La **figura 3.3** muestra el conjunto S formado por tres cadenas distintas. Si entendemos estas cadenas como derivadas de un modelo común (y desconocido), las partes comunes en las cadenas de S son más probables que pertenezcan al modelo original. De la zona de discrepancia (marcada con fondo amarillo) no se tiene ninguna información pudiendo, el modelo original, contener cualquiera de los símbolos marcados en amarillo, un símbolo distinto a estos, una combinación de símbolos de longitud desconocida o, simplemente, no tener ningún símbolo en el lugar marcado.

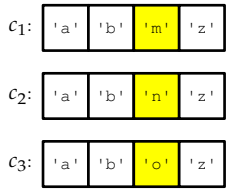


Figura 3.3: Conjunto de cadenas $S = \{c_1, c_2, c_3\}$

Un buen candidato a centroide sería aquel que el contenido del centroide se base en los elementos comunes de las cadenas de S. Asociando a la idea del centroide como el máximo común entre cadenas al que se le añaden los elementos discrepantes. Para el ejemplo dado, el centroide ideal que proponemos se muestra en la **figura 3.4**. Aunque no podamos afirmar que el centroide es el modelo base de las cadenas, planteamos la hipótesis que es una buena aproximación a este.

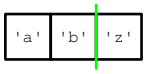


Figura 3.4: Centroide ideal del conjunto de la **figura 3.3**

Al ensayar el cálculo del centroide usando la distancia Levenshtein (d_l), obtenemos que las cadenas c_1, c_2 y c_3 son mejores candidatos a centroide que nuestro candidato c_i.

$$\begin{aligned} \text{Evaluacion}(c_1) &= d_l^2(c_1,c_1) + d_l^2(c_1,c_2) + d_l^2(c_1,c_3) \\ &= 0^2 + 1^2 + 1^2 = 2 \end{aligned} \tag{3.7}$$

$$\text{Evaluacion}(c_1) = \text{Evaluacion}(c_2) = \text{Evaluacion}(c_3) \tag{3.8}$$

$$\begin{aligned} \text{Evaluacion}(c_i) &= d_l^2(c_i,c_1) + d_l^2(c_i,c_2) + d_l^2(c_i,c_3) \\ &= 1^2 + 1^2 + 1^2 = 3 \end{aligned} \tag{3.9}$$

Observamos que la métrica Levenshtein (sin modificación) presenta dos problemas. El primero es que nuestro candidato a centroide puntúa peor que las cadenas c_1, c_2 y c_3, por lo que descartan a c_i como centroide. Nuestro centroide se basa en un máximo común entre cadenas al que se le añaden las partes discrepantes. Dado que la métrica Levenshtein original asigna el mismo peso a la incorporación de elementos al centroide como a su eliminación o sustitución, estos pesos no son idóneos para la obtención de nuestro centroide.

Como solución a este problema, es necesario primar la operación de inserción al centroide frente a otras que eliminen o sustituyan elementos de este.

Un segundo problema, ya menor, es que las cadenas c_1, c_2 y c_3 tienen todas la misma evaluación. A falta de mejores candidatos, esto haría que todas ellas deban ser consideradas como centroide del conjunto. Aunque la existencia de más de un centroide no tendría que ser un problema para posteriores operaciones, lo ideal sería que el centroide fuera un punto único.

Aunque no es posible evitar la aparición de múltiples centroides sobre un conjunto de cadenas, es posible asignar un mecanismo de desempate sencillo que no altere la métrica significativamente.

Ambos problemas pueden ser solucionados alterando los costes de operación de las transformaciones, primando el coste de inserción (frente a la sustitución y eliminación). La **tabla 3.2** muestra los pesos propuestos. Puede observarse que los costes de operaciones antagónicas no son idénticos, la distancia de edición considerada no es, por tanto, una métrica sino una quasimétrica; aunque, tal y como se demostró en la **condición para métrica nº 3**, la no simetría de la métrica base no afecta a la simetría de la distancia media al centroide que sigue manteniendose.

Operación	Coste
Inserción	1,0
Eliminación	1,5
Sustitución	2,1
Aproximación	0,01

Tabla 3.2: Operaciones y costes

El coste por sustitución, que se basa en la mejora de Sellers (Sellers, 1974), es elevado aunque cumple la relación

$$coste_{sustitución} < coste_{adición} + coste_{eliminación} \tag{3.10}$$

que no impide que sea usada si es necesario. Además, con objeto de desempatar entre posibles candidatos a centroide, el coste por sustitución incorpora un coste variable proporcional a la diferencia entre los símbolos sustituidos. La constante de proporcionalidad aplicada (que se ha denominado «Aproximación») es dos órdenes de magnitud inferior al coste de inserción por lo que solo se apreciará su efecto en los casos de empate entre dos posibles centroides.

El cálculo de la aproximación requiere que exista una conversión establecida entre los símbolos del alfabeto sobre el que se construyen las cadenas y los números reales. Esta imposición no es demasiado restrictiva ya que siempre se puede asignar un número de orden a los elementos del alfabeto que se emplee.

Reevaluando los candidatos a centroide con los nuevos costes,

$$\begin{aligned} \text{Evaluacion}(c_1) &= \mathrm{d}_\mathrm{l}^2(c_1,c_1) + \mathrm{d}_\mathrm{l}^2(c_1,c_2) + \mathrm{d}_\mathrm{l}^2(c_1,c_3) \\ &= 0^2 + (2.1 + 0.01)^2 + (2.1 + 0.02)^2 \\ &= 8.9465 \end{aligned} \tag{3.11}$$

$$\begin{aligned} \text{Evaluacion}(c_2) &= (2.1 + 0.01)^2 + 0^2 + (2.1 + 0.01)^2 \\ &= 8.9042 \end{aligned} \tag{3.12}$$

$$\begin{aligned} \text{Evaluacion}(c_3) &= (2.1 + 0.02)^2 + (2.1 + 0.01)^2 + 0^2 \\ &= 8.9465 \end{aligned} \tag{3.13}$$

$$\begin{aligned} \text{Evaluacion}(c_i) &= \mathrm{d}_\mathrm{l}^2(c_i,c_1) + \mathrm{d}_\mathrm{l}^2(c_i,c_2) + \mathrm{d}_\mathrm{l}^2(c_i,c_3) \\ &= 1^2 + 1^2 + 1^2 = 3 \end{aligned} \tag{3.14}$$

La distancia de edición propuesta soluciona los dos problemas que planteaba la distancia Levenshtein: escoge el candidato deseado y establece un sistema de desempate entre candidatos (donde había un triple empate, ahora hay un candidato mejor que deshace el empate). Por estos motivos, consideramos que los pesos de la **tabla 3.2** son más apropiados en la determinación del centroide.

La **figura 3.5** muestra el resultado del cálculo del centroide aplicado a tres cadenas extraídas del corpus de tonás[4]. Bajo cada cadena se muestran unas líneas que identifican las subcadenas comunes entre el centroide (identificado como «C») y las cadenas 1, 2 y 3. El color de cada línea indica con qué cadena se comparte la subcadena y el número sobre la línea la posición de inicio de la subcadena en dicha cadena.

Ejemplo centroide con datos reales

3.4 Que se usará en el capítulo de aplicación de estas técnicas

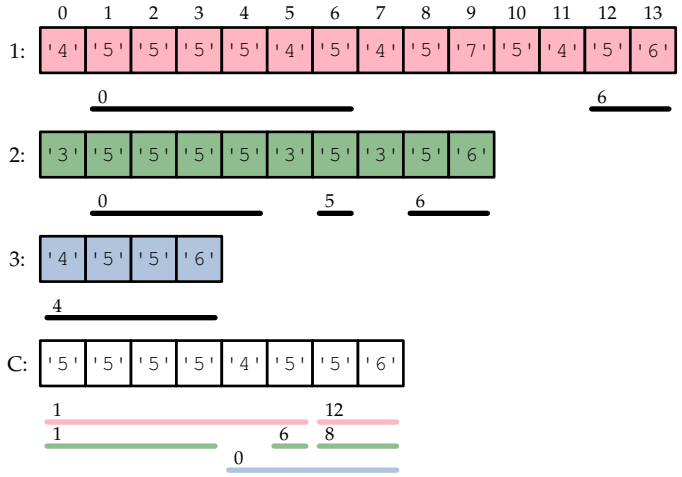


Figura 3.5: Interpretación del centroide usando *edit distance*

El centroide obtenido intenta conservar la mayor subsecuencia en común con cada una de las cadenas del conjunto inicial. El resultado es la combinación de dichas subsecuencias de la forma más compacta posible. Además, el orden relativo de los distintos componentes o subcadenas comunes se conserva en el centroide, generando una cadena en la que se mantienen el orden de los *tokens* que pasan al centroide.

Como ya se ha comentado, las distancias medias al centroide pueden usarse como una métrica si el conjunto de cadenas que se evalúa está compuesto exclusivamente por dos elementos[5].

Distancia de edición media en conjuntos de dos cadenas

3.5 Esta afirmación fue demostrada en la **seccion 3.1.1**

Es interesante detenerse un momento para destacar la relación entre una distancia de edición DE y la distancia de edición media al centroide (DEMC). Para ello, se van a generar pares de cadenas mínimas (c_1 y c_2) cuya diferencia se pueda expresar con, a lo sumo, una única operación de distancia de edición. Posteriormente, esas mismas cadenas son comparadas empleando la DEMC proporcionándose el coste de edición media al centroide y, como cálculo adicional, el centroide calculado y las operaciones de transformación[6]desde el centroide hasta cada cadena. La **tabla 3.3** muestra los resultados obtenidos.

3.6 «0» significa que no hay ninguna operación y «+» que hay una inserción en el centroide.

Operación DE	c_1	c_2	Coste DEMC	Centroide	Operaciones DEC
Ninguna	['A']	['A']	0	['A']	0 , 0
Inserción	[]	['A']	0,5	[]	0 , +
Eliminación	['A']	[]	0,5	[]	+ , 0
Sustitución	['A']	['B']	1	[]	+ , +

Tabla 3.3: Comparación de operaciones DE y DEMC

Aunque no se proporciona una demostración, la tabla muestra que cualquier operación de edición entre dos cadenas se puede realizar usando, a lo sumo, el operador de inserción al centroide. La DEMC sobre dos cadenas degenera a una distancia de edición con los pesos adaptados según se indica en la **tabla 3.4**.

Operación	Coste
Inserción	0,5
Eliminación	0,5
Sustitución	1,0

Tabla 3.4: Costes DE equivalentes a DEMC de dos elementos

En el caso de la DEMC entre dos cadenas, permite una forma alternativa de cálculo del centroide a partir de la lista de operaciones efectuadas en los elementos de una de las cadenas. Según se muestra en el **algoritmo 3.1**, el algoritmo indica que todo carácter de una cadena que no sea eliminado o sustituido por otro para llegar a la segunda cadena pertenecerá al centroide.

```
operaciones ← Operaciones de transformación DE de c1 a c2
pos ← 0
for all o ← operaciones do
  if o = "Ninguna" then
    centroide ← centroide + c1[pos]
    pos ← pos + 1
  else if o = "Eliminación" || o = "Sustitución" then
    pos ← pos + 1
  end if
end for
```

Algoritmo 3.1: Cálculo de centroide para DEMC de dos elementos

Numéricamente hablando, es posible acotar el valor de la DEMC respecto a la métrica Levenshtein estándar:

$$d_{DEMC}(c_1,c_2) \in [0{,}5;1] \cdot d_{Levenshtein}(c_1,c_2) \tag{3.15}$$

La distancia de edición media al centroide siempre tiene un valor comprendido entre la mitad de la distancia de edición y la distancia de edición de las mismas cadenas (**ecuación 3.15**). El factor concreto entre ambas métricas depende de las transformaciones aplicadas en la distancia Levenshtein, donde el caso mínimo se da en aquellas cadenas que solo tengan operaciones

de inserción y eliminación y el máximo cuando todas las operaciones de transformación entre las cadenas son de sustitución.

Distancia de edición media en conjuntos de tres cadenas

Con el fin de ilustrar el desempeño de la DEMC para más de dos cadenas se ha construido una tabla análoga a **tabla 3.3**; pero empleando tres cadenas. En este caso, la **tabla 3.5** muestra combinaciones de tres cadenas mínimas que representan distintos conjuntos de operaciones al centroide y sus distancias. Como se muestra, además de la operación de inserción, a partir de conjuntos de tres o más cadenas pueden aparecer también las operaciones de eliminación (−) y sustitución (×)[7].

Al considerar tres o más cadenas, no es posible comparar estas medidas de dispersión directamente con una única métrica DE estándar ya que no se proporciona un procedimiento de medida de distancia para más de dos elementos[8].

Además, el cálculo del centroide deja de ser una tarea trivial. A partir de tres cadenas, no se ha podido encontrar una relación que permita relacionar medidas simples de DE con la DEMC o algún algoritmo de cálculo directo del centroide.

3.7 Para facilitar la comprensión de la tabla se ha excluido el coste por aproximación en la operación de sustitución.

3.8 En (Powell, Allison, & Dix, 2000) se habla del alineamiento de 2 o 3 cadenas exclusivamente; pero presenta limitaciones tanto en número de cadenas empleables como las transformaciones disponibles.

c_1	c_2	c_3	Coste DEMC	Centroide	Operaciones DEC
['A']	['A']	['A']	0	['A']	0,0,0
[]	[]	['A']	0,333	[]	0,0,+
['A']	['A']	[]	0,5	['A']	0,0,−
[]	['A']	['B']	0,666	[]	0,+,+
['A']	['A']	['B']	0,7	['A']	0,0,×
[]	['A']	['A','A']	0,833	['A']	−,0,+
[]	['A']	['A','B']	0,833	['A']	−,0,+
['A']	['B']	['C']	1	[]	+,+,+
[]	['A']	['B','B']	1	[]	0,+,++
['A']	['B']	['A','B']	1	['A','B']	−,−,0
['A']	['A','B']	['B','B']	1,2	['A','B']	−,0,×
['A','A']	['A','B']	['B','B']	1,4	['A','B']	×,0,×

Tabla 3.5: Operaciones DEMC de tres cadenas

n-Gramas

Los conceptos de centroide y distancia media al centroide no requiere de una forma específica de métrica sobre la que basarse. Quizás la segunda familia de métricas de cadenas más usada tras

la distancia de edición es el conteo de n-gramas en las cadenas y el Coeficiente Dice[9] para estimar la similitud entre las mismas.

[3.9] Definido en la **página 23**.

Los n-gramas y la métrica basada en ellos presentan ciertas peculiaridades que se comentan a continuación, antes de hablar de la distancia en n-gramas media al centroide.

Un «n-grama» es una subcadena de n elementos.

$$N\text{-grama} :: (Símbolo,Símbolo,...,Símbolo)_n \quad \text{(TIPO 3.16)}$$

La función n-gramas(x) toma una cadena x y devuelve una lista de todos los n-gramas presentes en la cadena.

$$n\text{-gramas} :: [Símbolo] \rightarrow [N\text{-}grama] \quad \text{(TIPO 3.17)}$$

Dado que un determinado n-grama puede aparecer en más de una ocasión en una cadena, la lista de obtenida por n-gramas(x), no debe considerarse un conjunto de n-gramas sino más bien un multiconjunto o una bolsa [10] (Syropoulos, 2000) que asocia a cada elemento del mismo una cardinalidad o frecuencia de aparición.

[3.10] En inglés se suelen usar indistintamente los términos *multiset* y *bag*

Definimos la n-similitud como la razón:

$$n\text{-similitud}(x,y) = \frac{2 \cdot |n\text{-gramas}(x) \cap n\text{-gramas}(y)|}{|n\text{-gramas}(x)| + |n\text{-gramas}(y)|} \quad (3.18)$$

Que consiste en la razón normalizada entre los n-gramas comunes a ambas cadenas y el número total de n-gramas en las dos cadenas. Como los n-gramas son multiconjuntos, el cálculo de la cardinalidad y las intersecciones entre ellos han de realizarse siguiendo las reglas específicas para estas agrupaciones.

A partir de la medida de la n-similitud existen muchas formas de construir una medida de distancia por medio de n-gramas. Una de las más comunes es:

$$n\text{-distancia}(x,y) = 1 - n\text{-similitud}(x,y) \quad (3.19)$$

[3.11] A modo de contra-ejemplo, las cadenas a=`['a','b','a','c''a']` y b=`['a','c','a','b''a']` tienen una 2-distancia(a,b)=0 y sin embargo no son iguales.

La distancia por n-gramas no es, en verdad, una métrica. Por un lado no cumple con la **condición 2** (identidad de los indiscernibles) de las métricas: «dos cadenas con una distancia = 0 ocurre si, y solo si, las dos cadenas son iguales»[11]. Por otro lado, el rango normal de distancias en un espacio métrico es $[0,\infty)$; en el caso de la n-distancia su rango es $[0,1]$. Este rango reducido implica que el valor de la distancia por n-gramas satura por abajo (donde no puede distinguir entre cadenas muy parecidas) y también satura por arriba (en el que tampoco puede discriminar diferencias cuando estas superan un cierto umbral)[12].

[3.12] Los pares de cadenas («detener»,«remetes») y («mal»,«cocodrilo») tienen ambos una 3-distancia=1, a pesar de que parece razonable pensar que el primer par es más parecido entre sí que el segundo. Por eso, más que usar el término completamente distintos, decimos que no son comparables.

Tal y como se hizo cuando se habló del cálculo del centroide usando distancia de edición, vamos a tomar el mismo ejemplo con datos reales que se usó en la **figura 3.5** y calcular el centroide obtenido al emplear como métrica base la distancia 3-gramas.

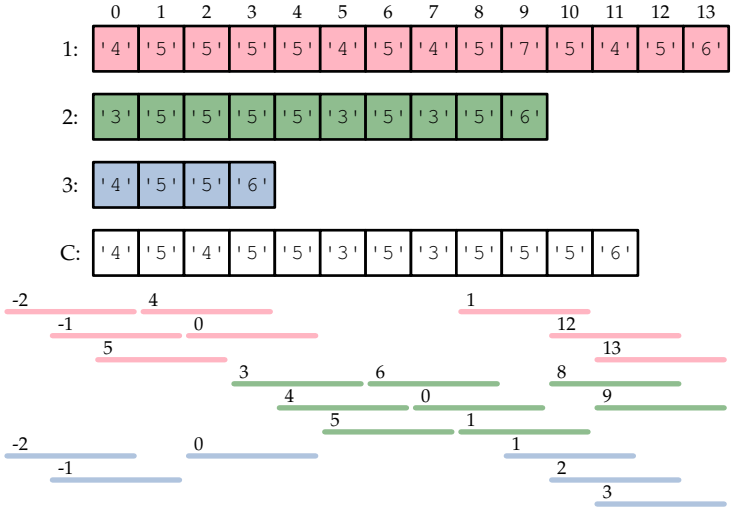


Figura 3.6: Interpretación del centroide usando la distancia de 3-gramas

El centroide generado (mostrado en la **figura 3.6** y usando el mismo convenio de líneas de color que en la **figura 3.5**) es una yuxtaposición óptima de distintos 3-gramas encontrados en las cadenas originales. Por la naturaleza de la distancia buscada, el algoritmo de búsqueda del centroide ha generado la cadena que incluya una mayor densidad de 3-gramas (mayor cantidad de ellos en la menor longitud posible) de los presentes en las cadenas 1, 2 y 3.

El resultado es una cadena que, efectivamente, tiene muchos *n*-gramas coincidentes con las cadenas; pero en un orden que hace difícil reconocerlos ya que el análisis por *n*-gramas no considera el orden de estos como un valor a conservar. Además, este resultado sufre de una *miopía* que no garantiza la conservación de estructuras de más de *n* elementos.

Si bien el centroide obtenido está obviamente relacionado con las cadenas, la validez de dicho centroide dependerá de las necesidades del problema sobre el que se aplique.

3.1.3 Cálculo del centroide

Hasta ahora se ha hablado de qué es un centroide y de qué dependen sus propiedades. Se han mostrado algunos centroides y

se ha hablado de la distancia media al centroide; pero no se ha indicado cómo se han calculado. Si bien el proceso de descartar una cadena como centroide es sencillo (basta con que exista una cadena con una distancia menor a conjunto de cadenas analizadas), el proceso de asegurar que una cadena es un centroide no lo es. Las operaciones de medida de distancia sufren de una asimetría en la que es fácil calcular la distancia entre dos cadenas; pero es muy difícil de encontrar una cadena que esté a una distancia dada de otra.

Dado que el tamaño del espacio de búsqueda de las cadenas aumenta exponencialmente con el número de símbolos disponibles en el alfabeto o la longitud máxima de la cadena a buscar, es inabordable hacer una búsqueda del centroide por fuerza bruta. Un alfabeto de 26 símbolos y cadenas de hasta 10 elementos[13] proporciona $\sum_{i=0}^{10} 26^i$ posibilidades (dando un valor superior a $1{,}4{\cdot}10^{14}$ posibilidades).

[3.13] No siendo estas condiciones muy exigentes.

Al no existir un algoritmo de cálculo de centroides es necesario emplear un algoritmo general de búsqueda en el espacio de cadenas. Sin descartar que en el futuro se prueben otras restricciones, la técnica de búsqueda deseada debe primar en la exactitud del cálculo del centroide frente a otras consideraciones como son el tiempo de cómputo o la memoria consumida (dentro de las capacidades de cálculo disponibles). Se han probado las siguientes estrategias de búsqueda:

— **Búsqueda voraz** Se establecen rondas sucesivas en la que solamente se prueban cadenas que añadan un símbolo (en cualquier parte) al mejor de la ronda anterior. El proceso se repite hasta que el mejor de una ronda tenga un desempeño peor que el mejor de la ronda anterior. En caso de que al final de una ronda haya dos o más candidatos con la misma puntuación, se escoge uno al azar.

  Este método de búsqueda es muy rápido y tiene un bajo consumo de memoria.

— **Búsqueda en anchura con podado** Se prueban todas las combinaciones posibles de un nivel y se podan las ramas cuya puntuación sea inferior a la máxima.

  En la práctica, esta estrategia es similar a la búsqueda voraz con la diferencia de que aquí no se conserva un candidato al final de cada ronda sino todos los candidatos de igual puntuación. El algoritmo es más lento en ejecución ya que se examinan más posibilidades que en el caso voraz y el consumo

de memoria está indeterminado ya que depende del número de candidatos que pasan de un nivel al siguiente[14].

[14] En la práctica, el ancho del árbol de búsqueda se ve limitado por la poda y este no crece indefinidamente.

— **Búsqueda por algoritmo genético** En este caso, a una población (que inicialmente se genera al azar) se le dota de mecanismos de evolución de individuos seleccionados por diversas condiciones. Un algoritmo genético bien calibrado debe de buscar la solución en un entorno de los mejores candidatos de la población, proveyendo de técnicas que eviten los mínimos locales.

  En la búsqueda empleada, se ha empleado una población de 1000 candidatos que ha ido evolucionando hasta que el mejor candidato se repite durante 100 generaciones consecutivas. La **tabla 3.6** muestra los parámetros de generación de la nueva población en una iteración. Indicando el método de evolución, que porcentaje de la nueva población se genera con este método y la distribución de probabilidad que determinan cómo se escogen los individuos a los que se le aplicará la operación indicada.

Operación	n	Extracción
Elite. Candidato copiado sin modificación	20%	Los n mejores
Mutación en contenido. Cambio de un símbolo por otro	30%	Distribución exponencial. Más probable cuanto mejor sea el candidato
Mutación en longitud. Aumento o disminución del tamaño de la cadena	20%	Distribución uniforme
Recruce. Se toman dos candidatos y uno sustituye una subcadena del otro	20%	Distribución uniforme
Azar. Generación al azar de nuevos candidatos	10%	

Tabla 3.6: Descripción de los operadores del algoritmo genético empleado

Para evaluar estos algoritmos de búsqueda se ha procedido a generar 200 conjuntos de cadenas (de entre 3 y 20 cadenas cada conjunto) y se ha buscado el centroide con los algoritmos de búsqueda descritos. La calidad de cada algoritmo (para el problema del cálculo del centroide) se ha determinado como el número de veces que cada buscador ha obtenido una cadena de distancia mínima al conjunto. Dado que las distancias de edición y de n-gramas no son comparables, primero se han evaluados los algoritmos para centroides que usan la distancia de edición y posteriormente para los que usan n-gramas.

Algoritmo de busqueda	Calidad *edit distance*	Calidad 3-gramas
Voraz	66%	5%
Anchura con podado	88%	6%
Algoritmo genético	100%	92%

Tabla 3.7: Idoneidad de los buscadores de centroides

Los resultados, mostrados en la **tabla 3.7**, reflejan cómo las búsquedas voraz y en anchura con podado caen en mínimos locales (especialmente en la búsqueda de centroides 3-gramas) con mayor frecuencia que la búsqueda por algoritmo genético que está diseñado para escapar de estos mínimos. A la vista de los resultados, se propone el uso del algoritmo genético presentado como método de obtención del centroide.

3.2 Ingeniería de descriptores sobre cadenas

Tal y como se ha constatado en el estudio del arte, las principales líneas de investigación de clasificación basada en descriptores se centran en los problemas de reducción dimensional (ya sea por medio de la extracción o la selección de características) y en los algoritmos de clasificación en sí. En general, se parte de un experto en el área de conocimiento del problema que posee los conocimientos necesarios para generar una lista inicial de descriptores.

Aunque existe alguna publicación, como (Scott & Matwin, 1999), que hace una enumeración de descriptores posibles aplicables en un área (documentos textos en este caso), el autor del presente trabajo no ha encontrado ningún listado de descriptores aplicables a cadenas (independientemente de la información codificada en ella).

Aportación de esta tesis a la ingeniería de descripciones

Es por ello que, para la realización de este trabajo, se ha efectuado una recopilación de descriptores aplicables a cadenas y se han categorizado en función de los requisitos que imponen a las cadenas sobre las que se aplican. Además, se presenta como aportación original algunas técnicas de construcción de variantes sobre los descriptores para aumentar las dimensiones del espacio de características.

El listado de descriptores[15] no puede ir acompañado de una interpretación de los mismos. Los descriptores son calculados sin conocimiento de la información representada en las cadenas y, por tanto, la interpretación de la información contenida en los descriptores ha de realizarse a posteriori para cada caso.

[3.15] Los nombres de los descriptores, identificados con un tipo de letra monoespaciado, no incluye tíldes ni ñ para que puedan ser usados directamente como nombres de variables en los sistemas de cálculo de los descriptores.

Un ejemplo de la necesidad de interpretación de un descriptor puede verse en la longitud de una cadena. Si decimos que una cadena tiene 3 elementos, puede significar el número de bytes que ocupa en memoria (en el caso de cadenas[16] en lenguaje C) o la duración de una captura de los datos (en el caso de una serie de datos muestreados).

Antes de pasar a la colección de descriptores, es necesario hablar sobre las limitaciones de la misma. La primera está relacionada con la trazabilidad de los descriptores seleccionados. Dado que las asunciones efectuadas sobre el contenido de las cadenas se han mantenido al mínimo, los descriptores mencionados son poco sofisticados y es difícil establecer una cronología de uso de los mismos. Ello nos ha obligado a renunciar a intentar atribuir el investigador que propuso su uso originalmente.

Desde el punto de vista de la utilidad de los descriptores, la utilidad de cada uno depende, en gran medida, del problema analizado y de la codificación empleada en la cadena[17]. La medida de la bondad de cada descriptor que permita determinar la selección o el descarte de cada descriptor es ya tarea de los procedimientos de selección de descriptores que se empleen.

[16] La longitud de la cadena no tiene por que coincidir con el número de caracteres de ella ya que una codificación como UTF8 puede usar dos o más bytes para un único caracter.

Limitaciones del listado

[17] Identificar el idioma en el que está escrito un texto puede hacerse analizando la frecuencia de aparición de símbolos o n-gramas. Si el texto ha sido comprimido previamente, el problema a resolver ha cambiado y probablemente los descriptores.

Clases de descriptores

Se han dividido los descriptores en tres categorías en función de los requerimientos exigibles a las cadenas con las que se trabajará:

— **Descriptores genéricos** que pueden aplicarse a cualquier cadena, independientemente de su contenido o campo de aplicación.

— **Descriptores diferenciales** que son aplicables cuando existe una relación que asocia a cada símbolo del alfabeto con un número. Esta asociación permite establecer un orden entre símbolos y otras operaciones matemáticas.

  Siempre es posible asociar un número a cada símbolo del alfabeto (nada más asociando un orden a este); pero no se garantiza que la utilidad de los descriptores sea la misma a aquellos casos en el que la relación símbolo-número sea natural.

— **Descriptores específicos** en el que la información descrita es específica del campo de conocimiento asociado a las cadenas y su codificación[18].

[18] En el campo de la evaluación psicológica de textos, la herramienta LIWC asocia a cada símbolo posible (una palabra en este caso) un valor en una serie de dimensiones como son: «alegría», «ira», «lejanía», …En este caso, los descriptores específicos podrían ser la suma total de la dimensión «alegría» o su valor medio en una frase o párrafo (Pennebaker, Boyd, Jordan, & Blackburn, 2015).

El asesoramiento de un experto en el campo, habilita la formulación de estos descriptores.

3.2.1 *Descriptores genéricos*

Los descriptores genéricos pueden aplicarse sobre cualquier tipo de cadena sin restricción sobre el contenido o el alfabeto[19].

[19] A modo de ilustración, se calcularán los descriptores de la cadena `['a','c','b','b','b','a','a','c','c']`

`longitud`

Longitud de la cadena. Número de elementos de la cadena.

`longitud`=9

`numSimbolosDistintos`

Número de símbolos distintos usados en la cadena.

`numSimbolosDistintos`=3

`simboloInicial`

t_0. Primer símbolo usado la cadena o de forma equivalente, el valor del primer token en la cadena.

`simboloInicial`='a'

`simboloFinal`

t_n. Último símbolo de la cadena o valor del último token de la cadena (donde $n =$ `longitud` $- 1$).

`simboloFinal`='c'

En algunos descriptores pueden aparecer distintos candidatos. Como mecanismo de resolución se escogerá, como norma general, aquel candidato más cercano al inicio de la cadena. Si el descriptor incorpora el sufijo «_rev», se escogerá al candidato más cercano al final de la cadena.

`moda` y `moda_rev`

El símbolo que más veces aparece.

El primero asigna mayor prioridad al inicio de la cadena y el segundo al final de esta.

`moda` = 'a',
`moda_rev` = 'c'

`frecuenciaModa`

Número de apariciones del símbolo moda.

`frecuenciaModa` = 3

`moda_`n y `moda_`n`_rev`

El n-ésimo símbolo que más veces aparece (`moda_1` es idéntico a `moda`). Si no se indica el valor de n, se presupone igual a 1.
Solo tiene sentido hablar del descriptor `moda_`n si el sistema de aprendizaje donde se use considera todos los descriptores de moda anteriores (de `moda_1` a `moda_`$(n-1)$).

`frecuenciaModa_n` y `frecuenciaModa_n_rev`

Frecuencia de aparición de la n-ésima moda.
Estas frecuencias también son dependientes de las frecuencias de orden inferior.

`modaConsecutiva_n` y `modaConsecutiva_n_rev`

Símbolo con mayor número de apariciones consecutivas. Este descriptor también presenta las mismas variaciones que el descriptor `moda`.

`modaConsecutiva='b'`

`frecuenciaModaConsecutiva_n`

Número de apariciones consecutivas de la moda.

`frecuencia_sym`

Porcentaje de aparición del símbolo *sym* en la cadena.

`frecuencia_a`=33%

`n-gram_m` y `n-gram_m_rev`

Es un descriptor análogo a la `moda_n`, pero aplicado a n-gramas. `n-gram_m` representa el m-ésimo n-grama (una subcadena de longitud n) más frecuente en la cadena.

`2-gram = 'ac'`,
`2-gram_rev= 'ac'`

`n-gram_Tm` y `n-gram_Tm_rev`

Este descriptor es una concatenación de los m n-gramas más frecuentes: `ngram_1`, `ngram_2`, `ngram_3`, ...`ngram_m`
El valor de m, el número de n-gramas que escoger, dependerá, entre otras cosas del número de símbolos disponibles y de la longitud de la cadena. En este trabajo, se ha escogido $m = 3$.

`2-gram_T='acbbcb'`

`markov_de_sym1_a_sym2`

Probabilidad de que el *sym*2 siga a *sym*1.
Para tener la matriz de markov completa, se incorporan dos símbolos extra (`'inicio'` y `'fin'`) que proporciona información del inicio y final de las cadenas.

`markov_de_a_a_c` = 66%
`markov_de_c_a_final` = 33%

3.2.2 Descriptores diferenciales

El segundo conjunto de descriptores es el que, sin hacer asunción sobre la información contenida en la cadena, si que impone una restricción a los símbolos miembros del alfabeto: debe existir una función de mapeo inyectiva:

$$aNúmero :: símbolo \rightarrow \mathbb{R} \qquad \text{(TIPO 3.20)}$$

que relaciona cada símbolo con un número real. Si existe esta función es posible construir descriptores aplicable a secuencias numéricas como pueden ser la obtención de diferencias entre símbolos, valores medios, mínimos o máximos, dispersión y otros. La función *aNúmero* facilita, además, la construcción de una relación de orden entre símbolos.

La conversión inversa (de número a símbolo) no es inmediata y, en ocasiones, no es factible. Si identificamos los descriptores diferenciales en función del tipo de información que aportan, podemos englobarlos en tres categorías: descriptores cuya información es no numérica, numérica pero no relacionada con los símbolos, y numérica y relacionada con los símbolos del alfabeto. De la primera categoría podemos mencionar descriptores que devuelven etiquetas, como son los descriptores de tendencia (`tendenciaInicial` y `tendenciaFinal`) o los que devuelven *n*-gramas (cuyas etiquetas están construidas por combinación de símbolos). Del segundo grupo, aquellos descriptores que devuelven cuentas (como `numMaximos`) o diferencias, distancias y otros (`ambito`, `sigma`, `pendiente`) y por último los que si devuelven valores relacionados directamente con símbolos (`maximo` o `media`).

Sólo en el último grupo de los mencionados tendría sentido aplicar una función de conversión inversa de valor a símbolo. Si bien para el procesado posterior de los descriptores, no tiene utilidad esta función de conversión inversa (y por tanto no es necesaria su implementación), si se le da un valor como herramienta de ayuda a la interpretación de los resultados. Formalmente, la propuesta de la función *aSímbolo* que implementar es:

$$aSímbolo :: Num\ a \Rightarrow a \rightarrow Either\ símbolo\ (símbolo,símbolo) \qquad \text{(TIPO 3.21)}$$

Donde ante un valor numérico *a*, la función devuelve o un símbolo (si el valor coincide con dicho símbolo) o una pareja de símbolos entre los que el valor suministrado reside.

Siguiendo con el ejemplo iniciado en la **nota 3.19**, mapearemos los símbolos de dicho ejemplo empleando la siguiente relación: $'a' \rightarrow 0$, $'b' \rightarrow 1$ y $'c' \rightarrow 2$ y aplicaremos, de forma transparente la función *aSímbolo* cuando sea aplicable.

`minimo`

Símbolo usado de menor valor.

`minimo='a'`

`maximo`

Símbolo usado de mayor valor.

`maximo='c'`

`ambito`

`maximo-minimo`. Distancia entre los símbolos máximo y mínimo.

`ambito`=2

`media`

Valor medio de los tokens.

`media`=1 (`'b'`)

`mediana`

La mediana de los valores de los tokens.

`mediana`=1 (`'b'`)

`sigma`

Desviación estándar de los valores de los símbolos de la cadena.

`sigma` = 0.82

`skewness`

Asimetría en la distribución de símbolos usados

`skewness` = 0

`curtosis`

Forma de las colas de la distribución de símbolos usados

`curtosis` = 1.5

Los descriptores genéricos *n*`-gram_T`*m* devuelven los *m* primeros *n*-gramas en el orden de máxima a mínima aparición. En ocasiones es más útil conocer los *m* primeros sin importar cuál de ellos es el más frecuente. Introducimos el sufijo «_o» para expresar la variación[20] en descriptores en los que el orden no es relevante.

n`-gram_T`*m*`_o` y *n*`-gram_T`*m*`_rev_o`

Variación sobre *n*`-gram_T`*m* y *n*`-gram_T`*m*`_rev` en el que el orden de los *n*-gramas más frecuentes no es relevante.

`tendenciaInicial`

Relación entre los valores de los tokens inicial (t_0) y el siguiente (t_1) en la cadena.
Los valores posibles son «descendente» (si $t_0 > t_1$), «horizontal» (si $t_0 = t_1$) o «ascendente» (si $t_0 < t_1$).

`tendenciaInicial`=horizontal

[20] Para calcular este descriptor, se calcula el descriptor básico sobre el que se basa y se reordenan los ítems siguiendo el orden establecido por el mapeo de símbolos. De esta forma la comparación de la etiqueta generada es más sencilla en las posteriores etapas de análisis.

`tendenciaFinal`

Relación entre el penúltimo token (t_{n-1}) y el último (t_n).
Los valores posibles son «descendente» (si $t_{n-1} > t_n$), «horizontal» (si $t_{n-1} = t_n$) o «ascendente» (si $t_{n-1} < t_n$).

`tendenciaFinal`=horizontal

`numMaximos`

Número de tokens cuyo valor es mayor que el de los tokens adyacentes.

`numMaximos` = 0

`numMinimos`

Número de tokens cuyo símbolo es menor que el de los tokens adyacentes.

`numMinimos` = 1

`numLlanos`

Número de tokens cuyo símbolo es igual a los símbolos de los tokens adyacentes.

`numLlanos` = 1

`salto`

$t_n - t_0$. Diferencia entre los valores del último y el primer token.

`salto` = 2

`pendiente`

`salto/longitud`. Relación entre el salto y la longitud de la cadena.
Esta pendiente no debe confundirse con la pendiente obtenida al calcular una regresión lineal.

`pendiente` = 0.22

`pendienteRegresion`

Pendiente obtenida por la regresión lineal de los valores de los tokens de la cadena.

`pendienteRegresion` = 0.08

`direccion`

Relación entre los símbolos de los tokens inicial (t_0) y el final de la cadena (t_n) en la cadena.
Los valores posibles son «descendente» (si $t_0 > t_n$), «horizontal» (si $t_0 = t_n$) o «ascendente» (si $t_0 < t_n$).

`direccion` = ascendente

3.2.3 *Descriptores específicos*

Los descriptores específicos son aquellos que extraen características de las cadenas que requieren de conocimiento sobre el contenido representado en las cadenas. Estos descriptores son específicos del problema que se está analizando y deben ser formulados por un experto en el área.

Los descriptores que se citan a continuación son los que han sido empleados durante el análisis de melodías flamencas que se efectúa en el **capítulo 4**.

`suavidadEuler`

El *Gradus Suavitatis* (nivel de agradabilidad) es una expresión matemática que indica cómo de agradable es escuchar un intervalo musical (0 representa el mayor confort y cuanto más crece el valor, más desagradable es el intervalo). Propuesto originalmente por (Euler, 1739). La **tabla 3.8** muestra los niveles de suavidad para los intervalos de hasta una octava.
Por extensión a un motivo, la `suavidadEuler` de una pieza es la suma de los *gradus suavitatis* de cada intervalo de la misma. Con objeto de que el descriptor sea independiente de la longitud de la melodía, el valor ha sido normalizado por el número de intervalos presentes en la cadena.

Intervalo	Semitonos	GS	B
unísono	0	0	0
	1	10	13,1
segunda	2	7	8,3
	3	7	10,1
tercera	4	6	8,4
cuarta	5	4	4,7
	6	13	16,7
quinta	7	3	3,7
	8	7	9,4
sexta	9	6	9,1
	10	8	9,3
séptima	11	9	12,1
octava	12	1	1

Tabla 3.8: Suavidad de Euler (GS) y Barlow (B)

`suavidadBarlow`

Otra expresión de la suavidad, más moderna, es propuesta en (Barlow, 2001). La **tabla 3.8** también muestra los valores de suavidad de Barlow correspondientes a la primera octava.
Igualmente al caso de `suavidadEuler`, el descriptor `suavidadBarlow` también está normalizado por el número de intervalos presentes.

`num_unisonos`

Número de intervalos con salto de 0 semitonos[21]. Normalizada por el número de intervalos.

`num_conjuntos`

Número de intervalos con salto de 1 o 2 semitonos[22]. Normalizados por el número de intervalos.

[3.21] O lo que es equivalente: el número de notas de igual valor a la justamente anterior

[3.22] Una de las características de la música flamenca es el predominio de intervalos conjuntos. Este descriptor pretende determinar si puede servir para clasificar dentro del flamenco.

num_disjuntos

Número de intervalos con un salto superior a 2 semitonos. Normalizados por el número de intervalos.

Si bien estos descriptores podrían haberse incorporado dentro del listado de descriptores diferenciales, presentan una semántica específica del ámbito de conocimiento considerado (la música). En este caso, un descriptor de salto entre tokens adyacentes que sea inferior a 2 no tiene, aparentemente, interés como descriptor de cadenas genéricas; pero un gran interés en la música. De ahí que se presente en una categoría distinta a aquellos.

3.2.4 Construcción de variaciones

Sobre los descriptores mencionados es posible construir variaciones que, si bien no modifican la naturaleza del descriptor, si que alteran algún aspecto de su cálculo derivando en valores distintos.

Prioridad para escoger símbolos

Como se ha comentado, las descripciones de algunos descriptores pueden dar lugar a situaciones en las que dos o más valores pudieran ser los valores de un descriptor. Quizás el ejemplo más típico de este tipo de descriptores es la `moda` en que dos o más símbolos pueden aparecer con la misma frecuencia. Por defecto se ha tomado el valor más cercano al inicio de la cadena como el escogido.

Se presenta como variación a estos descriptores la posibilidad de escoger el valor del descriptor dando la prioridad a los elementos más cercanos al final de la cadena.

Esta variación se ha denotado añadiendo el sufijo «_rev» al nombre del descriptor.

Irrelevancia del orden

A veces, un exceso de información es tiene efectos perjudiciales a la hora de analizar datos. Los descriptores de *ranking* usados *n*`-gram_T`*m* informan de los elementos con mayor frecuencia de aparición y en qué orden.

Esta variación se construye cuando la información deseada es cuáles son los *m* conjuntos más frecuentes sin importar cuál lo es

más. Si no importa el orden de los elementos, todas las permutaciones del los valores de un descriptor son equivalentes entre sí.

Trabajar con valores distintos entre si, pero equivalentes, requiere modificar la lógica de todos los algoritmos que usen dicho valor. Siendo esta modificación inabordable cuando se usan paquetes de software de análisis externos. Es por ello que se ha optado seleccionar un representante canónico que represente a todo el conjunto de permutaciones. Cuando se usa esta variación, se calcula el valor del descriptor y éste es sustituido por el representante canónico que es el almacenado.

El representante canónico escogido es la permutación que tiene sus elementos ordenados numéricamente (de menor a mayor). Para aplicar esta variación es necesario que esté definida la función *aNúmero* para transformar los símbolos en números que puedan, posteriormente, ser ordenados.

El sufijo «_o» se ha usado para denotar esta variación.

Diferencial de la cadena

Si está definido la función *aNúmero*, es posible efectuar una diferenciación discreta en la cadena antes de calcular los descriptores generando una nueva cadena sobre la que calcular nuevos descriptores. Si la cadena original tiene n símbolos (s_i), tras la derivación habrá $n-1$ valores calculados con la expresión:

$$d_i = aNúmero(s_{i+1}) - aNúmero(s_i) \tag{3.22}$$

A esta variación se le ha asignado el sufijo «_dif».

Descriptores de rango

Todos los descriptores descritos han sido calculado por características globales de la cadena (que incluyen a todos los elementos de la misma). A partir de estos descriptores es posible construir variaciones que se calculen empleando sólo porciones de la cadena original y así analizar características específicas de una región de la misma.

Para especificar la región a considerar, hay que emplear distancias independientes de la longitud. Por ello se propone que se usen posiciones situadas a un porcentaje (o tanto por unos) del inicio de la cadena. De esta forma, el descriptor `moda_[0%,50%]` buscaría el símbolo que más aparece en la primera mitad de la cadena.

Ciertos técnicas de análisis de datos requieren que todas las observaciones contempladas tengan todos sus atributos definidos. El empleo de un rango muy estrecho o la aplicación sobre cadenas cortas, puede derivar en descriptores de rango calculado sobre subcadenas vacías. En aquellos casos en los que no pueda permitirse una subcadena vacía se propone la siguiente regla:

– Todo extremo cerrado que en cuyo extremo no coincida con un símbolo, este se expandirá hasta que lo incluya.

Así, una cadena de tres elementos (que estarán situados en las posiciones 0%, 50%y 100%) si se le calcula el rango [10%, 30%), el extremo izquierdo se expandirá hasta que incluya al primer elemento.

De esta forma, todo rango que incluya un extremo cerrado, garantiza que al menos produce una subcadena con un elemento.

3.3 Construcción de un grafo

La construcción de un grafo dirigido requiere explicitar cómo se construyen los nodos y los arcos entre ellos. Los nodos serán subcadenas extraídas de las cadenas analizadas y los arcos enlazarán dos nodos si ambas subcadenas aparecen en una misma cadena (sin solapamiento entre ellas) con origen del arco la subcadena que aparezca primera y el destino la que aparezca segunda.

El problema fundamental que se plantea es la determinación de las subcadenas que servirán de nodos en el grafo. Para ello se emplearán técnicas de análisis estructural que destaquen aquellas subcadenas que se consideren más relevantes.

Dentro de las técnicas de análisis estructural de cadenas, la inducción de gramáticas es la más representativa. En la presente sección se comentarán los problemas existentes en los sistemas de inducción automática, y se expone una técnica de construcción de un catálogo de subcadenas relevantes y de colocaciones de subcadenas (denominadas «arcos»).

Problema de la inducción automática de gramáticas

La inducción de una gramática es un proceso doble de identificación de subcadenas (símbolos no terminales) y de las reglas de producción que manipulan símbolos. Los sistemas automáticos seleccionan ambos en función de unos heurísticos que buscan minimizar alguna función objetivo (un ejemplo de objetivo a minimizar es el número total de reglas de producción de la gramática (Rissanen, 1978)).

Tal y como se mostró en las **tablas 2.2**, un mismo corpus de cadenas puede ser descrito con diversas gramáticas generadas a partir de funciones objetivo distintas. En general, las reglas empleadas para la inducción de gramáticas son independientes del contenido y de la información contenida en las cadenas. Así, las subcadenas identificadas presentan un bajo contenido semántico. Aunque es posible establecer reglas de inducción que incorporen conocimiento sobre el corpus analizado[23], la complejidad de la definición de la función objetivo es similar a la inducción manual de una gramática por lo que no es práctico.

[3.23] Estos sistemas son denominados de inducción de gramáticas supervisado.

Así, un texto como: «Una generación cualquiera. Esa generación misma. La generación perdida.» genera una gramática (empleando el algoritmo Sequitur) que identifica como elementos interesantes: «a_generación_», «era», «._», «er», «a_».

Si bien la compresión del texto, en reglas y longitudes de estas, es importante, los elementos obtenidos no reflejan elementos significativos de la lengua española: no ha captado el concepto de palabra (o lo que es equivalente, la función separadora del espacio) ni la estructura repetida de artículo-nombre-adjetivo[24].

[3.24] Algunas de estas limitaciones pueden mitigarse empleando un corpus de mayor tamaño; pero, por construcción, este algoritmo nunca será capaz de establecer reglas para identificar conceptos como un artículo o un adjetivo.

Además del problema de la baja semántica de las cadenas, existe un problema adicional asociado con los niveles existentes en las cadenas que consideramos. Desde un punto de vista estructural, en este trabajo consideramos que las distinta subcadenas identificadas dentro de las cadenas provienen de dos niveles semánticos distintos: por una lado está la información proporcionada por el modelo original (que denominaremos información estructural) y por el otro la producida por las distintas transformaciones[25] que se han producido durante la transmisión de las cadenas (esta la denominaremos información accidental).

[3.25] Por citar algunas: incorporación de material extraño, deformaciones, extracciones de información y sustitución, etc.

Para nuestro problema, la construcción de una gramática ideal consistiría en un primer conjunto de reglas que, a partir del símbolo inicial, construyese los modelos y un segundo conjunto que se aplicara sobre los modelos para generar las alteraciones. Esta estructura de gramática en dos pasos ayudaría a entender tanto la información estructural presente en las cadenas, como los mecanismos de derivación y alteración de dichos modelos hasta las instancias reales. Lamentablemente, las heurísticas de inducción de gramáticas no tienen la capacidad de distinguir entre estos dos niveles, generando subcadenas que combinan la información estructural y la accidental. El desconocimiento de los modelos iniciales imposibilita la aplicación de este esquema directamente.

Establecer una gramática completa y útil (en la que exista una relación directa entre la información contenida en las cadenas y los símbolos no terminales identificados) sobre un corpus de cadenas es un proceso costoso que requiere de una cantidad elevada de cadenas (que no siempre está disponible) y del esfuerzo de un experto en el campo de trabajo . Estas condiciones no son casi nunca logrables, por lo que no existen muchas gramáticas que analicen información codificada en cadenas si esta presenta una mínima complejidad.

Reconocimiento de patrones como producto derivado de la inducción de gramáticas

Aunque no pueda ser construida una gramática completa que cada símbolo no terminal tenga alto contenido semántico, es posible seleccionar a posteriori aquellos símbolos que si sean más relevantes. Para ello es necesario, además de la gramática, de información asociada a cada cadena del corpus (como puede ser una etiqueta que identifique a la cadena en una categoría). Al cruzar la información entre símbolos y etiquetas, es posible seleccionar como relevantes, aquellos símbolos que sólo aparezcan en cadenas con la misma etiqueta. De esta forma, un sistema de inducción de gramáticas no supervisado se convierte en un sistema de reconocimiento de patrones (subcadenas que solo aparecen en un tipo de cadenas) supervisado.

Arcos

Si bien la obtención de subcadenas relevantes en el corpus es importante y puede ser usado como patrón de búsqueda en las cadenas, el empleo de los arcos del grafo proporciona estructuras más ricas de información. Un arco es equivalente al concepto de «colocación» empleado en lexicografía. Las colocaciones son agrupaciones de elementos (no necesariamente adyacentes) que presentan una frecuencia de uso significativa y cuyo significado puede ser la suma de sus constituyentes o uno específico para la colocación. Ejemplos de colocación son los *phrasal verbs* en la lengua inglesa o expresiones hechas como, por ejemplo, «mercado negro».

Un arco es una estructura de información construido sobre dos subcadenas relevantes que se centra en elementos estructurales descartando los accidentales. Podemos definir un arco como una relación entre elementos distantes de una cadena o, desde un punto de vista más intuitivo, como una subcadena que presenta un hueco en su interior[26]. La relación «Arco» se entiende como una colocación de dos subcadenas pertenecientes a la misma cadena que cumple las siguientes condiciones:

3.26 Otra posible interpretación es la de una subcadena de la que se sabe cómo empieza y cómo termina; pero que no tiene información de su desarrollo interior. Esta interpretación es menos preferible ya que transmite la idea de que la información importante es la que no se proporciona.

Condición de orden. El primer elemento del arco (origen) debe aparecer antes en la cadena que el segundo elemento (destino).

Condición no solapamiento Las subcadenas componentes no pueden solaparse.

Obviamente, la representación natural de los arcos es un grafo en el que las subcadenas son representadas por nodos y los arcos entre subcadenas como arcos dirigidos del grafo.

LOS ARCOS DE UN CORPUS FORMAN UN GRAFO

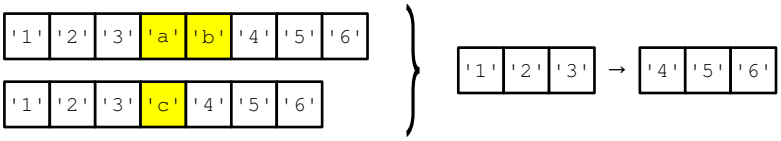


Figura 3.7: Arco de dos cadenas con elementos accidentales en la misma posición

La **figura 3.7** muestra el caso de arco más sencillo en el que dos cadenas comparten elementos estructurales y los accidentales (marcados en amarillo) se sitúan en la misma posición. Un arco siempre será una relación binaria entre dos elementos en el que la distancia entre los mismos no es relevante. Así, la **figura 3.8** que emplea una cadena sin estructuras accidentales puede usarse igualmente para definir arcos. En este caso, el hueco sería de tamaño 0.

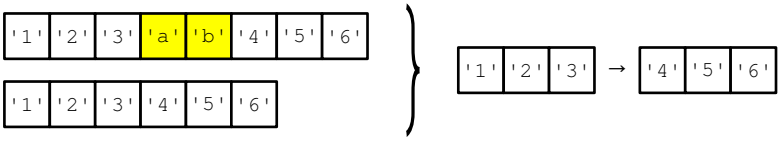


Figura 3.8: Arco con una cadena sin elementos accidentales

Las figuras **3.9** y **3.10** muestran las dos últimas situaciones básicas que pueden darse en la construcción de arcos de dos cadenas con, a lo sumo, un elemento accidental. En el primer caso, no hay forma de verificar si los símbolos `'3'` y `'4'` son estructurales o accidentales por lo que se dejan fuera en la construcción del arco.

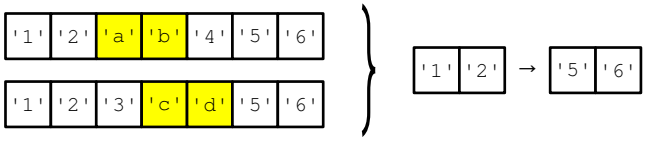


Figura 3.9: Arco a partir de cadenas con elementos accidentales en posiciones distintas con solapamiento

El caso en el que los elementos accidentales no solapan plantea el problema de la multiplicidad de arcos: con dos cadenas se pueden generar 3 arcos indicando distintas relaciones entre partes estructurales de las cadenas.

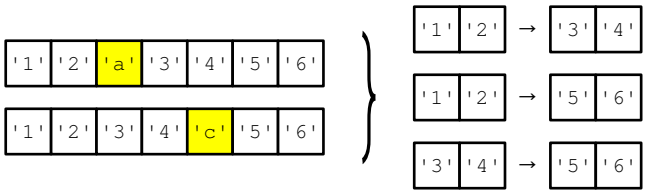


Figura 3.10: Arco a partir de cadenas con elementos accidentales sin solapamiento

Cardinalidad de los arcos

Es posible contar el número de posibles arcos que se pueden formar en una cadena de longitud N (la cardinalidad del conjunto de arcos generable). Para ello, vamos a describir un arco como una partición de la cadena en 5 elementos: el origen y el destino del arco y las zonas descartadas de la cadena antes del origen, entre origen y destino y tras el destino. Por lo tanto, un arco se puede describir como una tupla (h_1,o,h_2,d,h_3) de números enteros donde o y d representan el tamaño del origen y destino del arco y h_i los tamaños de los elementos rechazados de las cadenas. Contar el número posible de arcos consiste en contar el número de soluciones posibles de la **expresión 3.23**:

$$h_1 + o + h_2 + d + h_3 = N \tag{3.23}$$

Con las restricciones:

$$0 \leq h_i \leq N, \qquad 1 \leq o \leq N, \qquad 1 \leq d \leq N \tag{3.24}$$

La cardinalidad de los arcos (#arcos(N)) es [27]:

$$\begin{aligned} \#\,\text{arcos}(N) &= \sum_{o=1}^{(N-1)} \sum_{d=1}^{(N-o)} \sum_{h_1=0}^{(N-o-d)} \sum_{h_2=0}^{(N-o-d-h_1)} 1 \\ &= \frac{1}{24}(N-1)(N+2)(N^2+N) \\ &= \frac{1}{24}(N^4+2N^3-N^2-2N) \end{aligned} \tag{3.25}$$

que es del orden de N^4.

[27] Si consideramos que el tamaño mínimo de un motivo son 2 elementos, la expresión de la cardinalidad sería: $\#\text{arcos}(N) = \frac{1}{24}(N^4 - 6N^3 + 11N^2 - 6N)$

IMPOSIBILIDAD DE BÚSQUEDA DE ARCOS POR FUERZA BRUTA

Esta magnitud de arcos, por cadena, descartan el uso de la fuerza bruta a la hora de calcular todos los posibles arcos con información estructural. Es imperativo establecer un conjunto de

estrategias para determinar qué arcos se considerarán. Establecer estrategias demasiado restrictivas reducirá enormemente los datos con los que trabajar mientras que estrategias más laxas aumentarán la cantidad de información a procesar. Dado el carácter exploratorio en esta etapa de desarrollo de las herramientas, se ha escogido como mejor la primera opción (obtener menos datos que sean más fácilmente analizables) y tras una evaluación de los resultados obtenidos considerar si se amplía el espacio de trabajo.

3.3.1 *Técnicas de selección de arcos*

A continuación se enumeran distintas estrategias propuestas de generación y filtrado de arcos. Estas estrategias están enfocadas a atajar el espacio de arcos desde dos puntos de vista: la reducción del número de subcadenas sobre el que se construyen los arcos y el filtrado de los arcos una vez construidos:

Inducción de Subcadenas

La inducción de subcadenas es el primer mecanismo de reducción de subcadenas con los que trabajar. No es un filtro en si mismo sino, más bien, un mecanismo de selección de arcos de cierto interés. Las estrategias escogidas para esta tarea han sido tomadas de algoritmos de generación de gramáticas por métodos no supervisados[28]. Estas estrategias se basan en la búsqueda, a partir de ciertos heurísticos, de subcadenas repetidas. Hay que destacar que las técnicas mencionadas no buscan exhaustivamente todas las repeticiones que aparecen en las cadenas, por tanto, distintos algoritmos de búsqueda proporcionan listados de subcadenas distintos.

[3.28] Algoritmos que fueron descritos en la **sección 2.1.6**.

La localización de subcadenas interesantes se realiza usando técnicas de inducción automática de gramáticas tomadas de algoritmos de compresión sin pérdidas. El principio tras estos algoritmos es considerar lenguajes compuestos por una única cadena (el texto a comprimir) y construir una gramática cuyas reglas de producción ocupen menos espacio que la cadena original. El proceso de descompresión consiste en derivar del símbolo inicial la única cadena posible del lenguaje.

Dos de la técnicas empleadas (Sequitur y Lempel-Ziv-Welch) realizan la búsqueda con un enfoque voraz. Esto es, van calculando las subcadenas interesantes a la vez que van recibiendo los símbolos de la cadena original (haciéndolos rápidos de ejecución

y, en el caso de LZW, permitiendo que se inicie el envío de la cadena comprimida antes de haber terminado el proceso de compresión completo). La búsqueda voraz, aunque menos eficiente, imita mejor el proceso mental de las personas que, sin necesidad de tener una cadena completa puede generar impresiones parciales sobre el modelo asociado a la misma. La tercera técnica (Re-Pair) requiere tener toda la cadena en memoria antes de poder empezar a determinar las subcadenas.

Es importante destacar una discrepancia existente entre la finalidad original de estos algoritmos y de cómo van a ser usados para el filtrado de subcadenas. La finalidad original es tomar una cadena y construir una serie de reglas de producción que permita reconstruir la cadena original (y solo ésta) sin perdida de información. Dicho de otro modo, las gramáticas están asociadas a un lenguaje compuesto por una única cadena. Por otra parte, el uso de los algoritmos para la búsqueda de subcadenas importantes parte de lenguajes de más de una cadena y aquí no hay intención de una reconstrucción de la cadena original ya que, de hecho, no existe una cadena única que recomponer.

A tenor de estas características, se propone una modificación común a todos estos algoritmos para adaptarlos al uso con múltiples cadenas. La modificación consiste en la incorporación de un símbolo nuevo, llamado `'separador'`, que tendrá un tratamiento especial en el proceso de inducción de la gramática. Todas las cadenas se concatenarán en una sola empleando el `'separador'` como nexo de unión[29]. En la generación de la gramática, los algoritmos han sido modificados de forma que el símbolo `'separador'` no pueda participar en ninguna regla de producción. Esta variación impide que se puedan construir símbolos no terminales que incluyan el final de una cadena y el inicio de la siguiente.

A partir de todos los símbolos no terminales, es posible construir un diccionario de subcadenas significativas[30] que aparecen en todo el corpus.

Filtrado por solapamiento de subcadenas

El diccionario construido durante la inducción de gramáticas puede señalar subcadenas que solapen cuando son localizadas sobre una cadena. Se plantea un conjunto de estrategias de filtrado de subcadenas en función de si presentan solapamiento o no:

3.29 Lamentablemente, el orden de las cadenas al concatenarse es relevante en los algoritmos voraces por lo que las subcadenas identificadas pueden variar en función de la ordenación previa.

3.30 Por construcción, una subcadena convertida en símbolo no terminal es una que aparece en más de una ocasión.

Aceptar todos los posibles arcos y no hacer ningún filtrado en caso de solapamiento.

Primar la longitud frente a la posición Se establece un orden en las subcadenas localizadas en cada cadena (primero las de mayor tamaño y en caso de igualdad de tamaño, las que aparezcan antes). Y se van aceptando las subcadenas (por orden) siempre que no solapen con una previamente aceptada.

Primar la posición antes de la longitud De funcionamiento análogo al último criterio, en este caso se altera el orden de elección de subcadenas primando la distancia al inicio frente al tamaño de las mismas.

Primar una mayor cobertura Seleccionar aquellas subcadenas que cubran mejor la cadena sin solapar. Es decir, conseguir que el mayor porcentaje de símbolos de la cadena pertenezcan a una subcadena. Este criterio es equivalente a resolver el clásico problema de la mochila (Karp, 1972; Mathews, 1896) (problema que es NP).

La **figura 3.11** muestra un caso simple de solapamiento de subcadenas. Las subcadenas 1 y 3 sólo son escogidas cuando prima la posición frente al tamaño. Las subcadenas 2 y 3 son escogidas en caso de mayor relevancia de tamaño o cobertura y las tres subcadenas en caso de no filtrar.

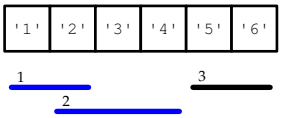


Figura 3.11: Casos de solapamiento entre cadenas

Filtrado por tamaño de subcadenas y arcos

Cuanto mayor sean las subcadenas, más específicas son, más información aportan y más identificables son. El filtrado por tamaño puede aplicarse tanto en la construcción del diccionario de subcadenas como sobre los arcos que se forman posteriormente.

Si denominamos l_o a la longitud de la subcadena de origen y l_d a la de destino de un arco. Se pueden establecer los límites de longitud mínima de subcadenas (m_s) y arcos (m_a) tal que todo arco y subcadena que satisfagan dichos límites se conserven. Para ello han de cumplir las relaciones:

$$m_s \leq l_o, \qquad m_s \leq l_d, \qquad m_a \leq l_o + l_d \tag{3.26}$$

La elección de estos límites permiten establecer distintas políticas de filtrado en función de los umbrales escogidos. Así, si $m_a \leq 2m_s$ la limitación de tamaño mínimo de arco no tendrá efecto y si $m_s \leq 1$ el que no tendrá efecto será el límite de tamaño mínimo de las subcadenas.

Filtrado por frecuencia de aparición

El número de apariciones de un arco en las cadenas analizadas está relacionado con lo asentado de una estructura de arco en las cadenas de un modelo. Cuanto mayor sea el límite mínimo de apariciones de un arco para considerarlo, arcos más relevantes se tendrán a costa de un menor número de arcos.

El umbral para seleccionar las frecuencias mínimas de aparición de los arcos depende del número de cadenas por modelo que se dispongan, así como de la variedad de estas.

Filtrado por categorías

Los arcos que aparezcan simultáneamente en distintas categorías pueden deberse a dos circunstancias indeseables: que hagan referencia a arcos que aporten tan poca información que puede aparecer en múltiples sitios[31]. La segunda circunstancia es que los arcos resalten características comunes inter-categorías. Estas características comunes a los modelos no ayudan a extraer las específicas de cada modelo; por lo que los arcos que aparezcan en más de una categoría son eliminados.

[3.31] La cadena «Los niños corren en casa» tiene un arco: «Los»→«en» que no aporta gran información sobre el contenido y que puede aparecer en múltiples contextos.

Es posible aprovecharse de las características de grafo de los arcos para filtrar algunos arcos. Para ello nos basaremos en las categorías en las que aparece cada nodo. Al ser un grafo dirigido, cada nodo puede aparecer en dos perfiles distintos: como origen de arcos y como destino. El filtrado de nodos (y consecuentemente de los arcos que se apoyan en ellos) se efectuará en función del número de categorías distintas en las que un nodo aparece en su función de su perfil como origen de arco o destino.

Para efectuar este filtrado, es necesario la construcción de un catálogo de nodos en el que se especifique las categorías de las cadenas en los que aparece y su función (si es un origen o es un destino).

A partir de dicho catálogo, se establecen distintos criterios de filtrado (resumidos en la **figura 3.12**) cada vez más restrictivos (en el que cada estrategia incluye los resultados de todas la que le suceden):

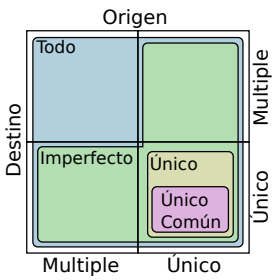


Figura 3.12: Filtrados de nodos según las categorías de cada perfil

Todo No se produce filtrado por las categorías contenidas en cada subcadena.

Único imperfecto Se requiere que cada nodo que permanezca tenga una única categoría como origen o destino.

Únicos Se requiere que los nodos tengan una única categoría tanto en su perfil de origen como el de destino. Estas categorías no tienen porqué coincidir.

Los grafos producidos tras aplicar este filtro se forman por nodos internos a una categoría o por nodos frontera en el que le llegan arcos de una categoría y salen de otra categoría.

Únicos y coincidentes La subcadena como origen y como destino solo aparece en cadenas de una misma categoría.

Este es el criterio más restrictivo de todos y da lugar a grafos inconexos en los que no hay contacto entre nodos de distintas categorías.

3.3.2 *Añadido para cadenas diferenciales*

En el cálculo de los arcos es posible establecer una variante que tenga en cuenta las características diferenciales de las cadenas[32]. En este caso, es posible añadir al catálogo de subcadenas relevantes un nuevo conjunto de subcadenas diferenciales que le añaden riqueza a los arcos generados.

[3.32] Una cadena diferencial es aquella en la que existe una función que mapeaba el alfabeto de las cadenas en un número entero

Una subcadena diferencial es una subcadena generada a partir de otra a la que se le han calculado las diferencias entre cada elemento de esta su contiguo (el **cuadro 3.1** muestra un ejemplo). Las subcadenas diferenciales permiten examinar evoluciones en la cadena sin importar el punto de inicio de dicha evolución.

```
['a','c','c','b']
['+2','+0','-1']
```

Cadena 3.1: Construcción de una subcadena diferencial

Para que las subcadenas diferenciales puedan usarse en los arcos es necesario establecer mecanismos que permitan su uso simultáneo a las subcadenas convencionales. Para ello, es necesario atribuirles las siguiente propiedades:

- Las posiciones asociadas a cada elemento de la subcadenas diferenciales son posiciones fraccionarias. Cada elemento de la cadena diferencial se situará entre los dos elementos de la cadena de la que se ha derivado. Es por ello que si un elemento diferencial está calculado como la diferencia entre los elementos en la posición p y $p+1$, el diferencial estará situado en $p + 0.5$.

  El tener definidas las posiciones permiten calcular si hay superposición entre cadenas y cadenas diferenciales.

- El tamaño de una subcadena diferencial (a efectos de hacer un filtrado de solapamiento) se mide en función de la cadena de la que deriva. Es decir, las dos cadenas mostradas en el **cuadro 3.1** se considerarán del mismo tamaño.

Estas dos propiedades permiten verificar las dos condiciones para la generación de un arco (la condición de **orden** y la de **solapamiento**), así como la aplicación de los distintos filtros de arcos enunciados.

A efectos prácticos, se puede trabajar con cadenas, sólo con cadenas diferenciales o combinando ambas. Esto último puede lograrse obteniendo las gramáticas por separado de subcadenas y subcadenas diferenciales (en cada cadena) y combinar las listas antes de obtener los arcos.

3.4 Conclusión

En este capítulo se han mostrado tres técnicas de adecuación que preparan las cadenas para abordar su análisis desde tres enfoques distintos. La primera de ellas es una metamétrica que emplea el concepto de centroide para establecer una medida de dispersión entre un grupo de cadenas.

A continuación se ha elaborado una lista genérica de descriptores y se han presentado mecanismos que permitan construir variantes sobre los descriptores calculados. Y, finalmente, se ha establecido una forma de conformar un conjunto de cadenas en un grafo dirigido.

Como se ha comentado previamente, estas técnicas de adecuación no son finales sino que abren la puerta al análisis y la extracción de información empleando los distintos algoritmos de aprendizaje automático disponibles en la literatura.

4 Aplicación al cante flamenco

4.1 Introducción

Las técnicas de adecuación de cadenas presentadas en el capítulo anterior organizan la información presente en las cadenas con objeto de poder aplicarlas en procedimientos de aprendizaje. La evaluación de estas técnicas ha de realizarse por medios indirectos ya que ninguna de estas etapas proporcionan resultados directamente evaluables[1].

[4.1] Ni las métricas, ni un listado de descritores o un grafo poseen una medida de calidad intrínseca.

El objetivo del presente capítulo es mostrar una serie, lógicamente reducida, de ejemplos de aplicación de análisis que usen datos procedentes de la adecuación de cadenas con información sobre cantes flamencos. El análisis y discusión de dichos ejemplos servirá para ilustrar la utilidad de los procedimientos mostrados en este trabajo. La obtención de resultados favorables en este campo de pruebas servirá al doble propósito de validar las herramientas propuestas y presentar resultados positivos en el campo estudiado.

Dada la naturaleza de demostrador de este capítulo, se han buscado técnicas de análisis de sencilla interpretación con el fin de que éstas no enmascaren la utilidad de los procedimientos propuestos. Obviamente, la selección escogida no debe ser interpretada como una descalificación de otras técnicas existentes con propósitos similares.

Aunque las posibilidades de análisis son innumerables, la exposición de este capítulo va a seguir la línea planteada por los **objetivos** enunciados en la introducción. Si bien ahora es posible plantear, además de las cuestiones, cómo se van a responder a estas preguntas y la aplicación al área de estudio planteada de cantes flamencos:

¿Qué cadenas derivan de un mismo modelo? La métrica desarrollada (DEMC) permite que se apliquen técnicas de agrupación que dividan el corpus de trabajo en conjuntos que procedan de un ancestro común.

Si se parte de la hipótesis de que los distintos estilos flamencos (denominados «palos») provienen de modelos distintos, es posible establecer un sistema de razonamiento basado en casos en los que se pueda clasificar una pieza en función de la cercanía a las piezas del corpus.

En el caso del flamenco, si existe una agrupación externa previa, es posible validar los resultados obtenidos.

¿Qué caracteriza a las cadenas que derivan de un mismo modelo? Los descriptores enunciados permiten caracterizar las distintas categorías identificadas. Además, la aplicación de los descriptores en rangos permite identificar qué zona de las cadenas es más relevante para dichas caracterizaciones.

Para ello se emplearán técnicas de clasificación en lo que, además de una clasificación correcta, interesa que la técnica informe sobre qué descriptores se basan para identificar una pieza en una categoría.

¿Qué partes de las cadenas son más probables que procedan del modelo? Ser capaz de identificar subcadenas estructurales permite diferenciar entre elementos que proceden del modelo original y las alteraciones aplicadas a lo largo del tiempo.

A diferencia de los descriptores, que extraen características específicas de la cadena (o una región de ella), las subcadenas estructurales son porciones directas de las cadenas consideradas relevantes en el corpus. La extracción de las subcadenas permite la construcción de un diccionario de subcadenas y establecer análisis sintácticos sobre las cadenas estudiadas.

Los arcos permiten establecer secuencias de mayor longitud de las meras subcadenas.

Corpus usado

Para verificar la utilidad de las técnicas propuestas, se ha trabajado con dos corpus independientes sobre los que se tiene un grado distinto de conocimiento:

- El conjunto de Tonás (Deblas y Martinetes) es un conjunto de cantes con características definidas que es bien conocido por los expertos. Este corpus ya ha sido analizado en estudios anteriores como (Mora, Gómez, Gómez, & Díaz-Báñez, 2016). Por lo que se empleará como grupo de control en el que se podrá comparar los resultados obtenidos en este trabajo con los previamente publicados. El objetivo principal de usar este grupo no es tanto la extracción de información, como la validación de las herramientas de análisis aquí presentadas.

– El segundo conjunto de cantes (los Fandangos de Huelva) es un grupo que ha sido estudiado desde distintos puntos de vista como son el cultural (Marqués Donaire, 2017), el geográfico (López, 2003) o incluso desde la identidad de género (Chuse, et al., 2015); pero del que no existe un consenso general de clasificación musical.

  Los resultados del análisis aplicado a este corpus serán, por tanto, originales de este trabajo. Aunque, en la medida de lo posible, se ha intentado contrastar dichos resultados con la información disponible.

Dado que, en general, los expertos en flamenco son capaces de identificar un estilo escuchando exclusivamente el primer tercio[2], el estudio se realizará en las mismas condiciones al entender que la información necesaria para la catalogación está presente en dicho primer tercio.

[4.2] El «tercio» es una medida de división de los cantes flamencos que, en contra de lo que el nombre puede indicar, no tiene porqué haber tres en un cante. Cuando pueda producirse una confusión entre esta definición de *tercio* y el concepto de una tercera parte, se pondrá en cursiva para recordar que su significado es el de división de una frase en el cante flamenco.

Lamentablemente, existen algunos estilos analizados cuyo primer tercio es idéntico al de otros y la discriminación entre ambos dependen del contenido de los siguientes tercios. Trabajar únicamente con el primer tercio hace imposible distinguir entre estos estilos. No obstante, no existe ningún impedimento para que en futuros estudios se consideren cadenas que incorporen más de un tercio.

Algunos de los procedimientos empleados generan gran cantidad de resultados de gran valor etnomusicológico; pero un análisis en detalle de dichos resultados no redundaría en una mejor validación de las herramientas propuestas (objetivo final de esta tesis). Es por ello que se ha preferido por hacer una evaluación cualitativa de los resultados obtenidos dejando para el **anexo C** un listado más completo de resultados específicos del cante flamenco.

Organización del resto del capítulo

Desde un punto de vista de exposición del material, el resto del capítulo está dividido en tres secciones: en la primera sección se exponen las técnicas generales de preprocesado y análisis usadas en la consecución de los objetivos mencionados. Las dos secciones siguientes presentan los resultados del análisis aplicado al corpus de tonás y al de los fandangos de Huelva respectivamente.

4.2 Procedimientos de análisis

4.2.1 Preparación de los datos

Generación de MIDI etiquetado

Cada corpus empleado ha sido recopilado de fuentes distintas y las piezas se presentan, en cada caso, en un formato específico. Es necesario efectuar un preprocesado que unifique el formato de todas las piezas usadas. Con esta homogeneización, se logra la posterior aplicación de los mismos procedimientos independientemente de la procedencia de los datos.

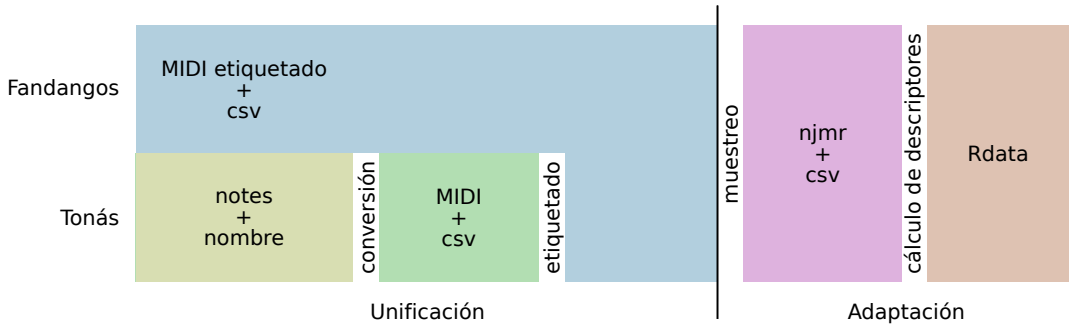


Figura 4.1: Preprocesado de las piezas

Se ha optado por elaborar una serie de pequeñas utilidades de conversión entre formatos en vez de desarrollar un tratamiento específico y monolítico para cada conjunto de piezas. Este tratamiento multi-etapa permite ajustar, cuando sea necesario, el formato de almacenamiento de la información sin necesidad de reescribir todas las herramientas de conversión. El proceso de conversión se ilustra en la **figura 4.1** (donde cada color representa un formato de almacenamiento de datos). Este proceso se ha construido en dos fases. Una primera fase consiste en la conversión de los ficheros de entrada a un formato común como es el formato MIDI y, a partir de este y en una segunda fase, la conversión a formatos útiles para el procesamiento.

Preprocesado de las tonás

De los dos conjuntos de piezas, las tonás es la que más transformaciones ha sufrido. Cada toná está registrada con ficheros en formato «notes»[3] que no es más que un fichero CSV (*Comma-Separated Value* (Shafranovich, 2005), formato muy extendido para almacenar tablas de datos) en el que cada línea representa una nota con este formato:

[4.3] Formato empleado en la fuente original de los datos: (Mora & López Gómez, 2010)

inicio (segundos), duración (segundos), nota MIDI, energía

El resto de la información asociada a cada pieza está codificada dentro del nombre de los archivos suministrados. Las tonás son convertidas a ficheros MIDI y la información de metadatos (donde se identifica la pieza y el estilo de la misma) es trascrita a un fichero de información en CSV.

Para terminar la primera etapa de conversión, los ficheros MIDI son etiquetados[4]. Un «MIDI etiquetado» es un fichero MIDI al que se le han añadido etiquetas de texto con un formato concreto [5] para indicar el inicio de cada tercio. Las etiquetas informan de los tercios proporcionando un punto de corte para la posterior extracción de los tercios deseados. Adicionalmente, un fichero adjunto codificado en CSV aporta los metadatos asociados al cante como son un identificador de la pieza, el palo o el estilo de fandango asociado y el interprete.

Los fandangos han sido transcrito directamente en formato MIDI al que se le han añadido posteriormente las etiquetas, empleando para ello el ya citado programa `jucetranscripcion`.

PREPROCESADO DE LOS FANDANGOS

Muestreo de las melodías

En este punto, todas las piezas de todos los conjuntos de cantes están en un formato único sin que hayan sufrido ninguna pérdida de información en la conversión (dada la versatilidad del formato MIDI). Lamentablemente, este formato no es el más idóneo para su tratamiento con los algoritmos presentados. De hecho, los datos no están ni siquiera representados como una cadena. En esta segunda fase, las piezas son codificadas como cadena y, a partir de esta representación, se calculan los descriptores usados en el análisis discriminante.

En el cante flamenco, el concepto del tiempo es difícil de manejar. El ritmo es una variable expresiva más, empleada por el intérprete que hace que dos interpretaciones de la misma pieza por el mismo cantaor puedan variar las duraciones de las notas significativamente. En los cantes denominados «a compás», aunque la interpretación está más constreñida a patrones rítmicos definidos, también se aprecian estas distorsiones de tiempo (Berlanga, 2014).

Es por ello que se ha optado por usar una codificación en las cadenas en el que el papel de la duración de las notas sea mínimo o nulo[6]. En las cadenas, la unidad básica de almacenamiento de la información en este trabajo, se sustituye la información temporal de las notas (la duración de las notas) por una relación de

4.4 El proceso de etiquetado se efectúa a mano con el programa `jucetranscripcion` comentado en el **anexo B**

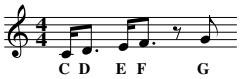
4.5 «@T»+número de tercio

4.6 Otros enfoques de análisis de melodías flamencas emplean técnicas como *Dynamic Time Warping* en el cual las duraciones se alteran para acomodarlas entre interpretaciones (Díaz-Báñez & Rizo, 2014).

orden (qué nota va a continuación de cuál) dando una estructura de datos para analizar más sencilla[7]. La conversión entre el tiempo y el orden se produce en el proceso de muestreo.

[7] La información perdida es de carácter temporal, jústamente aquella que queremos minimizar.

Se presentan dos estrategias distintas de muestreo que se ilustrarán a partir de las notas de la **partitura 4.1**:



Partitura 4.1: Melodía para mostrar técnicas de muestreo

Ignorar las duraciones de las notas (cadena 4.1) Cada nota proporciona exclusivamente un símbolo en la cadena independientemente de su duración.

```
['C','D','E','F','G']
```

Cadena 4.1: Muestreo una nota un símbolo

Repetir símbolos en función de un patrón temporal En este caso, la duración de las notas se refleja en el número de repeticiones que presenta un símbolo en la cadena. Para ello, se fija una duración patrón y las repeticiones del símbolo se definen en la **expresión 4.1**:

$$repeticiones = \left\lfloor \frac{duración}{patrón} \right\rfloor \tag{4.1}$$

Todas las notas de duración inferior al patrón escogido desaparecerán de la cadena. En el presente trabajo se ha escogido como patrón la nota de menor duración de cada pieza.

```
['D','F','G']
```

Cadena 4.2: Muestreo patrón=corchea

Las cadenas **4.2** y **4.3** muestran el proceso de muestreo de la partitura patrón. En la primera, las notas cortas de adorno han sido eliminadas en el muestreo por no llegar al patrón. En la segunda cadena, se ha escogido como patrón la nota más corta. El número de repetición recalca las notas con mayor duración en la pieza original.

```
['C','D','D','D','E'
'F','F','F','G','G']
```

Cadena 4.3: Muestreo patrón=semicorchea

Una estrategia posible (y descartada) es el proceso de muestreo fijo en el que se registra la nota que suena tras cada intervalo de tiempo.

El empleo de este muestreo corre el riesgo de coincidir con notas cortas (que se registrarían) despreciándose otras notas más largas adyacentes; pero de duración menor que el periodo del periodo de muestreo.

```
['C','E','silencio']
```

Cadena 4.4: Muestreo fijo = negra

La **cadena 4.4** muestra el efecto en el que las notas registradas son las más cortas y, posiblemente, menos importantes de las que aparecen en la melodía.

Formato «njmr»

Antes de codificar la pieza en una cadena, se efectúa además un transporte de las notas. Las notas MIDI se representan por un número entero entre 0 y 127 siendo las notas más habituales las

que están en el rango central (sobre el 60). El transporte suma (o resta) una cantidad fija a cada nota de la pieza manteniendo la melodía, aunque cambie la tonalidad de la pieza. Este transporte busca dos objetivos: unificar la tonalidad de todas las melodías haciéndolas independientes de la tonalidad empleada por el cantaor y facilitar la interpretación de las notas de las cadenas.

Para ello, el transporte se efectúa restando al valor de cada nota el valor de la última nota de la pieza (la tónica[8]). Quedando como valores negativos aquellas notas más graves que la tónica y la mayoría de las notas comprendidas en el rango entre 0 y 12.

Tras el muestreado y la transposición, todas las cadenas son registradas en un único fichero CSV[9] en el que cada pieza está representada por una línea.

La información presente en el fichero está codificada usando la descripción «njmr». En este formato, cada línea representa una pieza en el que en el primer campo es el título de la pieza y el resto son etiquetas de indicador de tercio o notas. Este formato, en contra de lo habitual, la longitud de cada línea depende de el número de notas muestreadas (y de tercios de la pieza). Un ejemplo de dicho formato se puede ver a continuación:

```
cante1;@T1;1;2;0;@T2;5;1;2
cante2;@T1;0;0;0;1;0;@T2;1;4;2;2;1;0
```

Los ficheros en formato njmr presentan, además, la ventaja añadida de los ficheros en texto plano: pueden ser inspeccionados fácilmente desde cualquier editor de texto para una rápida interpretación.

Cálculo de descriptores

Las herramientas de análisis discriminantes usadas están presentes en la literatura desde hace tiempo y ya están implementadas en distintas herramientas de software. Para la extracción de descriptores se ha optado por usar la plataforma de análisis estadístico R (R Core Team, 2017).

Dado el gran volumen de descriptores por pieza, el uso de ficheros CSV deja de ser práctico generando esta información en un formato más compacto y que facilite su manipulación en los siguientes procesados. Es por ello que los descriptores han sido calculados desde R leyendo los ficheros njmr, pero se han almacenado en el formato binario propio de la plataforma denominado «Rdata» para almacenamiento y acceso más fácil.

4.8 La nota tónica es la nota base sobre la que se definen las melodías y armonías de una pieza. Además, se usa como nota de reposo final en la música clásica, popular y en el folclore (Benward, 2014).

4.9 El formato CSV funciona como *lingua franca* de paso de información entre distintas aplicaciones ya que es un formato compacto, de fácil lectura y escritura y está soportado universalmente por los programas de procesado de datos

Para cada fragmento musical, se han definido 775 descriptores desglosados en las siguientes categorías (la **tabla 4.1** muestra la cantidad de descriptores en cada categoría):

Descriptores Genéricos	757
Sintesis	28
Markov	729
Descriptores Diferenciales	10
Descriptores Específicos	8

Tabla 4.1: Descriptores usados por tipo

- Descriptores genéricos (ver la **sección 3.2.1**) aplicables a cadenas con cualquier tipo de símbolos.
  - Descriptores de síntesis y
  - Descriptores de Markov. Cada elemento de la matriz de Markov ha sido definido como un descriptor independiente.
- Descriptores diferenciales (**seccion 3.2.2**) aplicables a cadenas cuyos componentes posean una operación de diferencia.
- Descriptores específicos (**seccion 3.2.3**) en los que los descriptores están fuertemente ligados al campo de trabajo.

Buscando que los descriptores sigan una estructura similar al razonamiento de los expertos, se ha decidido calcular todos los descriptores en regiones procesables por personas. De esta forma un descriptor es calculado para el *tercio* completo, en la primera o segunda mitad o en la parte inicial, central y final de la pieza (primer, segundo y tercer tercio del *tercio* analizado)[10]. Por tanto, el juego propuesto de descriptores es calculado 6 veces por pieza considerada, dando un total de 4650 descriptores por pieza.

[4.10] No se considera procedente dividir en más partes un *tercio* ya que se antoja difícil ser capaz de identificar «el segundo cuarto» de un fragmento de audio.

4.2.2 *Procedimientos de agrupación*

Clasificación jerárquica empleando k-mean

Tal y como se ha descrito, cuando se hablaba de las particiones en el **apartado 2.1.3**, el algoritmo *k*-mean presenta inconvenientes en dos aspectos importantes (Jain, 2010). El primer aspecto está relacionado con la información que proporciona el algoritmo: el sistema siempre devuelve *k* grupos (sea esta cantidad adecuada o no) para los datos que está procesando[11]. Además, la información proporcionada es limitada: para cada par de cadenas sólo establece si están en el mismo grupo o no.

[4.11] Por ese motivo se considera más un algoritmo de cuantización (o reducción de información) que un proceso automático de agrupación.

El segundo aspecto negativo de la partición está centrado en la dificultad para converger en la solución óptima. Ni se garantiza que haya convergencia, ni se garantiza que si converge la partición obtenida sea óptima.

Existen distintos métodos de construcción de una clusterización jerárquica a partir de *k*-mean. Uno de ellos es H*k*-means (Tung-Shou, et al., 2005) consistente en realizar una partición en 2 grandes clusters, que luego son subdivididos sucesivamente en grupos de 2 elementos hasta que la distorsión total [12] de cada cluster sea inferior a una dada. Lamentablemente, este método hereda los problemas de *k*-means en tanto que si dos cadenas relacionadas entre si caen en clusters separados (debido a un mínimo local), estos ya aparecerán como elementos separados.

[12] La distorsión se define como la suma de las distancias de cada elemento del cluster al centroide. En nuestro caso, la distorsión sería igual a la distancia al centroide media por el número de elementos del cluster.

El esquema propuesto, en este trabajo, de construcción de una agrupación jerárquica a partir de las ideas de (Fred & Jain, 2002; 2005). La idea básica consiste en generar distintas particiones sobre los mismos datos (por ejemplo, cambiando el número de grupos, *k*, a generar o las posiciones de los centroides iniciales). Combinando las particiones obtenidas, es posible generar una matriz de co-ocurrencia necesaria para la construcción de la agrupación jerárquica.

Dentro de los posibles algoritmos de combinación de particiones[13], se ha escogido usar, por su simplicidad, el *Cluster-based Similarity Partitioning Algorithm* (CSPA) de (Strehl & Ghosh, 2002) que si bien no es muy eficiente ($O(n^2)$), para el número de elementos que estamos empleando (inferior a 100) no es un gran problema en ordenadores modernos.

[13] Revisados en (Hore, Hall, & Goldgof, 2009)

El algoritmo consiste en tomar una serie de particiones y determinar, para cada pareja de elementos analizados, la proporción de las particiones en las que esos elementos terminan en la misma categoría. De esta forma, el CSPA transforma la asignación de etiquetas (en cada partición) en una distancia probabilística con la que ya se puede construir una clusterización jerárquica.

Clasificación jerárquica directa

Una segunda forma de construir la matriz de confusión entre piezas es realizar una estimación directa de la distancia media al centroide entre cada par de piezas estudiadas. La **sección** demostró que una DEMC entre dos cadenas se comportaba de forma equivalente a una distancia de edición con pesos específicos.

El análisis de evaluación de la jerarquía directa es idéntico al realizado en el caso de la jerarquía construida empleando DEMC + CSPA. Siendo necesaria destacar una diferencia: el rango de valores de las distancias. En el caso anteriormente visto, las distancias están referidas a la probabilidad de que las dos piezas

comparadas estén situadas en dos categorías distintas. Esta escala satura en sus dos extremos: una distancia de 0% no significa que las dos cadenas comparadas sean iguales, sino que son tan parecidas entre sí que siempre aparecen en el mismo grupo. Sobrepasado un cierto umbral de similitud, la probabilidad no tiene capacidad de discriminar la diferencias. Igualmente pasa en el otro extremo de la escala: en el momento que son suficientemente diferentes, presentarán una distancia de 100% siendo la escala incapaz de identificar cuanto más o menos lo son.

Por otro lado, la distancia directa DEMC no satura en ningún extremo. Una distancia de 0 significa que las dos piezas comparadas son idénticas (solo se establece para este caso) y no existe límite superior al comparar las piezas. La ausencia de límite en las medidas dificulta la comparación de las matrices de co-ocurrencia entre el método directo y el probabilístico ya que, al no estar acotado el primero, no se puede asignar una paleta de colores completamente equivalente entre ambas.

4.2.3 Caracterización de categorías

Características específicas de un grupo

El primer procedimiento de caracterización de una categoría, pretende buscar qué valores de qué descriptores son específicos a un grupo definido de piezas dado. Hay que destacar que las características identificadas siempre serán respecto a los otros grupos comparados y no deben de tratarse como una propiedad inherente del grupo o de aplicación universal.

Para determinar los descriptores y valores discriminantes, se va a construir un clasificador binario por cada categoría (Pertenece a la categoría *X* / no pertenece) estableciendo dos requisitos para seleccionar el algoritmo de búsqueda:

- El modelo de clasificación obtenido ha de ser fácilmente interpretable[14]. De forma que el análisis del modelo obtenido permita identificar fácilmente qué descriptores y qué valores de éste son los propios de la categoría estudiada.
- El procedimiento debe permitir seleccionar el nivel de complejidad, en número de descriptores o en la relación entre estos, en la búsqueda de estas características singulares.

4.14 Así, un clasificador que requiera del uso simultáneo de 30 descriptores y una secuencia compleja de operaciones (como podría ser una red neuronal) queda fuera de este procedimiento.

Concretamente, el límite establecido es de buscar sistemas de selección que solo requieran de un descriptor para identificar la categoría, aunque esta restricción elimine combinaciones más inteligibles para las personas[15]. Lamentablemente el estudio de la complejidad relativa de las reglas escapa fuera de ámbito de este estudio y simplemente se establecerá como frontera dura el uso de un único descriptor.

[15] Una posible regla que se descartaría sería: "pieza que empiece en la nota Do y que tenga más de 10 notas". Ésta es, para una persona, más sencilla de aplicar que "la pendiente de la pieza es de 0.0125 semitonos por nota" que no se descartaría.

Ambos requisitos están orientados hacia una misma finalidad: ser capaces de construir conocimiento de los palos a partir de descriptores individuales que permitiría detectar las características más relevantes a considerar en el cante flamenco y, por ende, facilitar la transferencia del conocimiento adquirido a los expertos en flamenco.

La estructura en clasificadores binarios posee una característica que usaremos en el análisis de resultados: un descriptor que presenta ciertos valores para identificar un estilo, presenta, igualmente, valores comunes para el resto de los estilos considerados. Así, si la región A de la **figura 4.2** representa los valores del descriptor que identifica una categoría, los valores de dicho descriptor en la región ~A (aquellos que no pertenecen a la región A) servirán para identificar propiedades comunes al resto de categorías empleadas.

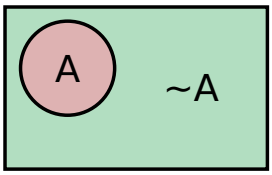


Figura 4.2: Propiedad dual de los valores discriminantes

Árbol de decisión

Dentro de los algoritmos de clasificación más comunes (Kotsiantis, Zaharakis, & Pintelas, 2007), los árboles de decisión son los que mejor cumplen con los requisitos recién expuestos. El modelo proporciona directamente las variables y los valores con capacidad de identificar las categorías y los árboles generados son fácil de podar para limitar la complejidad de las reglas. Para su cálculo, se ha utilizado la librería rpart (Therneau, Atkinson, & Ripley, 2015) de R.

Algoritmo de búsqueda de los descriptores

El **algoritmo 4.1** ilustra el procedimiento de búsqueda de los descriptores clasificadores de tipo que se ha empleado: Se entrena el mejor árbol posible y, si cumple con las condiciones de complejidad impuestas es almacenado y las variables empleadas en dicho árbol son eliminadas sucesivamente de la lista de descriptores (para que la misma combinación no pueda ser encontrada nuevamente). El proceso se repite hasta que no sea posible construir un árbol con las condiciones impuestas.

La conjunto de árboles aceptados, proporciona una doble información relevante para entender los datos. Por un lado informa de conjuntos descriptores que son capaces de explicar el una categoría; pero además, proporciona el cómo estos descriptores identifican cada categoría. Estos árboles permiten elaborar un

```
buscaDescriptores(d,c,cm,ld) =
Require: d {Datos}
Require: c {Categoría a entrenar}
Require: cm {Complejidad máxima}
Require: ld {Lista de descriptores}
listaArboles ←Set()
candidato ← entrenaArbol(d,c,ld)
if complejidad(candidato) ≤ cm then
  listaArboles.add(candidato)
  for descriptor ∈ candidato do
    listaArboles.add(buscaDescriptores(d,c,cm,ld−descriptor))
  end for
end if
return listaArboles
```

Algoritmo 4.1: Búsqueda descriptores clasificación de tipo

perfil que acote las características específicas de cada categoría existente del corpus.

La complejidad máxima elegida dependerá de la naturaleza de los datos examinados, número de categorías y de descriptores. Dicha complejidad deberá escogerse apropiadamente en función de cada caso.

Clasificadores globales

La búsqueda de clasificadores globales tiene como objetivo encontrar conjuntos mínimos de descriptores con capacidad de clasificar correctamente todas las categorías presentes. A diferencia de los descriptores clasificadores por tipo en los que se pretende encontrar las características que hacen de un grupo lo que es, estos descriptores están orientados a destacar las diferencias entre categorías.

Si bien las restricciones impuestas en los descriptores clasificadores por tipo siguen siendo útiles aquí (la interpretabilidad y la capacidad de escoger la complejidad máxima), la complejidad inherente del sistema ha aumentado ya que los clasificadores dejan de ser binarios para convertirse en clasificadores multi-categoría. La estrategia de identificación de descriptores, por tanto, ha variado con objeto de primar la simplicidad de los resultados.

```
buscaClasificador(d,n,ld) =
Require: d {Datos}
Require: n {Número de descriptores usados}
Require: ld {Lista de descriptores}
descriptoresGlobales ←Set()
n ← 0
while |descriptoresGlobales| = 0 do
  n ← n + 1
  for all candidato ← n-set(listaDescriptores) do
    acierto ← entrena(d,candidato)
    if acierto = 100% then
      descriptoresGlobales.add(candidato)
    end if
  end for
end while
return descriptoresGlobales
```

Algoritmo 4.2: Búsqueda descriptores clasificación global (búsqueda en un espacio de n dimensiones)

El **algoritmo 4.2** muestra el esquema seguido. Es un análisis por fuerza bruta en el que el espacio de los descriptores es reducido a subespacios definidos por n descriptores. Si el conjunto de descriptores es capaz de clasificar todos los casos, se considera un clasificador válido. Si no hubiera ningún clasificador válido de orden n, se repite el proceso aumentando el orden hasta que exista algún clasificador válido.

Independientemente de la técnica de entrenamiento que se escoja, este enfoque es muy ineficiente debido a la explosión combinatoria de n-sets que probar[16]. Este procedimiento de búsqueda impone, por tanto, límites prácticos a la hora de aplicarse que impiden usar esta búsqueda por fuerza bruta en conjuntos de datos que requieran 4 o más descriptores simultáneamente.

[16] El número de n-sets posibles de una lista de m descriptores es $C_{m,n}=\binom{m}{n}=\frac{m!}{n!(m-n)!}$. Para los descriptores usados, tomar 2 descriptores implican más de 10^7 ensayos y tomar 3 implica más de $16{\cdot}10^9$.

La clasificación Naive-Bayes (King, Feng, & Sutherland, 1995), presenta una serie de características que lo recomiendan para esta tarea:

Naive-Bayes

- Es de ejecución rápida (en entrenamiento y en clasificación).
- La interpretación de los resultados es fácil.
- Construye clasificadores multi-categoría.
- La clasificación puede aplicarse aun cuando existan discrepancias en los descriptores usados el clasificador entrenado y presentes en los datos.

El clasificador Naive-Bayes funciona sobre la intersección entre los descriptores usados en el entrenamiento y los descriptores disponibles durante la fase de ejecución.

Esta última característica permite, en cierto modo, optimizar la búsqueda de los clasificadores mínimos. Es posible efectuar un único entrenamiento del modelo Naive-Bayes empleando todos los descriptores disponibles y, dentro del bucle de búsqueda por fuerza bruta, solamente es necesario medir la eficacia del conjunto de descriptores ensayados. Dado que Naive-Bayes presupone la independencia entre los descriptores, el que un descriptor haya sido entrenado y, posteriormente, no usado no altera el resultado de un clasificador que no lo hubiera entrenado originariamente. Esta propiedad evita realizar un nuevo entrenamiento en cada ensayo de un clasificador.

Margen de Confianza

Además, Naive-Bayes proporciona una segunda ventaja: el clasificador funciona asignando probabilidades de que el elemento clasificado pertenezca a cada categoría presente. La categoría finalmente seleccionada es la que tenga la mayor probabilidad calculada. Dichas probabilidades (calculadas para cada categoría posible) pueden servir para establecer un margen de confianza dentro de los conjuntos de descriptores clasificadores. Si denotamos $m_{j,i}$ a la diferencia porcentual entre la categoría con mayor probabilidad y la siguiente, en una observación i empleando el n-set descriptor j, podemos definir el margen de confianza del set como:

$$M_j = \frac{1}{N} \sum_i m_{j,i} \tag{4.2}$$

la media de los margenes de todas las observaciones.

Implementación

La implementación de la búsqueda de los descriptores indicados en el **algoritmo 4.2** se ha efectuado en lenguaje R empleando el clasificador Naive-Bayes proporcionado por el paquete e1071 (Meyer, Dimitriadou, Hornik, Weingessel, & Leisch, 2017).

4.2.4 *Cálculo de arcos*

En la **sección 3.3** se discutió qué eran los arcos, cómo se derivaban a partir de los símbolos no terminales extraídos de gramáticas inducidas y alguna técnica de filtrado para seleccionar los más relevantes. En esta sección, se comentarán los aspectos prácticos que definen el procedimiento de construcción de arcos empleado.

La determinación de los motivos aptos para la construcción de arcos se efectúa conceptualmente en dos fases: la generación de un diccionario de motivos y la localización de estos en las piezas. Dado que el número de fragmentos identificado es usualmente elevado, tras la identificación de los motivos se efectúa un filtrado de los considerados menos interesantes.

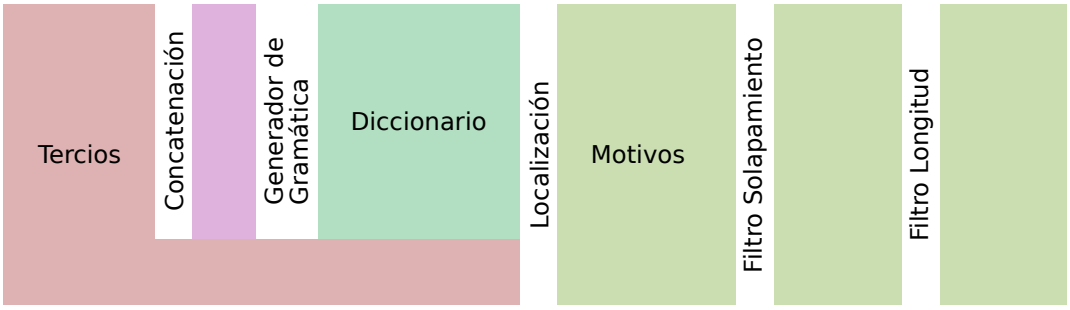


Figura 4.3: Proceso de generación de motivos

La **figura 4.3** ilustra el proceso completo. A las piezas se le extrae el primer tercio (con el que se va a trabajar) y estos son concatenados en una sola cadena. Para evitar el sistema identifique motivos a caballo entre dos tercios (que incluya el final de un tercio y, simultáneamente, el inicio del siguiente) la concatenación se efectúa empleando un símbolo único separador que no está permitido que aparezca dentro de ningún motivo localizado[17]. La existencia del separador de tercios evita que un motivo se construya saltando del final de un tercio al inicio del siguiente.

[4.17] Los algoritmos de inducción de gramáticas no distinguen entre los símbolos del alfabeto. La introducción de un símbolo separador especial que debe tratarse de forma distinta al resto de los símbolos, ha requerido adaptar todos los algoritmos de inducción de gramáticas usados.

La cadena que contiene a todos los tercios considerados es sobre la que se aplican los algoritmos de inducción de gramáticas de los que nos quedamos con las definiciones de los símbolos no terminales generados en un diccionario de motivos.

Una vez obtenido el diccionario, se buscan en cada tercio cada uno de los motivos del diccionario formando una lista de motivos instanciados que se reduce por medio de los filtros ya descritos.

A continuación se describen algunos aspectos relevantes en la construcción de las listas de motivos.

Instanciación de motivos al emplear catálogos

Tal y cómo se ha mencionado, los generadores de gramáticas que se han usado son Sequitur, Re-Pair y Lempel-Ziv-Welch. El mero proceso de la generación de la gramática ya identifica los símbolos no terminales (que acabarán siendo los motivos) y

su posición. Lamentablemente, el resultado difiere del aquí propuesto (generar un diccionario de motivos y posteriormente buscar su aparición).

Todas las gramáticas generadas por los algoritmos usados son tipo 2 de Chomsky (gramáticas libres de contexto). Estas gramáticas generan las cadenas por medio de una estructura piramidal en la que el símbolo inicial se divide en otros símbolos no terminales que a su vez se dividen en otros y así sucesivamente hasta que todos los símbolos son terminales (y, por tanto, no pueden volver a dividirse). En esta estructura, cualquier par de motivos identificados en la inducción o uno está contenido dentro del otro o no comparten ningún termino común.

En cambio, el empleo de un catálogo de subcadenas y su posterior búsqueda en las cadenas permiten que aparezcan situaciones en las que un motivo solapa parcialmente con otros.

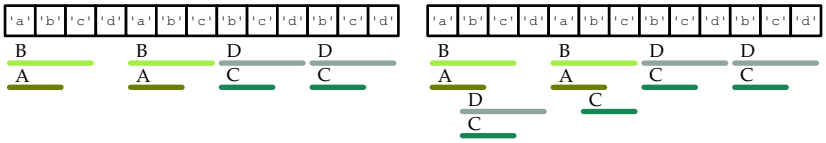


Figura 4.4: Diferencias entre gramática y uso de catálogo

La **figura 4.4** ilustra, con un ejemplo, las diferencias entre los dos enfoques empleados. La cadena mostrada ha generado una gramática empleando el algoritmo Sequitur. A la izquierda de la figura se muestran los motivos identificados (con líneas de color) y las posiciones identificadas para esos motivos durante la inducción de la gramática. Como se ha comentado, hay una estructura piramidal en la que el símbolo `<B>` contiene a `<A>` y a `'c'`; pero no incluye a ningún símbolo parcialmente.

En cambio, si tomamos los símbolos identificados en la gramática y los buscamos a lo largo de la cadena, vemos (en la derecha de la figura) que el símbolo `<B>` está compuesto por `<A>` y por `<C>` (con cierto solapamiento) y que existe una relación entre `<B>` y `<D>`.

Todos los símbolos generados por la gramática están presentes siempre en el catálogo y el empleo de éste siempre localiza las apariciones de símbolos no terminales que el generador de gramática detecta y, puede que, algunos más. Por ello, con el catálogo, el conjunto de motivos instanciados siempre será tan numeroso o más que al aplicar exclusivamente la gramática, justificando su empleo.

El número de símbolos no terminales determinados puede ser bastante elevado (como se mostrará en las figuras **4.20** y **4.21**). No siendo raro que un simple token o término de una cadena puede estar presente simultáneamente en decenas de motivos.

GESTIÓN DEL SOLAPAMIENTO DE MOTIVOS

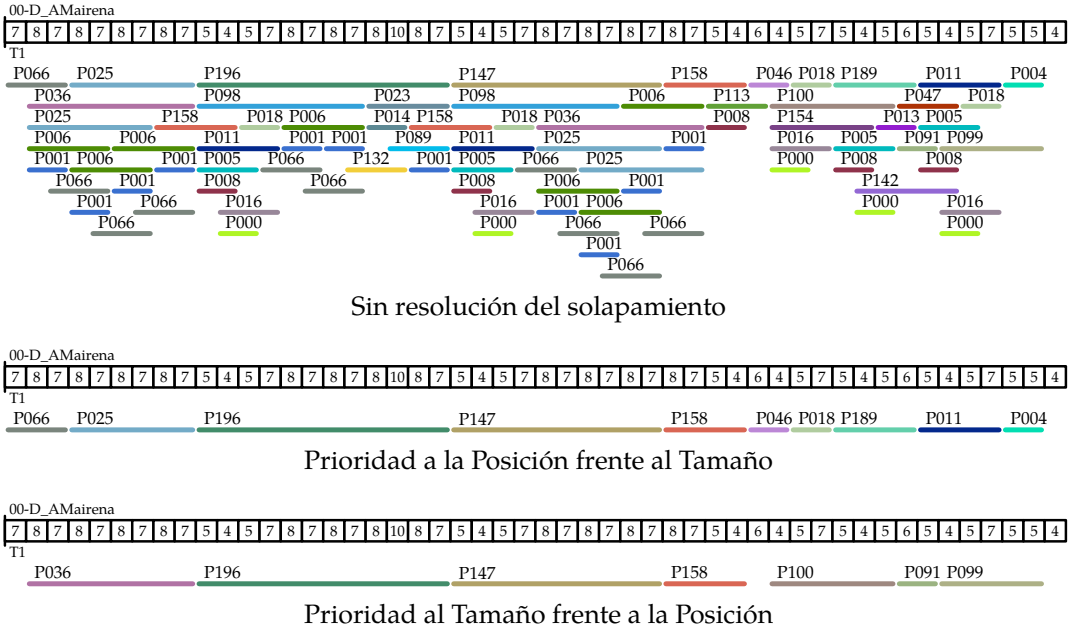


Figura 4.5: Comparación de gestión del solapamiento de motivos

La **figura 4.5** muestra cómo afectan las estrategias filtrado mencionadas, primando el tamaño o la posición de los motivos en una cadena. Aunque no hay garantía de que ninguna de estas estrategias consigan cubrir la cadena entera, el algoritmo de posición frente al tamaño tiende a cubrir todos los huecos (a costa de seleccionar motivos de menor tamaño) mejor que en el caso de tamaño frente a posición (que ni siguiera intenta cubrir los huecos que se van generando).

FILTRADO POR EL TAMAÑO DE SUBCADENAS

Otro sistema de restringir el número de subcadenas con el que trabajar es filtrar en función del tamaño de las subcadenas. La **figura 4.6** muestra las cadenas que se conservan si se impone un tamaño mínimo de 6 elementos. Obviamente, cuanto mayor sea el tamaño mínimo escogido, más identificables serán estos motivos (a costa de trabajar con un listado de motivos inferior).

Cálculo de arcos

Una vez obtenido el listado de motivos en cada tercio analizado, la construcción de los arcos no es más que el producto cartesiano

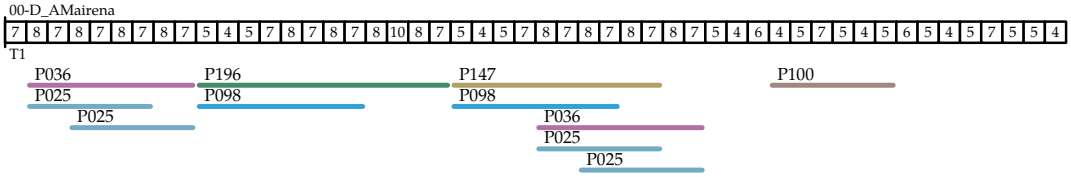


Figura 4.6: Efecto de un filtrado por tamaño de subcadenas (tamaño mínimo 6)

de los motivos en cada tercio (eliminando aquellos pares que no cumplen las condiciones de arco expuestas en la **sección**) y posteriormente el filtrado de los arcos considerado menos relevantes (con los filtros descritos en la **3.3.1**). El orden de aplicación de los filtros propuestos se muestra en la **figura 4.7**.

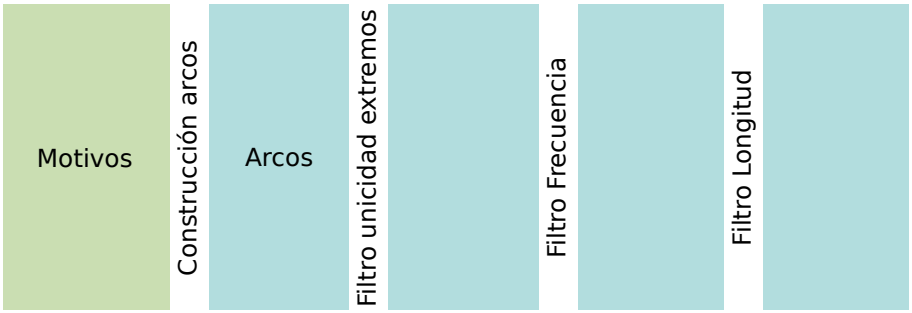


Figura 4.7: Proceso de generación de arcos

4.3 Tonás

4.3.1 Corpus

Las tonás son una familia de cantes que incluyen a las Deblas, los Martinetes (tipo 1) y los Martinetes (tipo 2). La selección de las tonás empleadas fue realizada por Joaquín Mora Roche (Mora, Gómez, Gómez, Escobar-Borrego, & Díaz-Bañez, 2010) y la transcripción por Cristina López Gómez(Gómez & Bonada, 2013; López-Gómez, 2013) estando disponible en la página web:

http://mtg.upf.edu/download/datasets/tonas.

La selección, que se muestra en la **tabla 4.2**, es una muestra representativa de cantes compuesta por 16 Deblas, 36 Martinetes 1 y 20 Martinetes 2. Esta transcripción está efectuada incluyendo adornos y otros motivos musicales no estructurales.

Este conjunto de piezas es bien conocido y no existen dudas sobre la clasificación de los distintos palos, por lo que este grupo de cantes se usará para validar las herramientas desarrolladas.

Id.	Tipo Toná	Cantaor
00	Debla	A. Mairena
02	Debla	Chano
03	Debla	Chocolate
04	Debla	J. Almadén
05	Debla	J. Heredia
06	Debla	M. Simón
07	Debla	M. Vargas
08	Debla	Naranjito
09	Debla	P. de Lucia
10	Debla	Talegon
11	Debla	T. Pabón
12	Debla	Turronero
14	Debla	A. Mairena
16	Debla	J. Merce
18	Debla	Diego Clavel
21	Debla	Gallina
24	Martinete 1	A. Mairena
25	Martinete 1	Chano
26	Martinete 1	Chocolate
27	Martinete 1	J. Almaden
28	Martinete 1	J. Heredia
29	Martinete 1	M. Simón
30	Martinete 1	M. Vargas
31	Martinete 1	Naranjito
32	Martinete 1	P. de Lucia
33	Martinete 1	Talegon
34	Martinete 1	T. Pabón
35	Martinete 1	Turronero
36	Martinete 1	A. Mairena
37	Martinete 1	A. Agujetas
38	Martinete 1	A. Mairena
39	Martinete 1	Chaqueta
40	Martinete 1	Curro Mairena

Tabla 4.2: Estilos de tonas usadas y su clasificación

Id.	Tipo Toná	Cantaor
41	Martinete 1	Diego Clavel
42	Martinete 1	Diego Rubichi
43	Martinete 1	Chocolate
44	Martinete 1	Indio Gitano
45	Martinete 1	Negro del Puerto
46	Martinete 1	Enrique Morente
47	Martinete 1	Jose Mendez
48	Martinete 1	Juan Talega
49	Martinete 1	Juan Talega
50	Martinete 1	Juan Talega
51	Martinete 1	Juan Talega
52	Martinete 1	Manuel de Angustias
53	Martinete 1	Miguel Vargas
54	Martinete 1	Mijita Hijo
56	Martinete 1	Niño Gloria
57	Martinete 1	Paco el Lobo
58	Martinete 1	Pansequito
59	Martinete 1	Pedro Sanz
61	Martinete 1	Tío Mollino
63	Martinete 2	A. Mairena
64	Martinete 2	Barullo
65	Martinete 2	Chocolate
66	Martinete 2	D. Agujetas
67	Martinete 2	E. Morente
68	Martinete 2	Chaqueta
69	Martinete 2	Torta
70	Martinete 2	Zambo
71	Martinete 2	Enrique Soto Sordera
72	Martinete 2	M. Agujetas
73	Martinete 2	M. Mairena
74	Martinete 2	M. Poveda
75	Martinete 2	Manolillo el Herrao
76	Martinete 2	Maria Solea

Tabla 4.2: Estilos de tonas usadas y su clasificación

Id.	Tipo Toná	Cantaor
77	Martinete 2	Matrona
78	Martinete 2	Niño Barbate
79	Martinete 2	P. Sanz
80	Martinete 2	Rancapinos
81	Martinete 2	S. Donday
82	Martinete 2	Tía Anica la Piriñaca

Tabla 4.2: Estilos de tonas usadas y su clasificación

4.3.2 *Agrupación de tonás*

Con objeto de evaluar la idoneidad de la DEMC a la hora de efectuar un análisis de agrupación, vamos a reproducir la operativa mostrada en el artículo (Mora, Gómez, Gómez, & Díaz-Báñez, 2016) usando la misma el mismo conjunto de datos y empleando los mismos indicadores de calidad allí mostrados. De esta forma, los resultados que se obtengan pueden compararse directamente con los citados en el artículo.

La agrupación efectuada es de tipo jerárquico y ésta se obtiene a partir de una matriz de distancias entre cada pieza. Vamos a presentar dos formas independientes de construir dicha matriz de distancias: usando la probabilidad de que dos piezas entren en el mismo grupo k-mean y una estimación directa de distancia.

Agrupación usando CSPA + DEMC

En la **tabla 4.3** se muestra un resumen de los valores empleados para aplicar el algoritmo de particionado por similaridad (CSPA): se han realizado 3 series de clasificaciones empleando desde 2 hasta 29 clusters. Dando un total de 84 procesos de agrupación en total.

Número de piezas agrupadas	73
Valores de k ensayados	2 .. 29
Particiones DEMC realizadas por k	3
Total de particiones efectuadas	84

Tabla 4.3: Características del CSPA ensayado

Una visualización de la matriz de distancias entre las tonás puede observarse en la **figura 4.8**. En ésta, cada distancia medida se representa por un cuadrado de color en el que cuanto más oscuro sea, mayor similitud tendrán las piezas comparadas. La diagonal principal representa la distancia de cada pieza consigo misma por lo que la distancia será mínima y nos da una referencia visual del tope de similitud.

Valoración cualitativa de la matriz de distancias

La matriz nos permite sacar algunas conclusiones cualitativas del corpus analizado: A groso modo, existen cuatro bloques

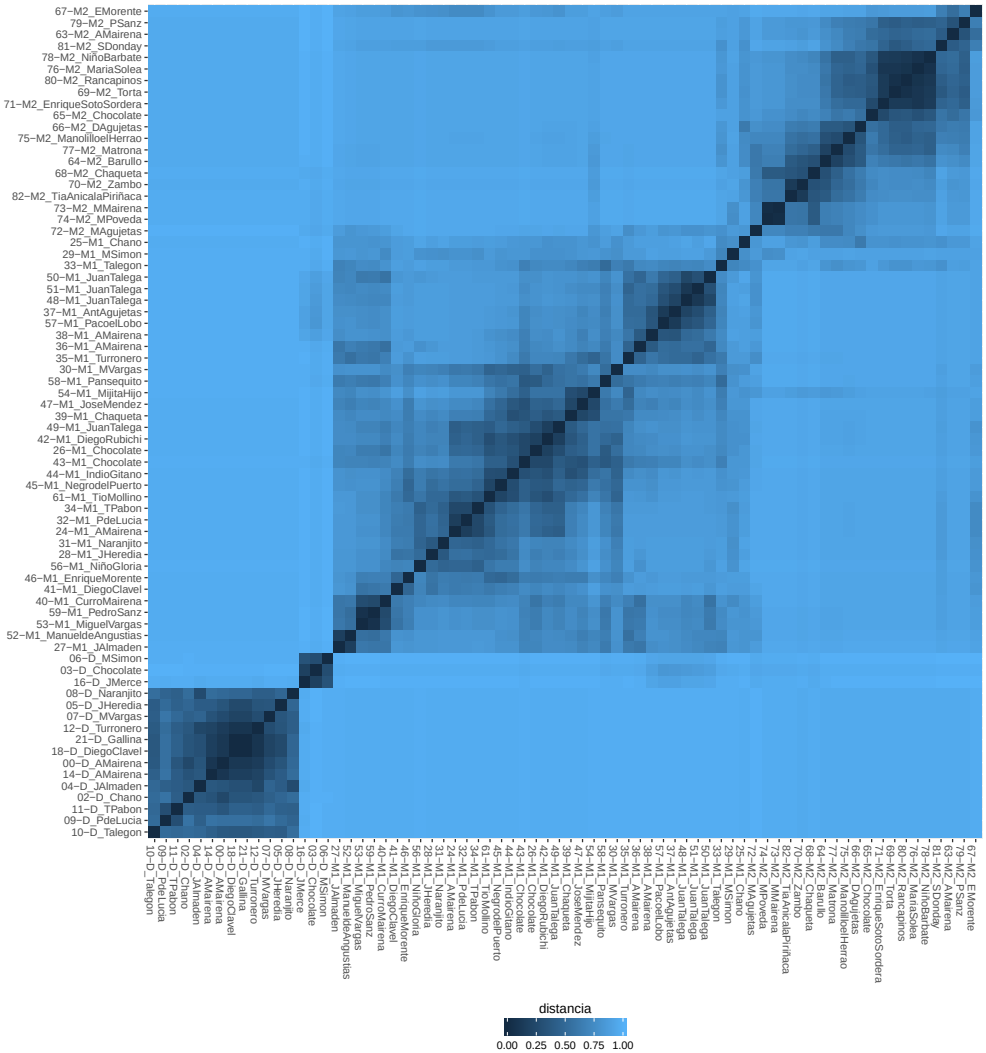


Figura 4.8: Matriz de distancias empleando DEMC+CSPA

sombreados bastante independientes entre si. Estos bloques coinciden con dos grupos de deblas y uno de martinetes 1 y otro de martinetes 2.

Los dos grupos de deblas están muy definidos. La mancha de cada uno es muy oscura (lo que implica gran similitud entre las piezas del grupo) y las interacciones de estos grupos con los demás es muy baja. El segundo grupo formado por 03-D_MSimon, 03-D_Chocolate y 16-D_JMerce se diferencia sustancialmente de las otras deblas.

Los grupos de martinetes tienen una manchas más claras que implican que existe una mayor variedad dentro de los grupos.

Aunque estos grupos están bastante definidos, existe un frontera más borrosa entre ellos definidos 25-M1_Chano, 29-M1_MSimon, 33-M1_Talegon y 72-M2_MAgujetas.

Aunque las relaciones intergrupos son claramente inferiores a las intragrupos, se aprecia que el grupo mejor conectado de los tres es el de los martinetes 2 presentan una cierta similitud con los martinetes 1 y a las deblas.

Cladograma

La matriz de distancias permite construir una agrupación jerárquica de las piezas analizadas en las que, no solamente, se etiquete cada pieza en una de las tres categorías consideradas, sino que se evalúen las relaciones entre piezas o grupos de piezas. Lamentablemente, una representación fiel de dichas piezas, sus agrupaciones y las distancias entre ellas requiere usar un espacio con un alto número de dimensiones [18] que hace imposible una representación exacta.

[18] Para mostrar fielmente n puntos, sería necesario un espacio de $n-1$ dimensiones.

Distintas técnicas se han desarrollado para efectuar la reducción dimensional (como las descritas en la sección de **extracción de características** en la **página 44**) y serían válidas en este contexto; pero, se ha preferido usar algoritmos específicos de visualización de estructuras jerárquicas en las que además de mostrar una representación espacial de las piezas, se incluye (en el mismo diagrama) información sobre grupos y distancias entre los mismos.

Los árboles de jerarquías (*split tree*, en ocasiones llamados cladogramas, árbol filogenético o dendograma en función de la forma del gráfico y la información representada) que se muestran a continuación han sido generados[19] con la ayuda del software SplitsTree 4 (Dress & Huson, 2004). La longitud de las ramas y el ángulo en las que estas se abren informan sobre las distancias entre cada clado formado. Además, para facilitar la interpretación de los mismos, se han coloreado manualmente las ramas para ayudar a identificar la categoría a la que pertenece cada nodo del árbol.

[19] En lo que, si no se dice nada en contra, se han empleado el algoritmo de representación de ángulos iguales sin raiz (Dress & Huson, 2004) y el algoritmo de distancias UPGMA (*Unweighted Pair Group Method with Arithmetic mean*) (Sokal, 1958).

La **figura 4.9** muestra el árbol formado a partir de la matriz de distancias ya expuesto. Nuevamente, es posible establecer una serie de observaciones cualitativas sobre la agrupación efectuada.

Salvo el caso de 67-M2_EMorente, los grupos de deblas y martinetes están agrupados por separado. Esta separación implica el éxito de la métrica desarrollada (que lo fue de forma independiente a este conjunto de datos).

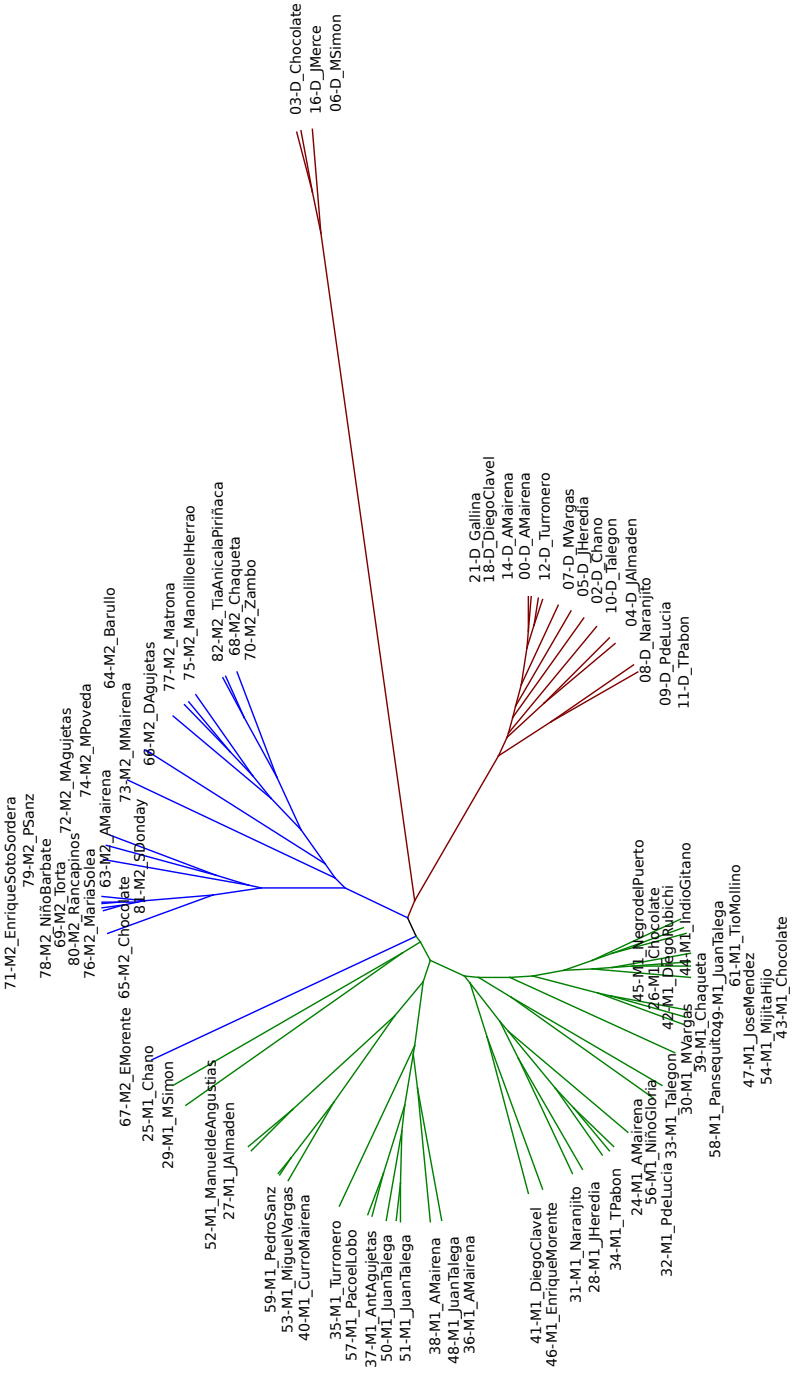


Figura 4.9: Árbol filogenético de distancias empleando DEMC+CSPA

Confirmando las observaciones efectuadas sobre la matriz de distancias, las deblas (formado por dos entidades bastante independientes entre si) es más compacto y diferenciado del resto de las tonás. Dentro de las deblas, el grupo de 03-D_Chocolate, 06-D_MSimon y 16-D_JMerce, aunque dentro de las deblas, forman un subconjunto diferenciado del resto.

Respecto a los martinetes, el ángulo subtendido de los martinetes 1 es más amplio que en los demás palos lo que da a entender que las variedades musicales de este palo son mayores y que tiene, por tanto, una forma musical más difusa.

La única pieza que parece fuera de sitio es 67-M2_EMorente que está situada en la frontera entre martinetes 1 y 2. En función de cómo se interpreten dichos límites, podríamos considerar que esta pieza está dentro de los martinetes 1 o de los martinetes 2 (en cuyo caso, las deblas sería una subrama dentro de los martinetes 2). Se ha escogido el punto de vista más conservador en el que ésta pieza se considera mal clasificada dejándola dentro del subárbol de los martinetes 1, contabilizándose como un error.

Evaluación cuantitativa

Con objeto de evaluar la idoneidad de la técnica de agrupación a partir de DEMC+CSPA, vamos a calcular el *f-score* de los palos a partir de la matriz de confusión de la agrupación (**tabla 4.4**). Para cada cluster puede determinarse la precisión (cuántos de los que han sido agrupados en el cluster x pertenecen realmente a dicho cluster) y la sensibilidad (cuántos de los que realmente pertenecen al cluster x han sido agrupados en dicho cluster). Y ha partir de estos el *f-score* que da una idea de lo acertado de la partición efectuada para un cluster. El *f-score* se construye como una media armónica escalada de la precisión y la sensibilidad según la **formula 4.3** (donde p es el valor de la precisión y s la sensibilidad).

		Agrupación		
		M1	M2	Debla
Real	M1	36	0	0
	M2	1	19	0
	Debla	0	0	16

Tabla 4.4: Matriz de confusión DEMC+CSPA

$$F_1 = 2\frac{p \cdot s}{p + s} \tag{4.3}$$

El factor de escala (2) garantiza que los posibles valores de F_1 están contenido entre 0 (fallo absoluto) y 1 (perfecta agrupación de los elementos).

A fin de calcular la precisión y la sensibilidad de cada cluster, se ha escogido la estrategia «uno contra el resto» en el que para cada grupo se verificarán las hipótesis: «Pertenece al grupo x» y «No pertenece al grupo x». Los valores obtenidos son:

M1	$Verdaderos\,Positivos = 36$	$Falsos\,Negativos = 0$
	$Falsos\,Positivos = 1$	$Verdaderos\,Negativos = 35$
	$Precisión = 0.973$	$Sensibilidad = 1.000$
	$F_1 = 0.986$	

Tabla 4.5: F_1 grupo Martinetes 1 usando DEMC+CSPA

M2	$Verdaderos\,Positivos = 19$	$Falsos\,Negativos = 1$
	$Falsos\,Positivos = 0$	$Verdaderos\,Negativos = 52$
	$Precisión = 1.000$	$Sensibilidad = 0.950$
	$F_1 = 0.974$	

Tabla 4.6: F_1 grupo Martinetes 2 usando DEMC+CSPA

Debla	$Verdaderos\,Positivos = 16$	$Falsos\,Negativos = 0$
	$Falsos\,Positivos = 0$	$Verdaderos\,Negativos = 56$
	$Precisión = 1.000$	$Sensibilidad = 1.000$
	$F_1 = 1.000$	

Tabla 4.7: F_1 grupo Deblas usando DEMC+CSPA

Habiendo evaluado cada agrupación por separado, se presenta a continuación los valores agregados de todos los grupos, usando *micro* y *macro averages*. El *micro average* consiste en la suma de los verdaderos positivos, falsos negativos, falsos positivos y verdaderos negativos de cada grupo haciendo un pseudo-grupo que los aúne a todos. Con estos agregados, se calcula nuevamente la precisión y la sensibilidad y se puede obtener el F_1 del agregado.

Micro average	$Verdaderos\,Positivos = 71$	$Falsos\,Negativos = 1$
	$Falsos\,Positivos = 1$	$Verdaderos\,Negativos = 143$
	$Precisión = 0.986$	$Sensibilidad = 0.986$
	$F_1 = 0.986$	

Tabla 4.8: F_1 agregado usando *micro average* de la agrupación DEMC+CSPA

La segunda forma de agregación empleada es el *macro average* en el que la precisión (sensibilidad) global es igual a la media de las precisiones (sensibilidades) de cada grupo. Con ellas se puede calcular, nuevamente, un nuevo F_1 agregado.

Macro average	*Precisión* $= 0.991$ *Sensibilidad* $= 0.983$ $F_1 = 0.987$

Tabla 4.9: F_1 agregado usando *macro average* de la agrupación DEMC+CSPA

Finalmente, un último parámetro de calidad de la agrupación que vamos a emplear es la exactitud (*accuracy*) definida como aciertos ($Verdaderos\ Positivos + Verdaderos\ Negativos$) frente al total de cadenas.

Grupo	M1	M2	Debla
Exactitud	0.986	0.986	1.000

Tabla 4.10: Exactitud agrupación DEMC+CSPA

En general, los resultados del indicador F_1 tanto en los clasificadores individuales como en los agrupados *micro* y *macro average*, así como el valor de la exactitud son excelentes siendo su valor muy cercano a el del clasificador perfecto.

Agrupación usando sólo DEMC

Como segundo enfoque, se ha construido la matriz de distancias directamente calculando la distancia de edición media al centroide para cada par de piezas. Este enfoque es mucho más eficiente en tiempo de cómputo ya que, como se demostró, la distancia entre dos piezas se degrada a una distancia de edición con pesos específicos.

En este caso, la matriz de distancias calculada ya no tiene los valores de la distancia comprendidos entre 0 y 1 sino que la distancia está en el mismo rango de las distancias de edición[20], por lo que los niveles de luminosidad mostrados en la matriz de distancias (**figura 4.10**) no son comparables con los de la **figura 4.8**[21].

Análogamente como se hizo con DEMC + CSPA, efectuamos una evaluación cualitativa de la matriz de distancias en la **figura 4.10**.

Como puede verse, los bloques de las deblas siguen existiendo aunque la división entre martinetes 1 y 2 es más difícil de apreciar. El grupo de las deblas se mantiene como un grupo compacto y mantiene su diferenciación con los martinetes 1 aunque se conserva la relación mayor (menor luminosidad, distancias menores) con los martinetes 2.

Empleando los mismos procedimientos que en la sección anterior, se ha construido un árbol filogenético a partir de la medida

[20] $d \in [0,\infty)$

[21] Con el fin de facilitar la interpretación, y sólo para la visualización de la matriz de distancias, se ha distorsionado la paleta de colores dando el 90% de la paleta a las distancias entre 0 y 25 y el 10% restante desde 25 hasta el final de la escala. El efecto conseguido es aclarar los tonos oscuros saturando en los claros.

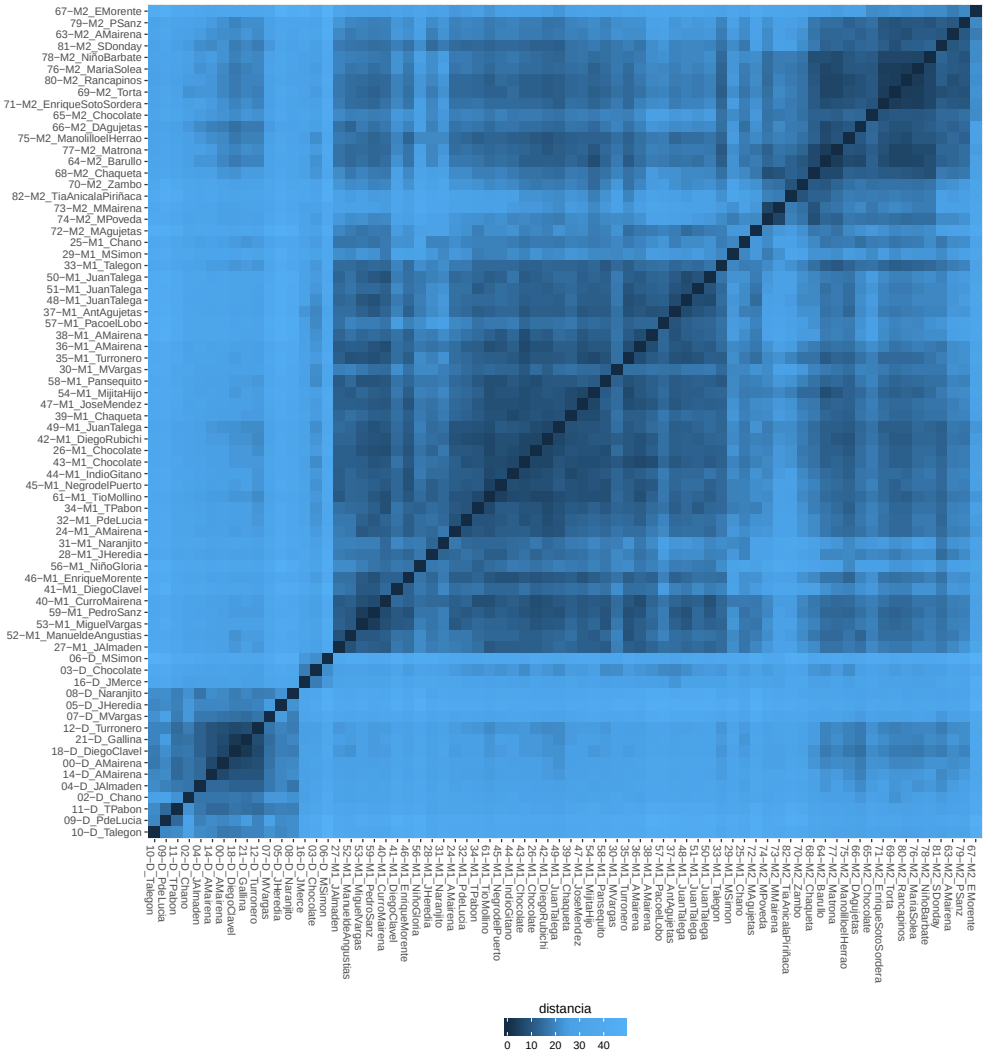


Figura 4.10: Matriz de distancias directas

directa de la distancia entre piezas (que se muestra en la **figura 4.11**). Como puede observarse, el grupo de las deblas sigue siendo un conjunto aislado del resto y bien definido; pero, en este caso, no existe un punto de inicio en el que partan tres ramas independientes por cada tipo de toná, sino que hay un origen del que parten los martinetes 1 y 2 y las deblas es una rama que sale desde dentro de los martinetes 2.

Además, hay 5 martinetes 1 (25-M1_Chano, 56-M1_NiñoGloria, 31-M1_Naranjito, 28-M1_JHeredia, y 29-M1_MSimón) que han aparecido dentro del conjunto de los martinetes 2.

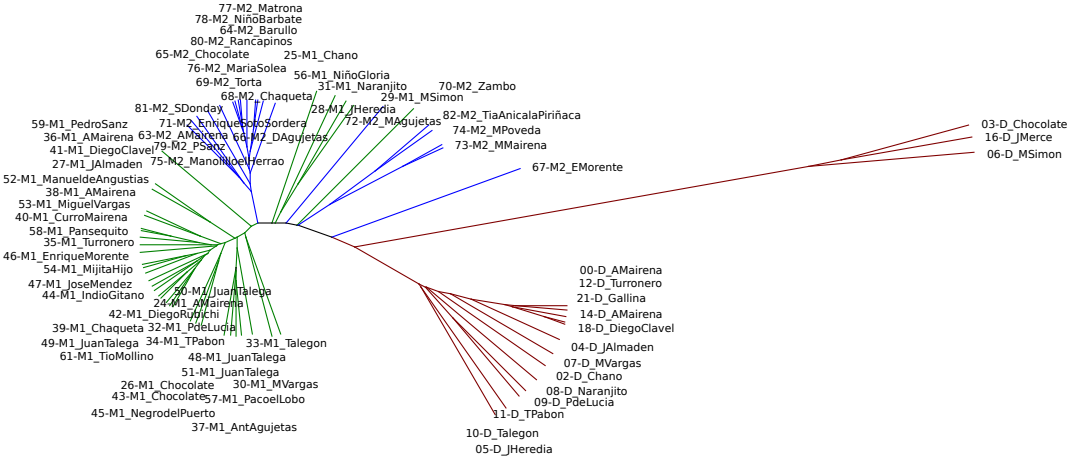


Figura 4.11: Árbol filogenético de distancias directas

Evaluación cuantitativa

La matriz de confusión para la aplicación de la métrica DEMC directa se muestra en la **tabla 4.11**. A partir de ahí podemos efectuar el cálculo de F_1 para cada cluster y agregados.

M1	*Verdaderos Positivos* $= 31$	*Falsos Negativos* $= 5$
	Falsos Positivos $= 0$	*Verdaderos Negativos* $= 36$
	Precisión $= 1.000$	*Sensibilidad* $= 0.861$
	$F_1 = 0.925$	

Tabla 4.12: F_1 grupo Martinetes 1 usando DEMC directo

M2	*Verdaderos Positivos* $= 20$	*Falsos Negativos* $= 0$
	Falsos Positivos $= 5$	*Verdaderos Negativos* $= 47$
	Precisión $= 0.800$	*Sensibilidad* $= 1.000$
	$F_1 = 0.889$	

Tabla 4.13: F_1 grupo Martinetes 2 usando DEMC directo

Deblas	*Verdaderos Positivos* $= 16$	*Falsos Negativos* $= 0$
	Falsos Positivos $= 0$	*Verdaderos Negativos* $= 56$
	Precisión $= 1.000$	*Sensibilidad* $= 1.000$
	$F_1 = 1.000$	

Tabla 4.14: F_1 grupo Deblas usando DEMC directo

		Agrupación		
		M1	M2	Debla
Real	M1	31	5	0
	M2	0	20	0
	Debla	0	0	16

Tabla 4.11: Matriz de confusión DEMC directa

Micro average	$Verdaderos\ Positivos = 67$	$Falsos\ Negativos = 5$
	$Falsos\ Positivos = 5$	$Verdaderos\ Negativos = 139$
	$Precisión = 0.931$	$Sensibilidad = 0.931$
	$F_1 = 0.931$	

Tabla 4.15: F_1 agregado usando *micro average* de la agrupación DEMC directa

Macro average	$Precisión = 0.933$	$Sensibilidad = 0{,}954$
	$F_1 = 0{,}943$	

Tabla 4.16: F_1 agregado usando *macro average* de la agrupación DEMC directa

Grupo	M1	M2	Debla
Exactitud	0.931	0.931	1.000

Tabla 4.17: Exactitud agrupación DEMC directa

Agrupación usando edit distance *directa*

A modo de comparación, se ha repetido el proceso de agrupación empleando la métrica *edit distance* estándar para poder obtener una medida de la mejora que presenta el método aquí propuesto sobre la referencia en métricas de cadenas. La matriz de distancias puede verse en la **figura 4.12** (nuevamente, el rango de la matriz de distancias es distinto a las dos anteriormente mostradas, por lo que no se pueden comparar los niveles de gris[22] entre unas y otras). Esta matriz ha sido calculada directamente aplicando la distancia de edición entre cada par de tonás.

Una vez obtenida la matriz de distancia, el procedimiento de generación del filograma en el que se pueden apreciar los bloques es idéntico a todos los casos presentados. El filograma obtenido se muestra en la **figura 4.13**.

Al igual que ocurría en el caso de la DEMC, las deblas surgen como una subrama de los martinetes 2; pero, además, en este caso, los martinetes 1 también surgen como derivadas de los martinetes 2.

Evaluación cuantitativa

La matriz de confusión para la aplicación de la métrica *edit distance* directa se muestra en la **tabla 4.18** y, repitiendo el proceso ya descrito, calculamos el indicador de calidad F_1.

[22] Si bien la distribución de manchas obtenida es similar a la obtenida empleando directamente la métrica DEMC (**figura 4.10**), las escalas son en ambas figuras distintas. La distancia máxima entre dos piezas en DEMC es del orden de 50, cuando si se emplea DE el máximo está en el orden de 70.

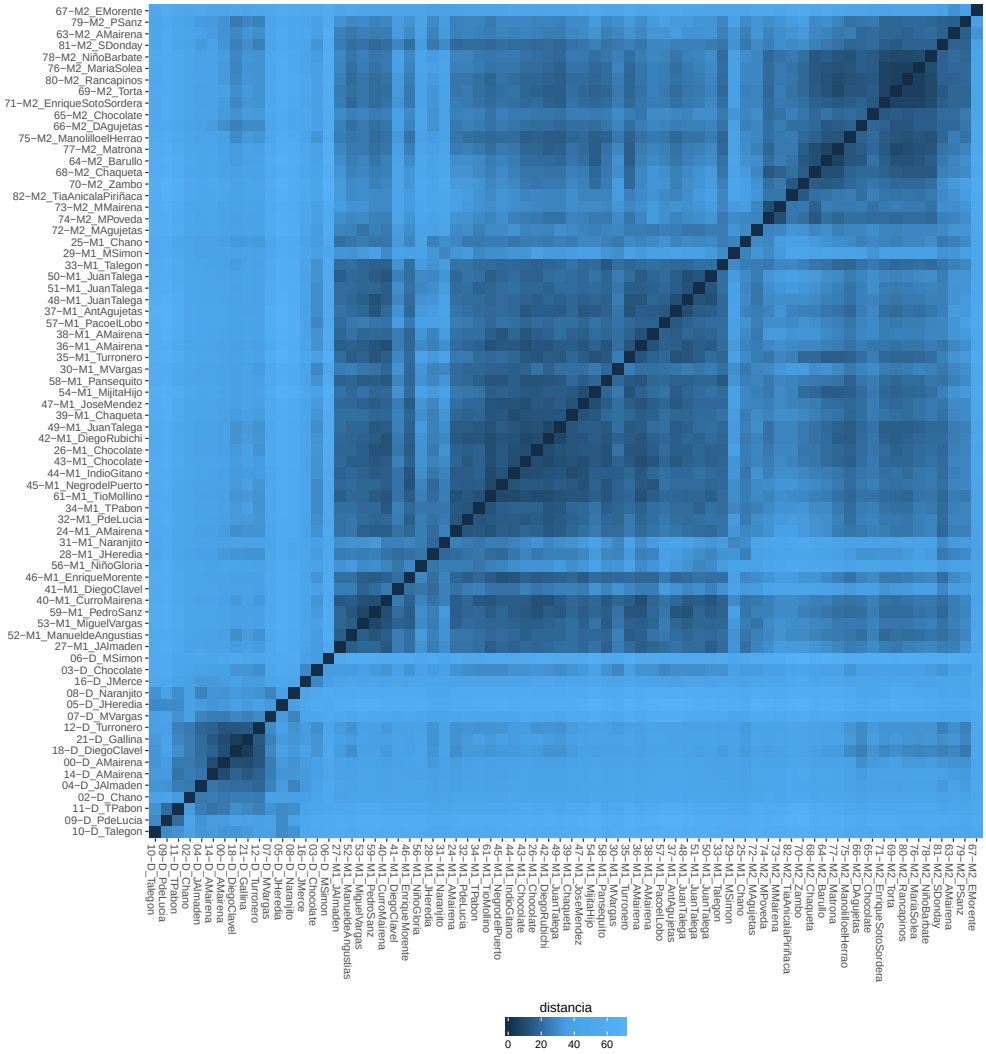


Figura 4.12: Matriz de distancias directas usando *edit distance*

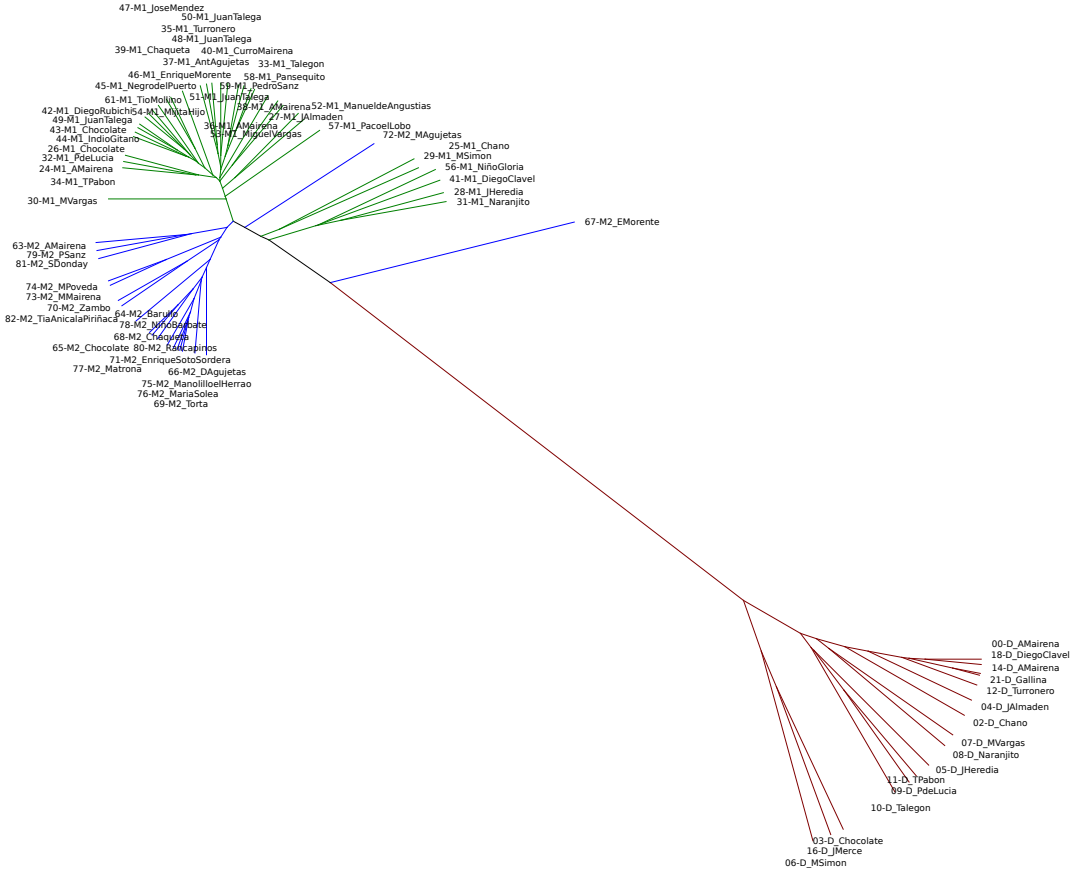


Figura 4.13: Arbol filogenético de distancias directas usando *edit distance*

		Agrupación		
		M1	M2	Debla
Real	M1	30	6	0
	M2	0	20	0
	Debla	0	0	16

Tabla 4.18: Matriz de confusión DE directa

M1	$Verdaderos\,Positivos = 30$	$Falsos\,Negativos = 6$
	$Falsos\,Positivos = 0$	$Verdaderos\,Negativos = 36$
	$Precisión = 1.000$	$Sensibilidad = 0.833$
	$F_1 = 0.909$	

Tabla 4.19: F_1 grupo Martinetes 1 usando *edit distance* directo

M2	$Verdaderos\ Positivos = 20$	$Falsos\ Negativos = 0$
	$Falsos\ Positivos = 6$	$Verdaderos\ Negativos = 46$
	$Precisión = 0.769$	$Sensibilidad = 1.000$
	$F_1 = 0.870$	

Tabla 4.20: F_1 grupo Martinetes 2 usando *edit distance* directo

Deblas	$Verdaderos\ Positivos = 16$	$Falsos\ Negativos = 0$
	$Falsos\ Positivos = 0$	$Verdaderos\ Negativos = 56$
	$Precisión = 1.000$	$Sensibilidad = 1.000$
	$F_1 = 1.000$	

Tabla 4.21: F_1 grupo Deblas usando *edit distance* directo

Micro average	$Verdaderos\ Positivos = 66$	$Falsos\ Negativos = 6$
	$Falsos\ Positivos = 6$	$Verdaderos\ Negativos = 138$
	$Precisión = 0.923$	$Sensibilidad = 0.923$
	$F_1 = 0.923$	

Tabla 4.22: F_1 agregado usando *micro average* de la agrupación *edit distance* directa

Macro average	$Precisión = 0.923$	$Sensibilidad = 0.944$
	$F_1 = 0.934$	

Tabla 4.23: F_1 agregado usando *macro average* de la agrupación *edit distance* directa

Grupo	M1	M2	Debla
Exactitud	0.917	0.917	1.000

Tabla 4.24: Exactitud agrupación *edit distance* directa

Agrupación de (Mora, Gómez, Gómez, & Díaz-Báñez, 2016)

El hecho de haber utilizado los mismos datos y el mismo mecanismo de evaluación de las agrupaciones que en (Mora, Gómez, Gómez, & Díaz-Báñez, 2016), permite realizar una comparación directa de resultados en el que el único elemento diferente es el algoritmo empleado para realizar la agrupación.

En el artículo citado, las distancias han sido calculadas a partir de descriptores musicológicos, distancia de edición y una combinación de estas empleando una distancia euclídea ponderada.

A continuación se muestra un resumen de los resultados publicados en el artículo ya citado:

Finalmente, la tabla que muestra la exactitud de referencia.

M1	*Precisión* $= 0.868$ *Sensibilidad* $= 0.917$ $F_1 = 0.892$

Tabla 4.25: F_1 grupo Martinetes 1 de referencia

M2	*Precisión* $= 0.778$ *Sensibilidad* $= 0.700$ $F_1 = 0.737$

Tabla 4.26: F_1 grupo Martinetes 2 de referencia

Debla	*Precisión* $= 0.813$ *Sensibilidad* $= 0.813$ $F_1 = 0.813$

Tabla 4.27: F_1 grupo Deblas de referencia

Micro average	*Precisión* $= 0.833$ *Sensibilidad* $= 0.833$ $F_1 = 0.833$

Tabla 4.28: F_1 agregado usando *micro average* de referencia

Macro average	*Precisión* $= 0.820$ *Sensibilidad* $= 0.810$ $F_1 = 0.815$

Tabla 4.29: F_1 agregado usando *macro average* de referencia

Grupo	M1	M2	Debla
Exactitud	0.889	0.861	0.917

Tabla 4.30: Exactitud agrupación de referencia

Comparación de resultados

El gráfico de la **figura 4.14** muestra los indicadores F_1 para los distintos grupos y algoritmos ensayados. En todos los grupos y agregados, los algoritmos propuestos mejoran al usado de referencia siendo la diferencia en los agregados de más de 10 puntos porcentuales.

Este resultado es importante porque la distancia de edición media al centroide es un algoritmo genérico diseñado sin especialización en el campo de la música y mejora significativamente a un algoritmo diseñado específicamente para esta tarea.

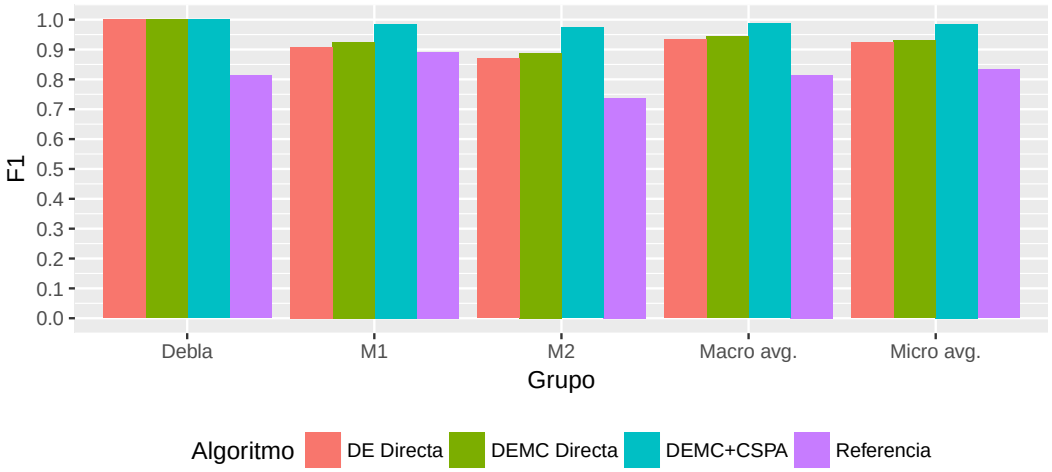


Figura 4.14: Comparación del indicador F_1 entre algoritmo DEMC+CSPA, DEMC directo y referencia

Si consideramos exclusivamente los algoritmos basados en la DEMC, el método de agrupación DEMC + CSPA supera (en los indicadores F_1) al DEMC directo, siendo la diferencia de unos 5 puntos percentuales.

Los resultados obtenidos son muy buenos con un F_1 superior al 0.9 (y en el mejor algoritmo de los propuestos superior a 0.98), resultados que validan el empleo de DEMC + CSPA para la agrupación de tonás (y presumiblemente para otros cantes).

4.3.3 *Extracción de descriptores diferenciadores*

Características específicas por tipo

En esta sección se va a presentar los resultados obtenidos respecto a qué descriptores son apropiados para identificar un tipo de toná o distinguir entre varios de estos tipos. La existencia de estos descriptores equivale a decir las características (registradas) que poseen estos tipos de cantes que los hacen específicos de los demás.

Es importante recordar que el límite de los resultados que se presentan, y las reglas de identificación asociadas, dependen del corpus que se tome de partida y del tratamiento de este. En el caso que nos ocupa se ha efectuado un análisis descriptivo[23], que emplea un corpus de tamaño reducido en el que no se ha aplicado ningún proceso de validación, en el que los resultados expuestos no deben extrapolarse a todo el posible universo de las

[4.23] Desde el punto de vista del análisis de datos existen dos grandes grupos funcionales: el análisis descriptivo que pretende describir características de los datos analizados y el inferencial que a partir de un cierto muestreo de datos, pretende determinar las características del conjunto completo de datos.
Los análisis de validación pretenden determinar un margen de error en datos inferidos, por lo que no tiene sentido aplicarse en un análisis descriptivo en el que se tienen disponibles todos los datos.

tonás. Así, cualquier regla o límite formulado ha de interpretarse desde este punto de vista.

Las definiciones de los descriptores usados se enuncian en el **capítulo 3**. En algunos casos, se ha sustituido el nombre genérico propuesto del descriptor en favor de uno con más aceptación en el campo del análisis musicológico. Cuando esta sustitución se produzca, se indicará la equivalencia entre los descriptores genéricos y los propios del campo.

A continuación, se efectúa un análisis cualitativo de algunos descriptores con capacidad de identificar cada tipo de cante analizado. En el **anexo C.1** está un listado con todos los descriptores específicos de cada tipo de toná y sus valores característicos.

Descriptores tipo para deblas

Los siguientes descriptores (y valores asociados) son característicos de las deblas. El listado se presenta organizado en bloques en función de la naturaleza del descriptor.

Los descriptores determinados nos permiten establecer características específicas de las deblas analizadas. Concretamente destacamos tres:

- La propiedad más sencilla identificada en las deblas es que la nota inicial es superior a la final.
- El resto de los descriptores con valores específicos para las deblas se centran en el cálculo de frecuencias de aparición: las notas más frecuentes y los intervalos de notas más frecuentes.
- La tercera característica es referente a que región de las deblas poseen más descriptores con valores específicos. En el caso que nos ocupa, el inicio del *tercio* (el primer tercio y la primera mitad) es más denso en descriptores con capacidad de identificar una debla.

A continuación, una descripción un poco más desglosada de estos resultados. Para hacer un listado más comprensivo de los descriptores, estos se han agrupado en función del principio común que los define.

Pendiente Como se ha indicado, las deblas tienen la nota inicial más aguda que la final (en el primer tercio). Esto puede apreciarse en los descriptores `Direccion`, `Pendiente`, `PendienteRegresion` y `Salto`.

Frecuencia de aparición de un símbolo Las deblas tienen, en el primer tercio del *tercio*, una presencia de la nota do (`'C'` en la notación anglosajona) superior al 13%. Esto se refleja en el descriptor `PClass_C_rango_0_a_33`[24].

[4.24] El descriptor `PClass_<S>`... (de *Pitch Class*) es el término musicológico para el descriptor descrito como `Frecuencia_<S>`.

Símbolos con mayor frecuencia de aparición Se ha detectado que algunas tríadas de notas más frecuentes en la parte inicial del *tercio*, son característicos en las deblas.

Las tríadas formados por las notas: la, si y do (`'ABC'`); la sostenido, si y do (`'AsBC'`); y si, do y re (`'BCD'`) son específicos de las deblas en el descriptor `PClass1_T_o_rango_0_a_33`[25]

[4.25] Equivalente a `N-gram_T_o_rango_0_a_33`.

Otros descriptores de la misma familia que se han verificado que tienen valores específicos son

- `PClass1_T_rango_0_a_50`,
- `PClass1_T_rango_0_a_33`,
- el ya mencionado `PClass1_T_o_rango_0_33`,
- `PClass1_T_rev_rango_0_33` y
- `PClass1_T_rev_o_rango_0_a_33`.

Como puede observarse, las tríadas características se localizan (salvo en un descriptor) en el primer tercio de las muestras.

Intervalo con máxima frecuencia de aparición El intervalo de máxima frecuencia de aparición (al inicio del tercio) es también característico. Los valores más típicos son los intervalos de si a la sostenido (`'BAs'`), de si a do (`'BC'`) o de do a si (`'CB'`).

Los descriptores específicos identificados son `PClass2_1_rango_0_a_50`, `PClass2_1_rango_0_a_33` y `Pclass2_1_rev_rango_0_a_33`.

Intervalos más frecuentes Finalmente, es posible obtener un perfil más delimitado de la melodía contabilizando los tres intervalos más frecuentes dentro de un rango.

Los descriptores de esta familia con valores específicos para deblas son:

Para la pieza completa,

- PClass2_T
- PClass2_T_o

Para la primera mitad de la muestra,

- PClass2_T_rango_0_a_50
- PClass2_T_o_rango_0_a_50

- PClass2_T_rev_rango_0_a_50
- PClass2_T_rev_o_rango_0_a_50

Para el primer tercio,

- PClass2_T_rango_0_a_33
- PClass2_T_o_rango_0_a_33
- PClass2_T_rev_rango_0_a_33
- PClass2_T_rev_o_rango_0_a_33

Para el segundo tercio,

- PClass2_T_rango_33_a_67
- PClass2_T_rev_rango_33_a_67
- PClass2_T_rev_o_rango_33_a_67

Como puede observarse, los descriptores de perfiles de intervalos con valores específicos para las deblas se centran, fundamentalmente, al inicio de las piezas.

Descriptores tipo para martinetes 1

El martinete 1 presenta menos características específicas que la debla. Aunque sigue observándose cómo los intervalos más relevantes son los del inicio del *tercio*.

Tríadas de notas más frecuentes En este caso, aparecen valores característicos para los martinetes 1 si se consideran las notas más frecuentes de toda la pieza (`PClass1_T`) y en caso del final (`PClass1_T_rango_50_a_100`)

Intervalos que más aparecen Se indican los descriptores con valores específicos separados por la región de aplicación. Nuevamente se aprecia como los intervalos más característicos son los existentes, fundamentalmente, al inicio de las muestras.

- PClass2_T

- PClass2_T_rango_0_a_50
- PClass2_T_o_rango_0_a_50
- PClass2_T_rev_rango_0_a_50

- PClass2_T_rango_0_a_33
- PClass2_T_o_rango_0_a_33
- PClass2_T_rev_rango_0_a_33

Descriptores tipo para martinete 2

Finalmente, los martinetes presentan características relevantes al inicio y al final de *tercio*. El único tipo de descriptor con valores específicos para este palo son los intervalos más frecuentes.

- PClass2_T

- PClass2_T_rango_0_a_50
- PClass2_T_o_rango_0_a_50
- PClass2_T_rev_rango_0_a_50

- PClass2_T_rango_0_a_33
- PClass2_T_o_rango_0_a_33
- PClass2_T_rev_rango_0_a_33
- PClass2_T_rev_o_rango_0_a_33

- PClass2_T_rango_50_a_100

- PClass2_T_rango_67_a_100

Efectos combinados de los descriptores

Una vez obtenido las características específicas de cada grupo, a modo de resumen se ha efectuado una gráfica que permite comparar las características individuales de cada tipo. La **gráfica 4.15** muestra una representación de descriptores[26] en función de la capacidad de estos de discriminar una o más categorías.

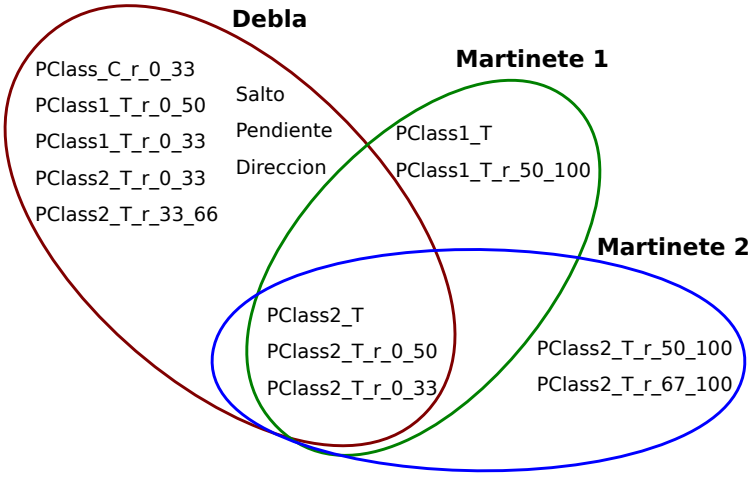


Figura 4.15: Diagrama de Venn descriptores clasificadores de tipo

[26] Se ha simplificado el diagrama eliminando descriptores cuya diferencia es el orden de procesado de las cadenas y resultados. Así, de `PClass2_T`, `PClass2_T_o`, `PClass2_T_rev` y `PClass2_T_rev_o` solo se mostrará el primero.

El diagrama de las tonás, al tener solo tres categorías, presenta propiedades interesantes que dependen de la propiedad dual de los descriptores[27]:

[4.27] Descrita en la sección **características específicas de un grupo** .

– La propiedad dual de los descriptores por tipo implica que una característica específica de una categoría es una característica común a las otras dos categorías.

  Así, en las deblas el descriptor `direccion` tiene el valor característico de descendente. La propiedad dual implica que los martinetes 1 y 2 comparten ambos la característica de una `direccion` horizontal o ascendente.

– No pueden aparecer descriptores que clasifiquen solo a dos tipos. Toda variable que sea capaz de justificar si pertenece o no a una categoria *A* y simultáneamente si pertenece o no a la categoría *B*, automáticamente es capaz de justificar la pertenencia a la categoría *C*.

Los valores del descriptor `PClass2_T` (los intervalos más frecuentes de cada pieza) son característicos para cada tipo de toná examinado lo que permite identificar las piezas examinando exclusivamente esta característica. Es más, dado que los valores de los descriptores `PClass2_T_rango_0_50` y `PClass2_T_rango_0_33` son específicos para cata tipo de toná examinada, implica que el estudio de los intervalos en la primera mitad, e incluso, en el primer tercio del *tercio* es suficiente para identificar el tipo toná que está sonando.

Descriptores de clasificación global

El segundo análisis que se ha efectuado sobre los descriptores es la búsqueda de el conjunto mínimo de descriptores capaz de reflejar las características específicas de cada categoría presente. La **figura 4.15** ya anticipó que ciertos descriptores de la familia del descriptor `PClass2_T` (intervalos más frecuentes) cumplían dicho requisito.

Adicionalmente, el empleo del análisis bayesiano para determinar la probabilidad de que un tercio sea de un tipo dado, nos permite determinar el margen (o confianza) de dicha clasificación. La **tabla 4.31** muestra todos los descriptores capaces de identificar correctamente las piezas (empleando un solo descriptor) y el margen de confianza[28] de dicho estimador.

[4.28] Que recordamos que consistía en la diferencia entre la certeza de la categoría escogida y la segunda categoría más probable.

Los márgenes, como pueden observarse son muy amplios (del orden del 95%) por lo que determinan sistemas muy robustos de clasificación.

Descriptor	Margen
PClass2_T_o_rango_0_a_50	0.9649485
PClass2_T_o_rango_0_a_33	0.9603936
PClass2_T_rango_0_a_33	0.9533544
PClass2_T_rev_o_rango_0_a_33	0.9517268
PClass2_T_rango_0_a_50	0.9513214
PClass2_T	0.9490839
PClass2_T_rev_rango_0_a_50	0.9472378
PClass2_T_rev_rango_0_a_33	0.9425255

Tabla 4.31: Descriptores globales ordenados por margen (cuanto mayor, mejor)

Es interesante observar los valores que presentan estos descriptores con capacidad de discriminar entre los distintos tipos de tonás. Las gráficas de la **figura 4.16** muestran los valores para los dos mejores descriptores identificados `PClass2_T_o_rango_0_a_50` y `PClass2_T_o_rango_0_a_33`. Ambos descriptores listan los tres intervalos más frecuentes para cada pieza (en la primera mitad de la pieza y el primer tercio, respectivamente). Como un intervalo es el salto de una nota a la siguiente, es posible representarlo como un punto en un diagrama cartesiano, por lo que todo valor derivado de `PClass2_T` se puede representar en el plano como tres puntos unidos por dos segmentos.

Así se ha hecho en la **figura 4.16** en el que, además, se ha usado el color para identificar la categoría de toná asociado a cada valor representado (rojo para deblas, verde para martinetes 1 y azul para martinetes 2).

Lo primero que se observa es que la mayoría de intervalos consignados están situados sobre la diagonal principal del gráfico. Más concretamente sobre la franja sombreada sobre dicha diagonal. Esta franja identifica al unísono (intervalos en que la nota inicial y final es la misma) y a los intervalos conjuntos (en el que el salto entre las notas es de 1 o 2 semitonos). Este gráfico confirma una característica del cante flamenco en el que la mayoría de los intervalos son conjuntos.

Si dividimos por palo, todos los intervalos consignados en las deblas son conjuntos, seguido por los martinetes 2 que tienen algunos intervalos fuera de la zona y por último los martinetes 1 que tienen gran cantidad de intervalos disjunto (de 3 o más semitonos).

Otro factor observable es el de la riqueza melódica. La relación entre melodías y un descriptor de la familia tipo `PClass2_T` es

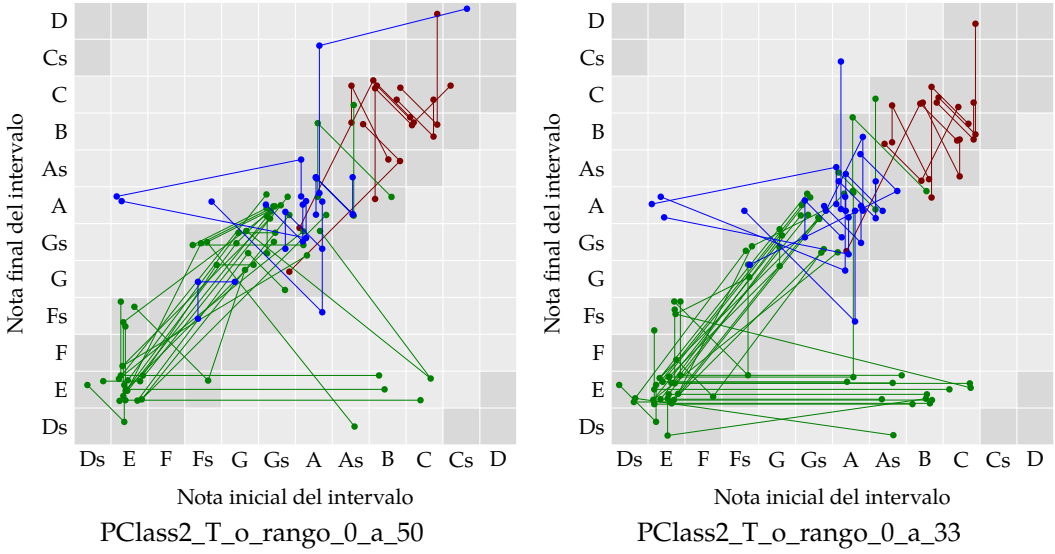


Figura 4.16: Visualización de los descriptores de clasificación global en tonás

sobreyectiva: para cada posible valor del descriptor existen múltiples melodías que lo generan en las que, por construcción, tienen cierta similitud entre sí. En cambio, distintos valores del descriptor provienen de melodías distintas. De ahí se deduce que los palos con mayor número de valores de estos descriptores implican una mayor riqueza melódica.

En la gráfica se observa el mayor número de líneas presentes en verde (martinetes 1) que en rojo (deblas) lo que implica que la riqueza melódica de los martinetes 1 es superior al de las deblas. Este resultado ya pudo observarse en la **figura 4.9** en el que las ramas de las deblas estaban mucho más compactas que en los martinetes 1.

La gráfica también da información del conjunto de notas de paso más importantes. Las deblas están centradas en las notas do (C) y si (B), los martinetes 1 sobre la nota mi (E), la (A) y sol sostenido (Gs) y los martinetes 2 sobre la nota la (A).

Por último, es posible determinar los efectos de tomar un rango de notas mayor o más pequeño. Comparamos los gráficos del descriptor `PClass2_T_o_rango_0_a_50` (el mejor descriptor clasificador hallado) con `PClass2_T_o_rango_0_a_33` (el segundo mejor), se observa que el segundo presenta una mayor densidad de líneas en los martinetes 1. Como vimos antes, las líneas están relacionadas con la variedad melódica, lo que significa que los inicios del martinete 1 es más variado; pero conforme se consideran más notas, los intervalos relevantes tienen a estabilizarse reduciendo la variedad general del inicio del martinete.

Los martinetes 2 y las deblas no presentan este efecto, lo que indica que el desarrollo melódico del la primera mitad de la pieza es más homogéneo.

4.3.4 *Extracción de arcos melódicos*

La determinación de arcos significativos, depende de los algoritmos de selección de motivos y de las técnicas de combinación de estos. En esta sección se discutirán los distintos resultados obtenidos durante el proceso de gestión de arcos sobre el corpus de tonás.

La construcción de un diccionario de arcos musicales significativos es análogo al proceso de construcción de un diccionario de palabras sobre un texto. Esta equivalencia permite aplicar las técnicas existentes de análisis de texto sobre melodías musicales. Sólo por citar un par de ellas, fuera ya del alcance del presente trabajo, se mencionará el tf-idf (*Term frequency–inverse document frequency*) (Aizawa, 2003; Ramos, et al., 2003) que evalúa la importancia de una palabra en un documento de un corpus o el *Latent Dirichlet Allocation* (Blei, Ng, & Jordan, 2003) que descubre conjuntos de palabras relacionados con un mismo tema. En ambos casos se pueden emplear los arcos (y los motivos que los definen) como sustituto del concepto de palabra y aplicarse con estas técnicas sin más modificaciones.

A continuación se presentan los distintos resultados obtenidos en la búsqueda de motivos musicales y posteriormente los correspondientes a los arcos.

Determinación de motivos

De todas la herramientas presentadas, el análisis que emplea el reconocimiento de patrones sintácticos es el más difícil de evaluar debido a su naturaleza cualitativa. Aun así, se considera que es una herramienta exploratoria válida[29]. Es más, en el campo del análisis de las tonás no se ha encontrado un estudio previo (ni teórico ni empírico) con los que comparar los resultados que aquí se proponen, por lo que los resultados se expondrán esperando posteriores trabajos que validen la utilidad de estos desarrollos.

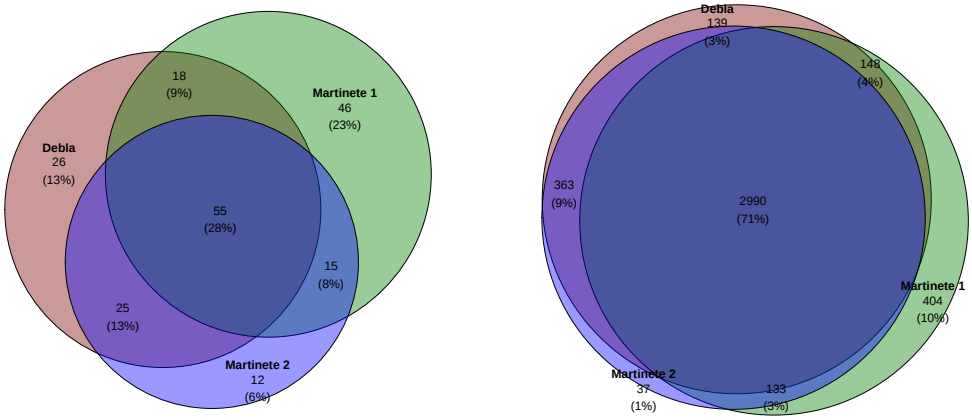
[4.29] Nadie pondría en duda la utilidad de los histogramas, aunque se nos antoja difícil una evaluación objetiva de la herramienta.

Distribución de motivos por palo

Para la construcción de los motivos se comentará, inicialmente, el algoritmo Re-Pair viendo, posteriormente, las diferencias con los otros algoritmos. Tras la inducción de la gramática y la identificación de motivos que aparecen en el corpus de tonás,

es posible determinar en qué categorías aparece cada motivo. Se han identificado 200 motivos distintos que aparecen un total de 4214 veces (en la **figura 4.17** se muestra la distribución de los motivos según los palos en los que aparecen).

Comparando los dos diagramas, se aprecia que existe un grupo reducido de motivos (los comunes a todos los palos) que tienen un crecimiento porcentual importante cuando consideramos el número de apariciones del grupo. En media, estos motivos aparecen 54 veces cada uno. Por contra, los grupos de motivos interesantes para la clasificación (aquellos motivos que solo aparecen en una única categoría) suman un total del 42% de los motivos únicos (algo menos de la mitad de los motivos identificados) que cuando los consideramos respecto el número de apariciones representan sólo un 14% de todos los encontrados (lo que en media representa 7 apariciones por motivo identificador).

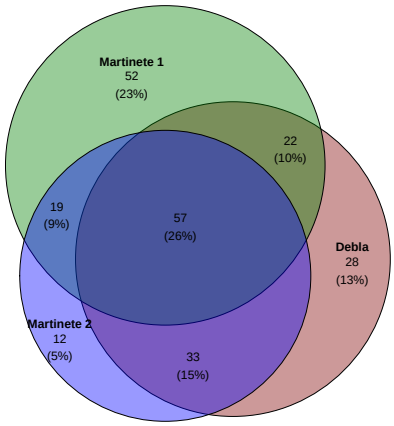


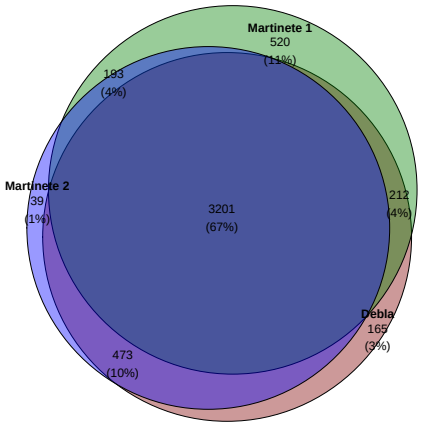
Motivos distintos Motivos encontrados

Figura 4.17: Proporción de motivos en Re-Pair

Estos porcentajes tan bajos de aparición de los motivos específicos de cada palo, hacen difícil la categorización de las piezas por la localización de estos motivos.

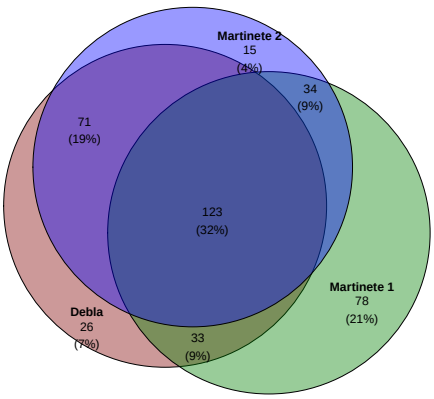
Estos resultados, sobre los motivos encontrados usando Re-Pair, son similares a los encontrados usando el algoritmo Sequitur (en la **figura 4.18**) y Lempel-Ziv-Welch (**figura 4.19**). El test Kolmogorov-Smirnov (Massey, 1951) confirma que los distintos algoritmos proporcionan distintos muestreos de motivos la impresión de que son series relacionadas dando un $p-value = 0.5752$ entre cada par de series.

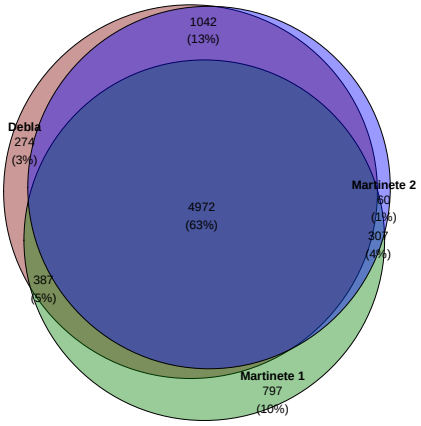




Motivos distintos Motivos encontrados

Figura 4.18: Proporción de motivos en Sequitur





Motivos distintos Motivos encontrados

Figura 4.19: Proporción de motivos en LZW

Las figuras **4.20** y **4.21** muestran la misma debla en la que se han identificado los motivos empleando los catálogos de Sequitur, Re-Pair y Lempel-Ziv-Welch. Podemos observar que:

– De los tres algoritmos, Re-Pair es el que identifica menos motivos. A diferencia de los otros algoritmos empleados (que son de búsqueda voraz) Re-Pair trabaja a partir de la pieza completa, por lo que en todo momento busca los motivos considerados óptimos. Frente a esta estrategia, Sequitur

Figura 4.20: Extracción de motivos Sequitur y Re-Pair

Figura 4.21: Extracción de motivos Lempel-Ziv-Welch

y LZW van identificando los motivos conforme se va analizando la cadena. Este enfoque obliga a tomar decisiones sin tener información completa de la misma.

– El enfoque predictivo de LZW hace que se añadan motivos en el catálogo antes de saber si la misma secuencia será interesante en un futuro o no (al no saber si ésta volverá a aparecer). Esta estrategia hace que el tamaño del catálogo de LZW sea superior a las otras expuestas ya que incluye símbolos no terminales no usados además de otros que si vuelven a aparecer[30].

  El catálogo de LZW es, por tanto, más grande que los basados en digramas (que solamente incluyen símbolos usados).

– En LZW, cada motivo se produce extendiendo otro motivo con un único nuevo token. De esta forma, para que haya un símbolo no terminal de tamaño n debe existir uno previo de tamaño $n-1$, por lo que el crecimiento de los símbolos es lineal.

[30] Cuando se impone un filtro de motivos por frecuencia se elimina este problema ya que los motivos que están en el catálogo; pero que no se usan, pueden eliminarse.

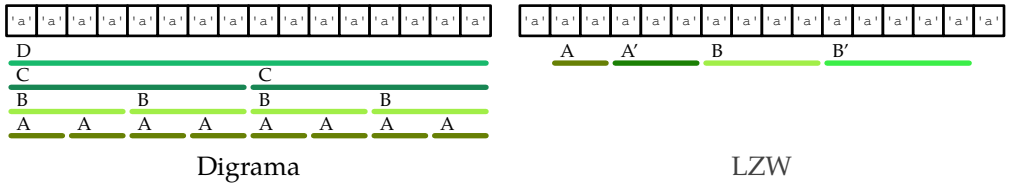


Figura 4.22: Sistema de digramas frente a LZW

En cambio, en los sistemas basados en digramas, cada motivo se produce por la combinación de dos motivos previos, por lo que a partir de un motivo de n tokens se puede construir un nuevo motivo de tamaño $2n$ siendo el crecimiento exponencial.

– Si bien los motivos detectado por cada algoritmo no tienen por qué coincidir, existen motivos que aparecen simultáneamente en dos o en más procedimientos[31].

[31] Para facilitar la identificación de motivos idénticos, se ha establecido una firma de color para identificar motivos independientemente del algoritmo empleado. Colores idénticos representan motivos idénticos.

Arcos melódicos

Arcos como un grafo

El diccionario de motivos identificados permite la construcción de un grafo en el que los motivos son nodos y los arcos del grafo

representan a los arcos melódicos de un mismo tercio. El grafo resultante es tan denso de nodos y arcos que resulta difícil de interpretar, por eso es útil podar el grafo para facilitar la visualización y posteriormente la interpretación del mismo. La **figura 4.23** es un ejemplo de un grafo generado a partir de los motivos identificados usando LZW tras un proceso de filtrado. Concretamente, en este grafo sólo se muestran arcos que aparezcan exclusivamente en una categoría[32] y, además, solo se representan motivos de una longitud mínima de 5 notas y que aparezcan, al menos, 4 veces en el corpus.

[4.32] Identificada por el color del arco: rojo para deblas, verde para martinetes 1 y azul para martinetes 2.

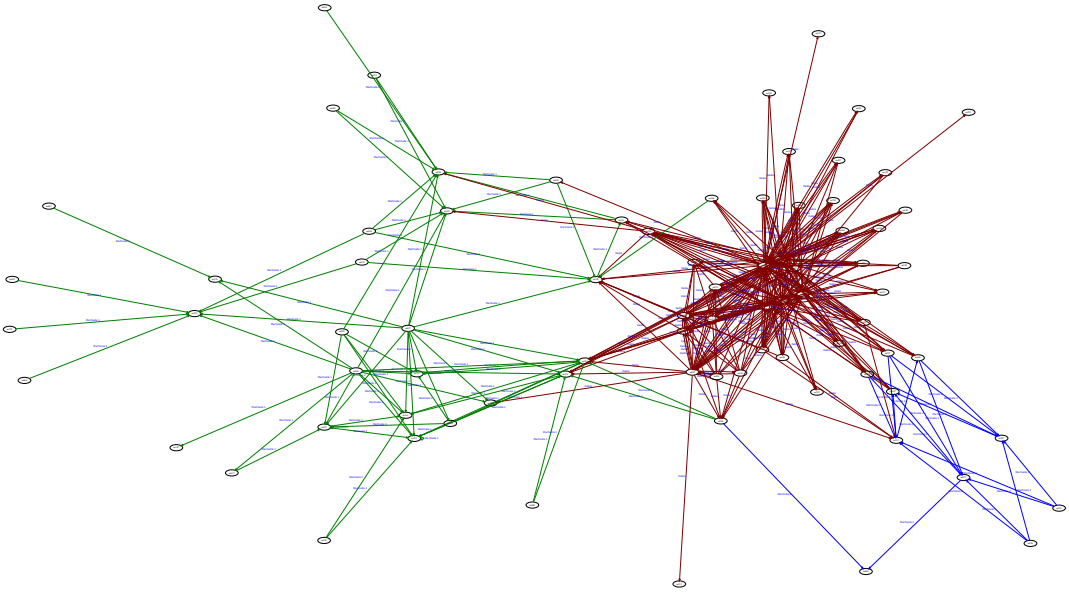


Figura 4.23: Grafo LZW con motivos de longitud mínima 5 y arcos con frecuencia de aparición mínima de 4

Aun con las limitaciones producidas por el filtrado, es posible confirmar alguna conclusión realizada cuando se analizó el árbol filogenético de las tonas (**figura 4.9**). Cada categoría forma un subgrafo conexo en el que hay nodos internos a la categoría (aquellos nodos cuyos arcos de entrada o salida son todas del mismo color) y nodos frontera (que son los nodos que soportan arcos de distintas categorías). Es posible usar los nodos frontera como medida de motivos musicales comunes entre distintos palos. La **figura 4.23** muestra como los martinetes 2 (en azul) tienen una frontera compartida con las deblas mayor que la frontera con los martinetes 1 (compuesta por un sólo nodo) lo que confirma que, melódicamente hablando, los martinetes 2 están más relacionados con las deblas que con los martinetes 1.

Esta conclusión sobre el grafo mostrado se mantiene, independientemente del algoritmo empleado para la selección de los

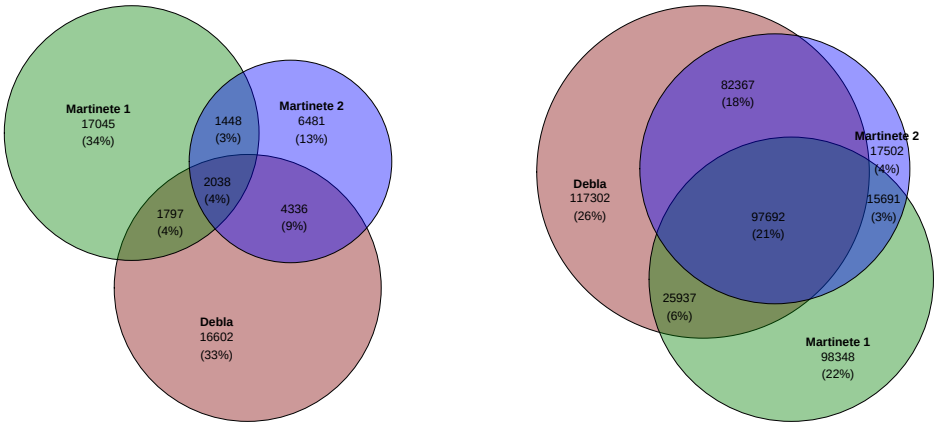
motivos y del nivel de restricción escogido para los filtros (obviamente, si los filtros son demasiado restrictivos el grafo terminará siendo un conjunto de subgrafos no conexos).

Distribución de arcos por palo

Al igual que se hizo con los motivos, se va a mostrar las proporciones existentes de los arcos en función de los palos en los que aparecen. Nuevamente, los resultados son bastante consistentes independientemente del algoritmo de extracción de motivos empleado. En esta ocasión, se solo se comentará los efectos de emplear el algoritmo LZW (mostrados en la **figura 4.24**) para extraer los motivos y simplemente se mostrarán las gráficas de distribución para el resto de algoritmos empleados.

En la muestra de tonás, se han localizado 380 motivos distintos con el algoritmo LZW que dan un número 144400 de hipotéticos arcos distintos. En el corpus examinado, se han encontrado 49747 arcos distintos que representan el 34.5% de todos los posibles. Estos arcos han aparecido un total de 454839 veces (una media de 9.14 apariciones por arco).

La **figura 4.24** muestra la proporción de arcos determinados en cada categoría. El diagrama de Venn de arcos distintos hace referencia a proporciones de arcos sin considerar cuántas veces aparece dicho arco y a su derecha se repiten las proporciones considerando, esta vez, el número total de arcos encontrados.



Arcos distintos Arcos encontrados

Figura 4.24: Proporción de arcos en LZW

El empleo de arcos como elemento caracterizador de las piezas (frente al empleo de motivos solamente) es bastante relevante. Si

se intenta identificar el palo de una pieza atendiendo a la categoría asociada a cada motivo identificado, casi dos tercios (63%) de todos los motivos encontrados aparecen simultáneamente en todos los palos analizados, por lo que no tienen utilidad para caracterizar las piezas. En cambio, el porcentaje de arcos encontrados que aparecen simultáneamente en piezas de los tres palos se reduce hasta un quinto (21%) de todos los arcos identificados.

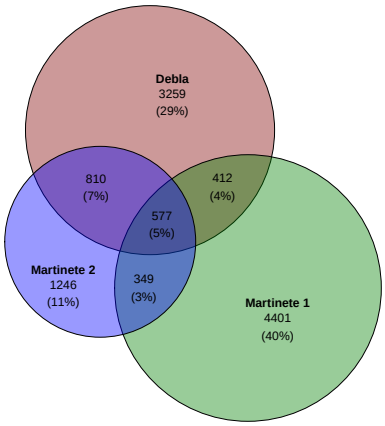
Conclusiones similares pueden derivarse de los porcentajes de arcos específicos de un palo, en el que pasamos del 3% de motivos encontrados específicos de una debla a un 26% de arcos específicos de la debla.

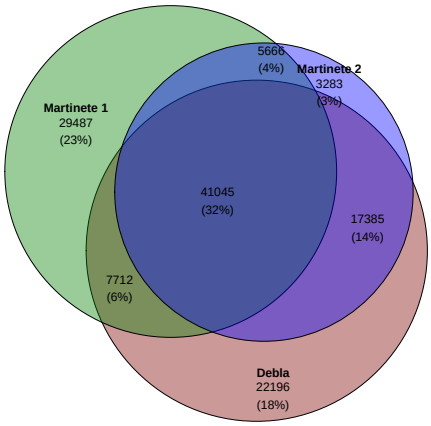
Este efecto que se produce al usar arcos en vez de motivos, también se reproduce si se consideran las proporciones sobre motivos distintos frente a arcos distintos.

Esta gráfica (y las similares con los otros algoritmos empleados) demuestra la ventaja al emplear arcos frente al uso de simples motivos ya que tienden a destacar las variaciones melódicas de cada palo frente a las melodías comunes de todos los palos.

Adicionalmente a las consideraciones sobre las proporciones de los arcos comunes a todos los palos o a los específicos de un único palo, es posible observar otras características a partir de las proporciones cruzadas entre palos: el porcentaje de arcos comunes entre deblas y martinetes 2 (39%) es superior al de martinetes 1 y 2 (24%) que, nuevamente confirma, la mayor similitud existente entre martinetes 2 y deblas que a los martinetes 1.

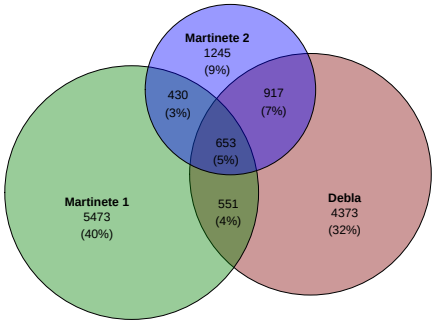
Tal y como se ha comentado, las variaciones producidas en las proporciones al considerar arcos en vez de motivos son similares independientemente del algoritmo escogido. Las figuras **4.25** y **4.26** muestran las proporciones encontradas en función del algoritmo de extracción de motivos.

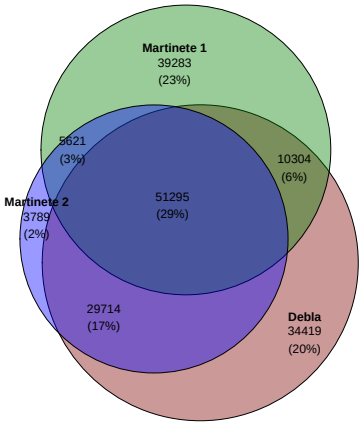




Arcos distintos Arcos encontrados

Figura 4.25: Proporción de arcos encontrados en Re-Pair





Arcos distintos Arcos encontrados

Figura 4.26: Proporción de arcos encontrados en Sequitur

Selección de arcos

Al hablar de los grafos, como una forma de visualizar los arcos, se mencionó la utilidad del filtrado de arcos como técnica para reducir el exceso de información. Queda escoger el tipo de filtrado más apropiado para captar características interesantes en las melodías flamencas y los parámetros más útiles en dichos filtros.

En general, hay dos características deseables en la selección de arcos:

Longitud del arco Arcos de longitud elevada representarán formas musicales más identificables y representativas del estilo.

Frecuencia de aparición del arco Cuanto mayor sea la frecuencia de aparición de un arco, más relevante será éste.

Lamentablemente, como suele suceder en la aplicación de filtros, el uso de un filtro con valores muy restrictivos reduce el número de arcos disponibles para un posterior análisis. La **figura 4.27** muestra el número de arcos (en media) que quedan en cada pieza al aplicar simultáneamente un filtro de longitud y frecuencia. Como puede verse, el algoritmo que más arcos encuentra es Re-Pair (con más de 500 arcos por pieza de media cuando no se aplica ningún filtro). Análogamente, el algoritmo LZW encuentra, con las mismas condiciones, 80 arcos en media.

Aunque el criterio de arcos en media permite medir cierto desempeño en la capacidad de extracción de los algoritmos presentados, no puede considerarse como un criterio definitivo de selección de los umbrales de los filtros ya que no tiene en cuenta la distribución de los arcos en los distintos tercios examinados. El concepto de arcos en media no informa de la cantidad de piezas que no contienen ningún arco melódico (debido a filtros muy restrictivos). Cuando esto ocurre, las piezas que, tras el filtrado, no presentan arcos melódicos quedan excluidas de posteriores análisis por lo que no estas no aportarán a los resultados.

Se establece, por tanto, un segundo criterio de selección de umbrales en los filtros que se basa en el número de piezas que presentan al menos un arco. La **figura 4.28** (en la **página 179**) muestra los valores calculados en tanto por uno.

Analizando conjuntamente la media de arcos por piezas y el porcentaje de piezas con arcos, se puede evaluar los umbrales para los filtros de arcos. Si bien el método puede emplearse con otros conjuntos de datos, los valores obtenidos son específicos del corpus actual y no son universales para otros campos de estudio (ni siquiera para otros corpus dentro del campo de la música flamenca).

A tenor de las gráficas de barrido de filtros, podemos afirmar (para el corpus de tonás) que:

– El algoritmo LZW presenta un mal desempeño tanto por arcos localizados como por porcentaje de piezas con arco. Este

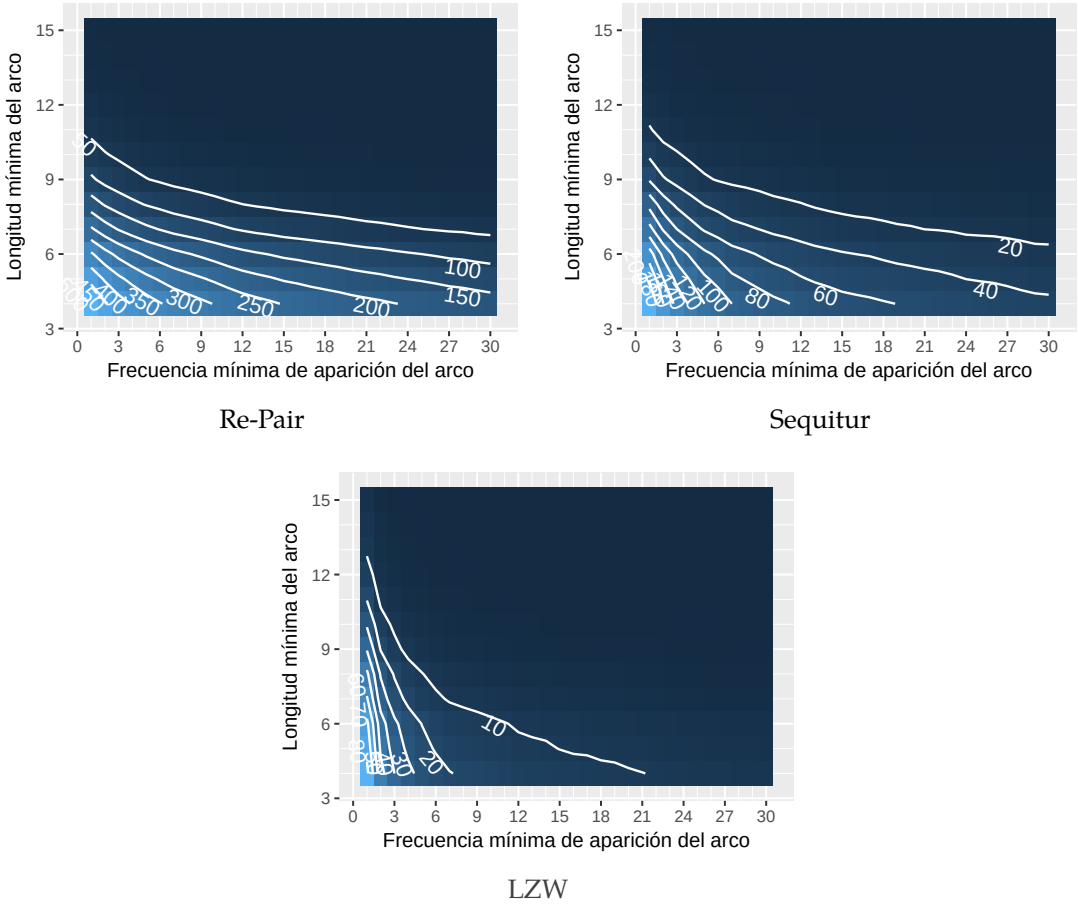


Figura 4.27: Arcos en media detectados en función de los filtros aplicados

mal desempeño descarta LZW entre los algoritmos a usar en posteriores análisis.

– El algoritmo que presenta mayor número de arcos es Re-Pair a costa de la longitud de los mismos. [33]. Re-Pair presenta muchos arcos de longitud mínima 4 (2 notas, hueco y 2 notas más).

  Esto es, Re-Pair fomenta la alta frecuencia de arcos sacrificando la longitud de los mismos.

– Por contra, Sequitur es capaz de identificar arcos más largos a costa de que la frecuencia de aparición de los mismos sean inferiores.

– La región del 100% de piezas con arcos es muy reducida en área y solo está presente para valores umbrales de tamaño y

[33] Este resultado tiene sentido, ya que el mecanismo de selección de Re-Pair se basa en buscar los términos de máxima repetición. De hecho, si se aplica exclusivamente un filtro de frecuencia, el 100% de piezas presentan al menos un arco con una frecuencia mínima de 90 apariciones (no mostrado en la **gráfica 4.28** para no distorsionar la gráfica).

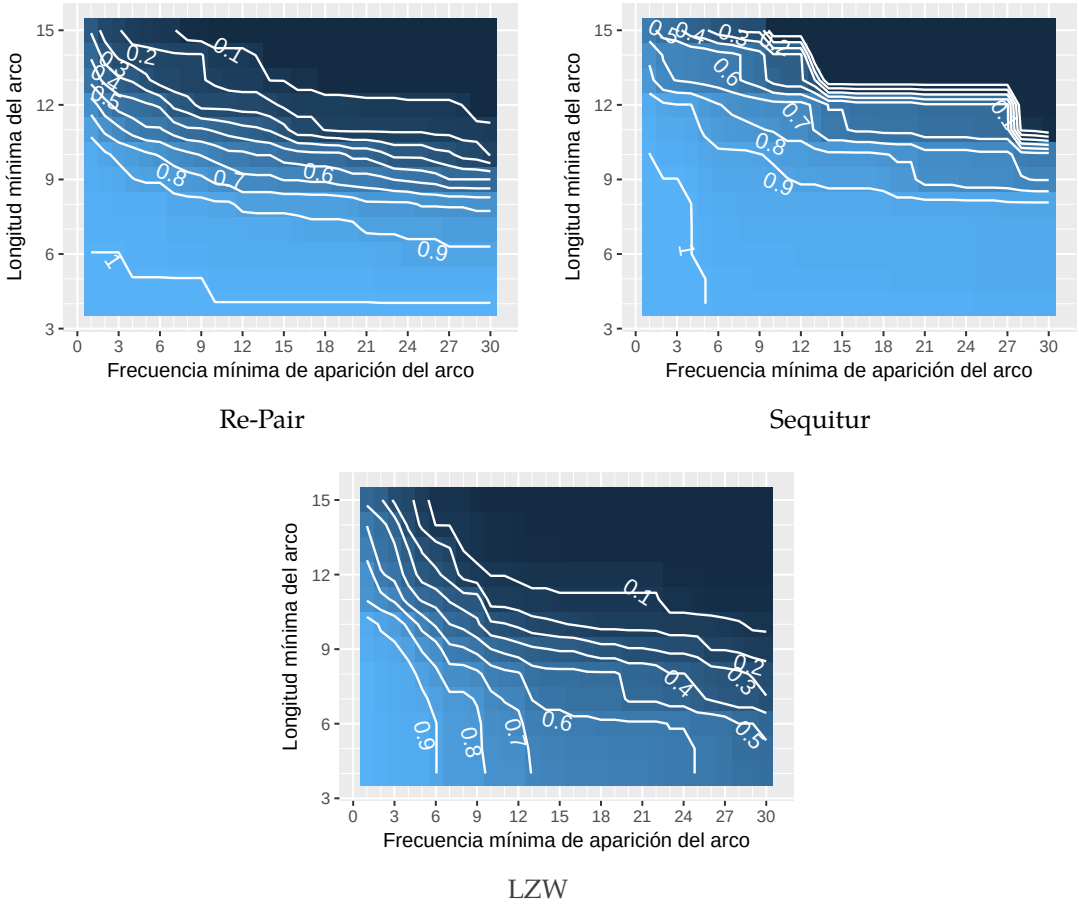


Figura 4.28: Piezas (en tanto por uno) que presentan al menos un arco en función de los filtros aplicados

frecuencia reducidas. Con el fin de aumentar la calidad de los arcos encontrados, se ha optado por usar parámetros de filtrado más restrictivos que proporcionen arcos de cierto tamaño y frecuencia de aparición (al precio de un cierto número de piezas que se queden sin ningún arco presente).

Los límites escogidos para el filtro, para el corpus que nos ocupa, son arcos con una frecuencia mínima de aparición de 4 veces y una longitud total mínima (la suma de las longitudes de los dos motivos que lo forman) de 12 notas. Si bien este límite descarta algunas piezas en el estudio, proporciona arcos más relevantes (musicalmente hablando) al ser más largos y frecuentes.

Una vez seleccionados los arcos tras el filtrado, estos presentan dos virtudes que nos ayudan a caracterizar los cantes y las categorías a las que pertenecen. La primera está relacionado con

la relevancia de los arcos obtenidos. Consideramos que los arcos identificados son relevantes debido a su frecuencia de aparición y la longitud de estos. Aunque musicalmente pueden existir otros motivos de relevancia, como pueden ser ciertas reglas (o alteración de estas) de composición musical, no entraremos a discutirlos ya que entraría dentro de un análisis musicológico más profundo que escapa del contenido de este trabajo.

Si bien podría argumentarse que un determinado adorno o elemento accidental podría tener una frecuencia de aparición elevada, si este elemento aparece con una frecuencia elevada se convierte en parte consustancial al estilo y, por tanto, se debe considerar parte estructural del mismo.

Esta propiedad de la relevancia por frecuencia y longitud se garantiza por medio de los parámetros de los filtros empleados.

La segunda propiedad de los arcos es que establece una relación de orden de aparición entre motivos (posiblemente distantes) dentro de las piezas. De esta forma, es posible identificar motivos que siempre aparecen al inicio de las piezas o al final o incluso parejas de motivos que aparecen en determinada secuencia. De nuevo, esta propiedad es fruto del procedimiento de construcción empleado.

La **figura 4.29** (en la **página 182**) muestra las primeras 9 piezas del corpus de tonás con los arcos identificados[34]. Estos arcos mostrados son los específicos del palo de las deblas que quedan tras el filtrado que requiere de una frecuencia mínima de 4 apariciones y una longitud mínima de 12 elementos.

[4.34] En este caso dado que nos interesa más una visión de conjunto que valores concretos, se han eliminado los identificadores de los motivos dejando solamente el código de color como identificación. El número a la izquierda de cada arco representa la frecuencia de aparición del mismo dentro del corpus.

Los arcos destacan partes estructurales de las piezas.

Al visualizar los arcos se aprecia un patrón que se repite en las piezas: los motivos constituyentes de los arcos (específicos del palo) no están repartidos homogéneamente a lo largo de las piezas sino que se concentran en zonas concretas de las mismas. Además, existen formaciones de motivos que se repiten en distintas piezas del mismo estilo. El ejemplo más claro se puede apreciar en las dos últimas piezas mostradas (08-D_Naranjito y 09-D_PdeLucia): en ambas aparecen los mismos motivos[35] en el mismo orden. Si llamamos «Arcada» a una sucesión de arcos en el que el segundo motivo de uno es el primero del siguiente, estas piezas presentan una arcada de 3 arcos (o 4 motivos).

[4.35] Recordemos que la igualdad del color en los motivos marcados implica motivos idénticos.

Como se muestra, todas las piezas están organizadas en arcadas. Las arcadas determan una estructura de tipo jazzística en la que las piezas tienen unas partes fijas (las llamadas estructurales) que están representadas por los motivos identificados y unas notas cubriendo los vanos de los arcos que identificaremos como

regiones accidentales, partes ornamentales o expansiones musicales (en función de la intención funcional que se quiera atribuir a las mismas).

El grado de restricción impuesto por los filtros hace que haya algunas piezas como la 03-D_Chocolate o 06-D_MSimón (piezas tercera y sexta de la figura) que no muestran ningún arco. La reducción de la exigencia de los filtros, aumenta la presencia de los arcos al coste de dificultar la visualización de las arcadas. Dado que lo que interesa en esta exposición es la claridad en la visualización de los datos, se han escogido umbrales restrictivos (con el coste de las ya mencionadas piezas sin arcos).

La **figura 4.30** (en la **página 183**) muestra las mismas piezas en los que se han identificado los arcos empleando el algoritmo Sequitur. Como se ha comentado anteriormente, los motivos con los que se constituyen los arcos no tienen por qué ser coincidentes y aunque en esta ocasión se observa un cierto solapamiento de motivos, siguen existiendo zonas de localización de motivos y zonas de hueco. La reducción de exigencia de los filtros en ambos casos hace que los arcos identificados converjan independientemente del algoritmo empleado.

Los arcos identificados permiten hacer una descripción melódica de las piezas analizadas. A continuación se mostrará una selección de las arcadas encontradas según el tipo de toná que estemos viendo. Las figuras **4.31**, **4.32** y **4.33** muestran los arcos desde un punto de vista que destaca el aspecto musical. En las deblas y martinetes 1, se muestran las arcadas de dos arcos más relevantes[36]. En el caso de los Martinetes 2 se presentan exclusivamente arcadas de un único arco (ya que se mostró que tenían una menor riqueza de arcos que los otros estilos).

[4.36] Considerando tanto la frecuencia de aparición como la longitud de los motivos.

El gráfico de puntos que acompaña, es una representación espacial de los valores que componen las cadenas donde las abcisas indican el tamaño de los motivos y las ordenadas representan los semitonos de distancia a la nota `'mi'` (que coincide con el valor del token o símbolo usado para describir la cadena). Con objeto de facilitar la visualización y comparación de motivos, estos han podido sufrir algún desplazamiento: un ligero desplazamiento vertical para impedir el solapamiento completo de notas de dos motivos distintos y uno horizontal que permita sincronizar motivos de distinta duración.

Simultáneamente a los gráficos de puntos se adjuntan los mismo motivos en pentagrama con las siguientes salvedades a la notación convencional: las notas no tienen duración asociada (debido al proceso de muestreo de las piezas) y se ha usado una

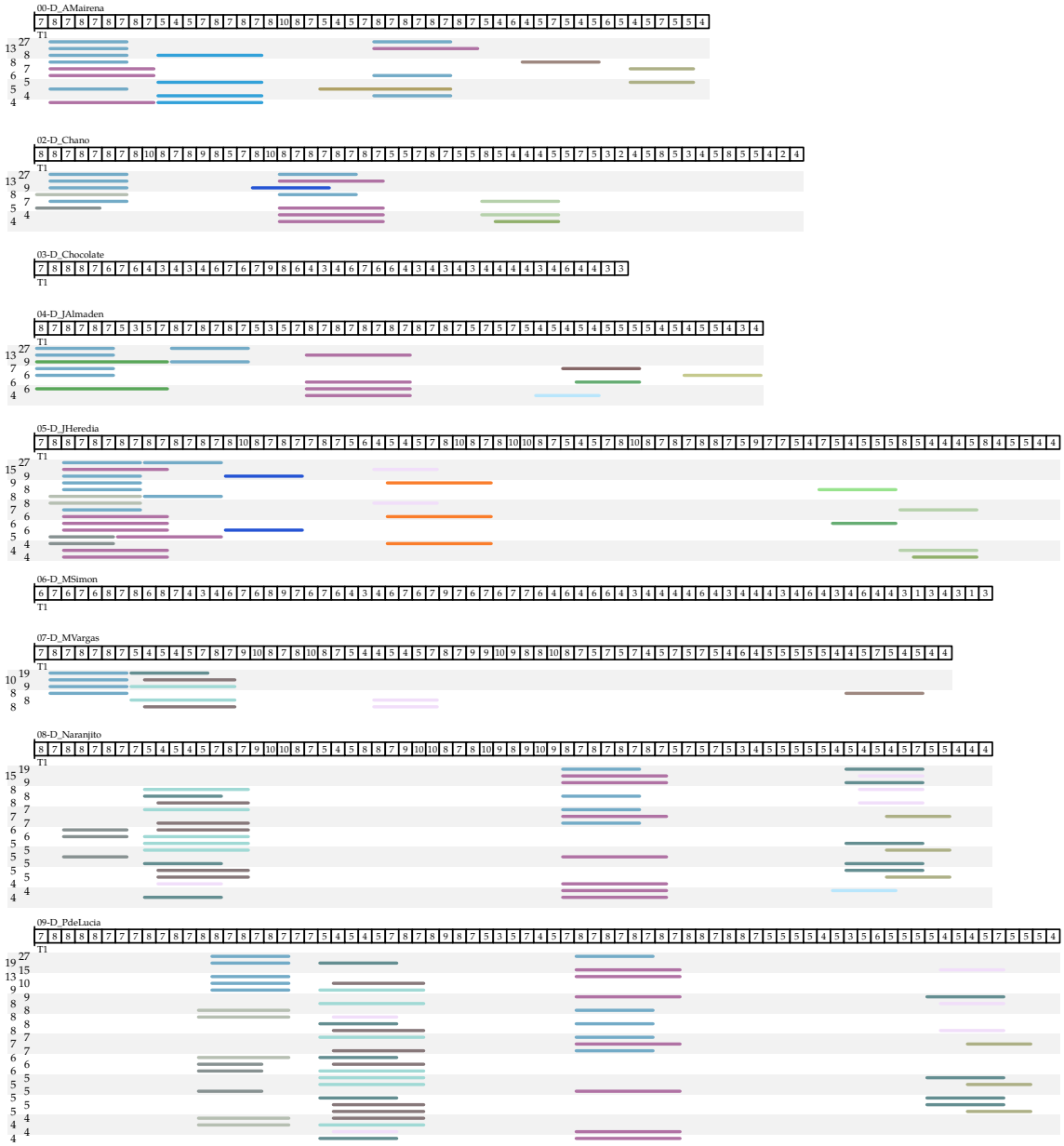


Figura 4.29: Arcos generados con Re-Pair. Filtro de longitud de arco 12, frecuencia mínima 4 y selección orimp

Figura 4.30: Arcos generados con sequitur. Filtro de longitud de arco 12, frecuencia mínima 4 y selección orimp

notación compacta para expresar motivos cuya diferencia es la incorporación de notas al inicio o final de otro motivo. Cuando este caso se ha dado, las notas de diferencia se han marcado entre paréntesis.

Finalmente y para facilitar la identificación de los motivos musicales, tanto en el gráfico de puntos como en los pentagramas se ha usado el mismo código de color que en los gráficos anteriores. Lo que permite, si se quiere, comparar unas formas de representar los arcos con otras.

Sin entrar en un análisis exhaustivo, destacamos las siguientes características musicales encontradas:

- Los rangos de notas de los motivos de las deblas y martinetes 2 son agudos (generalmente superiores al 5, la nota `'la'`) cuando los de los Martinetes 1 son graves (no superan al 5).

- Muchos de los motivos identificados son variaciones entre si. Las técnicas de creación de variaciones encontradas en estas piezas de flamenco son idénticas a las empleada en la música clásica o popular (Vivien, 2010)(Sulzer, Baker, Koch, & Christensen, 1995). Entre otros:

  Alteración de notas El motivo es idéntico salvo un número reducido de notas cuyo valor varía.

  En el gráfico se aprecia esta variación como dos motivos solapados salvo en las notas de la variación.

  Transposición La melodía es transpuesta, esto es: todas las notas varían de frecuencia en la misma proporción.

  Gráficamente se ven dos melodias con el mismo contorno desplazadas verticalmente. En la **figura 4.33** (en los motivos al origen del arco) se aprecia esta técnica.

  Expansión interválica El contorno melódico sufre una expansión o distorsión en el eje (vertical) de las notas.

  En esta variación, el perfil de los motivos presenta máximos y mínimos en las mismas posiciones, pero con valores distintos.

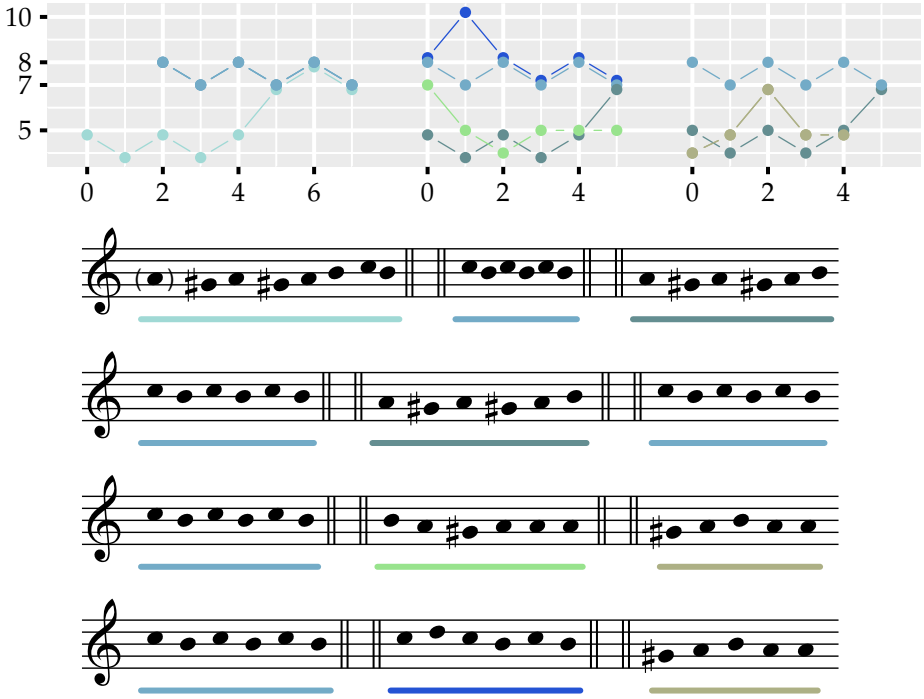


Figura 4.31: Selección de arcos específicos Debla

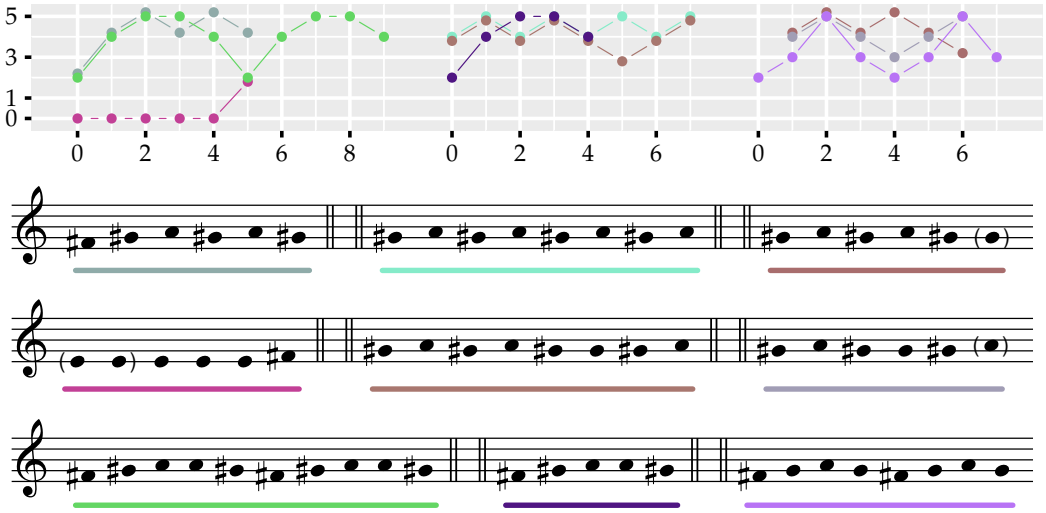


Figura 4.32: Selección de arcos específicos Martinete 1

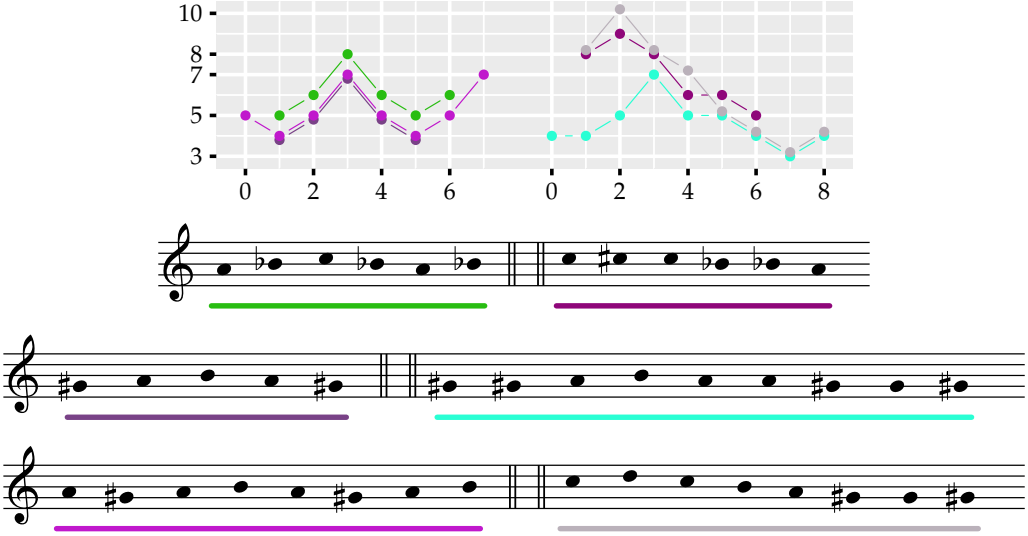


Figura 4.33: Selección de arcos específicos Martinete 2

4.4 Fandangos de la provincia de Huelva

El conjunto de tonás, examinado en la sección anterior, era un conjunto de piezas bastante estudiado en el que no existían dudas sobre estilos o agrupaciones de cada pieza. Este corpus ha sido usado como grupo de control con el objeto de verificar la bondad de las herramientas de análisis desarrolladas.

Una vez demostrado la utilidad de estas, se han ensayado las mismas sobre un nuevo corpus (los fandangos de Huelva) del que no existe una opinión uniforme ni en la literatura ni entre los conocedores del flamenco.

Aunque la estructura general de análisis en los fandangos copia la seguida en las tonás, la falta de una clasificación previa aceptada en los fandangos, la mayor variedad de estos y las diferencias estructurales de los fandangos impiden seguir exactamente los mismos procedimientos que en las tonás. Aun así, las herramientas de análisis propuestas siguen siendo de aplicación y se muestran los resultados obtenidos.

4.4.1 Corpus

El conjunto de fandangos con el que se va a trabajar procede de un trabajo no publicado del Dr. Mora Roche. En este trabajo, se han recopilado 88 fandangos que representan otros tantos estilos de fandangos de Huelva. Dichos fandangos han sido transcritos y posteriormente clasificados teniendo en cuenta criterios estrictamente musicales.

La transcripción se ha efectuado manualmente realizando una reducción en las piezas que eliminan detalles ornamentales específicos de cada intérprete. Las transcripciones resultantes se presentan, en consecuencia, como ejemplos paradigmáticos de cada estilo considerado.

Clasificación Mora Roche

La clasificación posterior se ha realizado considerando únicamente el contenido del primer tercio del fandango y se ha efectuado a dos niveles: cercanía entre piezas (identificada con un número) y subgrupo de gran afinidad (identificada con una letra). El primer nivel se basa en la coincidencia de la nota «tenor»[37] El segundo nivel se basa en la comparación de otras características como la caída final de la melodía o la nota máxima del fragmento.

Dos situaciones especiales se han tenido en cuenta: la posibilidad de que una pieza esté identificada en un grupo, pero sin

[4.37] Terminología tomada del canto gregoriano. La nota tenor, *repercusio* o tono de recitación, es alrededor de la cual se establece la melodía. Es la segunda nota más importante en una melodía tras la tónica (Pujadas, 2016).

cercanía a ningún subgrupo. Este caso está denotado por el número identificador de grupo sin letra. Un segundo caso se encuentra cuando un estilo no se parece a ningún grupo. En dicha circunstancia, se ha optado incluirlo en un grupo «desconocido». El grupo desconocido no implica que todos los fandangos que pertenecen a él sean parecidos entre si; más bien, cada elemento identificado en este grupo es distinto a todos los demás fandangos considerados (sean del grupo desconocido o de cualquier otro). El grupo desconocido se ha denotado como «¿?» y sirve como convención para evitar crear grupos específicos con un único integrante. Denominaremos a esta clasificación, la clasificación de fandangos JMR.

Clasificación geográfica

Es importante destacar que, hasta ahora, la clasificación utilizada por expertos está basada en criterios geográficos (Quiñones Castilla, 2012) y no por características musicales que tanto la clasificación JMR como el presente trabajo propugnan. La clasificación por área geográfica que se muestra se ha simplificado en cuatro grandes grupos: Alosno, Huelva, Resto y Desconocido. Alosno y Huelva son las dos poblaciones con mayor variedad de estilos de fandangos y se merecen un grupo propio cada una.

Hay un conjunto de áreas que definen uno o dos variedades de fandangos en los que termina confundiéndose la población con el estilo (como pueden ser los fandangos de Valverde o Encinasola). Estos estilos se han identificados en el grupo «Resto». Por último, el grupo «Desconocido» incluyen piezas de creación moderna en las que no está claro a qué área asociarla[38].

La **tabla 4.32** identifica los fandangos pertenecientes al corpus utilizado, la región geográfica a la que se le asocia y el grupo y subgrupo sonoro al que pertenecen. Los nombres de los estilos se basan en la región en la que se desarrollan, el autor que lo desarrolló o una combinación de ambos[39].

Fandangos	Área Geográfica	Grupo
Aldeano	Resto	17a
Alosno Angel de Pura	Alosno	5
Alosno Antiguo 1	Alosno	1a
Alosno Antiguo 2	Alosno	7a
Alosno Antiguo 3	Alosno	7a
Alosno Antiguo 4	Alosno	11a

Tabla 4.32: Estilos de fandangos usados, área geográfica y sus agrupaciones JMR

[4.38] La diferencia entre Resto y Desconocido estriba en que en las primeras sí se conoce el área específica a la que se asocia el fandango (aunque no sea relevante para este estudio) y en el segundo no se conoce dicha atribución.

[4.39] Salvo el fandango identificado como de «Dos Hermanas» que indica el lugar donde se registró el cante.

Fandangos	Área Geográfica	Grupo
Alosno Antiguo 5	Alosno	2
Alosno Antiguo 6	Alosno	4a
Alosno Antiguo 7	Alosno	5a
Alosno Antiguo 8	Alosno	2
Alosno Antiguo 9	Alosno	15a
Alosno Antonio Abad	Alosno	17
Alosno Antonio Toscano	Alosno	2
Alosno Bartolo v. A	Alosno	13a
Alosno Bartolo v. B	Alosno	13a
Alosno Bartolo v. Toronjo	Alosno	15a
Alosno Cané 1 alto	Alosno	3b
Alosno Cané 1 bajo	Alosno	3b
Alosno Cané 2	Alosno	2a
Alosno Diego Perrengue	Alosno	3a
Alosno Fernando Camisa	Alosno	13b
Alosno Juan Maria Blanco	Alosno	1a
Alosno Juana Maria	Alosno	17
Alosno Manolillo el Acalmao	Alosno	2a
Alosno Manuel el Colorao	Alosno	17b
Alosno Manuel el Colorao bemol	Alosno	17b
Alosno Marcos Jimenez	Alosno	11
Alosno Marcos Jimenez antiguo	Alosno	11a
Alosno Paco Toronjo	Alosno	¿?
Alosno Pepe Toronjo	Alosno	12b
Alosno Perez de la Matea 1	Alosno	13a
Alosno Perez de la Matea 2	Alosno	3
Alosno Valiente 1	Alosno	7b
Alosno Valiente 2	Alosno	7b
Alosno Valiente 3	Alosno	1
Alosno Valiente antiguo	Alosno	4a
Cabezas Rubias	Resto	6
Calañas	Resto	10

Tabla 4.32: Estilos de fandangos usados, área geográfica y sus agrupaciones JMR

Fandangos	Área Geográfica	Grupo
Calañas antiguo	Huelva	6
Cortegana	Huelva	14a
Corto de S. Eulalia	Resto	10a
Corto de S. Eulalia Toronjo	Resto	10a
Cruz de la Fuente	Resto	17
Cruz del Hoyo	Resto	11a
Cruz del Llano	Resto	11a
Dos Hermanas	Huelva	2a
El Cerro	Resto	8
El Cerro Quintos	Resto	7
Encinasola	Resto	¿?
Encinasola del pandero	Resto	2
Herrerito	Huelva	5a
Huelva 1	Huelva	14b
Huelva 10	Huelva	14b
Huelva 11	Huelva	16
Huelva 12	Huelva	5a
Huelva 13	Huelva	17a
Huelva 14	Huelva	7b
Huelva 2	Huelva	12b
Huelva 3	Huelva	14a
Huelva 4	Huelva	2a
Huelva 4 v.2	Huelva	2a
Huelva 5	Huelva	2
Huelva 6	Huelva	2
Huelva 7	Huelva	3a
Huelva 8	Huelva	1a
Huelva 9	Huelva	1
Huelva Base de Pepe la Nora	Huelva	5a
Huelva Cané 1	Huelva	3a
Huelva Cané 2	Huelva	3a
Huelva Corto	Huelva	17

Tabla 4.32: Estilos de fandangos usados, área geográfica y sus agrupaciones JMR

Fandangos	Área Geográfica	Grupo
Huelva Valiente 1	Huelva	7b
Huelva Valiente 2	Huelva	12a
Largo de Sta. Eulalia	Resto	11
Noche de los pinos	Resto	8a
Paymogo	Alosno	13b
Paymogo-Alosno	Alosno	13b
Pepe la Nora	Huelva	5a
Pepe Sanz	Huelva	7b
Peque de la Isla	Huelva	1a
Repicao de Las Veredas	Resto	8a
Riotinto	Desconocido	16
Rojita	Huelva	12a
Santa Barbara	Huelva	12b
Tharsis	Desconocido	¿?
Valverde	Resto	9a
Valverde Gatillo	Resto	9a
Zalamea 1	Resto	7a
Zalamea 2	Desconocido	¿?

Tabla 4.32: Estilos de fandangos usados, área geográfica y sus agrupaciones JMR

En el total de 88 estilos considerados, se han identificado 17 grupos perceptivos (divididos en un total de 32 subgrupos) y han quedando 4 estilos sin ser asignados a ningún grupo.

4.4.2 Agrupación de fandangos

El proceso de división de los fandangos en grupos se efectuará de forma distinta al de las tonás por dos motivos:

– La clasificación de tonás es incuestionable. Todos los expertos están de acuerdo con la categoría a la que pertenece cada pieza. En contra, la clasificación presentada de los fandangos es un estudio que no ha sido consensuado.

  Esta clasificación no debe, por tanto, utilizarse como un patrón indiscutible.

– El corpus de tonás estaba compuesto de 72 piezas divididas en 3 categorías, por lo que en media significa que cada categoría está representada por 24 tonás. En el caso de los fandangos, si descontamos las piezas no clasificadas tenemos una media inferior a 4 piezas por categoría.

Estos motivos hacen que cambie el enfoque en el análisis de grupos. El análisis de las tonás era un sistema de agrupación informado en el que se sabía de antemano el número de grupos en el que había que ubicar cada pieza. Con los fandangos en cambio, se trata de un sistema en el que no hay información a priori del número de categorías a considerar y por lo tanto, requiere de una estimación del número de grupos más apropiados para la distribución dada. que salen.

De forma análoga a como se trabajó con las tonás, se ha construido una matriz de distancias empleando DEMC + CSPA (**figura 4.34**) entre los distintos fandangos analizados. Sobre dicha matriz, se ha efectuado una agrupación jerárquica aditiva empleando *complete-linkage* como algoritmo de cálculo de la distancia inter-grupo.

El dendograma que representa la agrupación jerárquica (en la **figura 4.35**) permite estimar el número óptimo de clusters (k) asociado a los datos. De entre todos los criterios estimación de k ensayados («frey» (Frey & Van Groenewoud, 1972), «McClain» (McClain & Rao, 1975), «c-index» (Hubert & Levin, 1976), «silhouette» (Rousseeuw, 1987) y «dunn» (Dunn, 1974))[40], todos sugieren el empleo de 20 categorías salvo frey (que sugiere $k = 9$) y mcclain ($k = 10$). Es por ello que se ha terminado escogiendo $k = 20$ categorías de fandangos.

[4.40] Todos ellos calculados con el paquete Nb-Clust (Charrad, Ghazzali, Boiteau, & Niknafs, 2014)

Este dendograma ya se muestra con los identificadores de las piezas coloreadas en función del grupo al que se le ha asignado. Además, para facilitar la visualización de las relaciones entre piezas, se ha construido un filograma (mostrado en la **figura 4.36**) en el que además de usar el color para reflejar el cluster asignado a cada pieza, se ha añadido al lado del identificador de la pieza una etiqueta con la categoría de la clasificación JMR.

A continuación se listan los grupos identificados[41] y las piezas que los componen.

[4.41] Los números de identificación se han asignado de forma arbitraria usando el orden en el que los grupos han aparecido en el dendograma.

Cluster 1 Alosno Cane 1 alto, Alosno Cane 1 bajo y Alosno Paco Toronjo

Cluster 2 Alosno Juana Maria, Alosno Manuel el Colorao, Alosno Manuel el Colorao bemol, Cruz de la Fuente, El Cerro, Huelva 9, Huelva Cane 1 y Zalamea 2

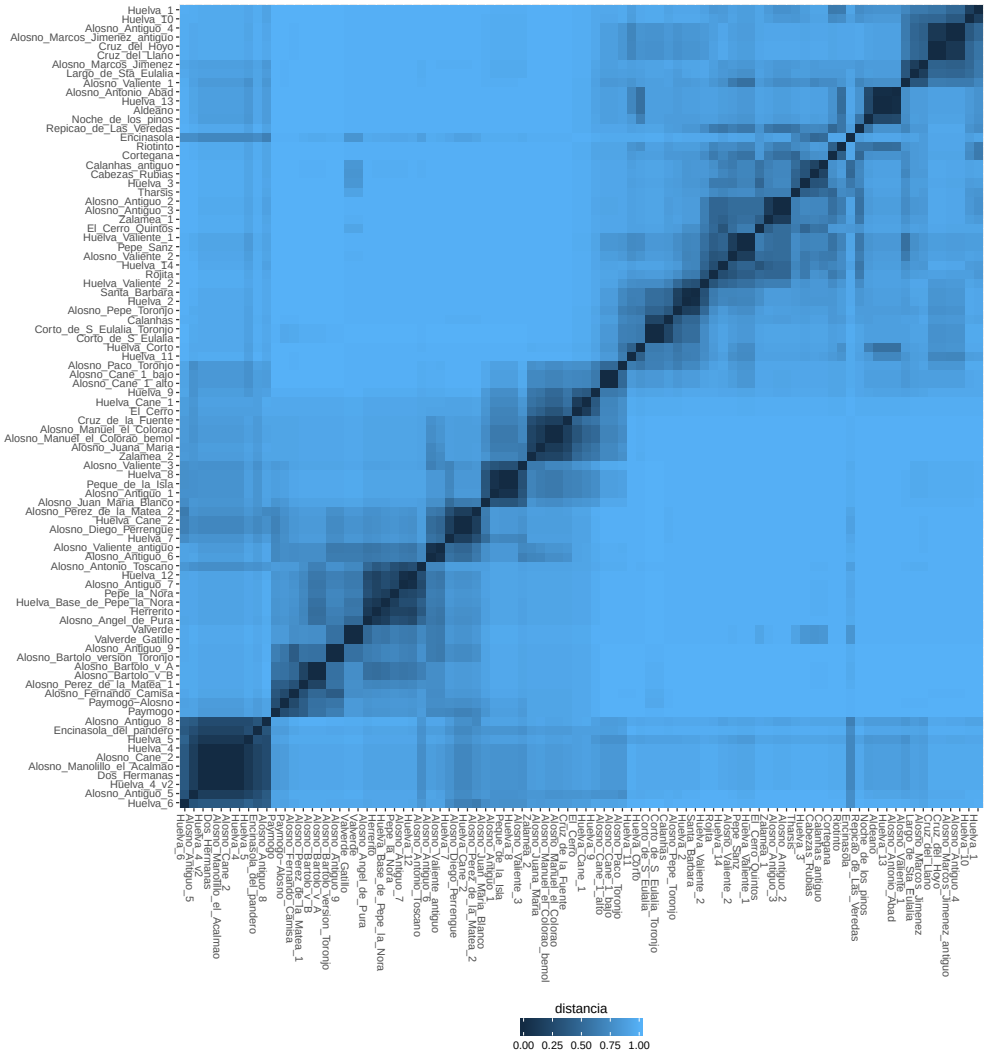


Figura 4.34: Matriz de distancias de fandangos empleando DEMC + CSPA

Cluster 3 Alosno Pepe Toronjo, Huelva 2, Huelva Valiente 2 y Santa Barbara

Cluster 4 Calanhas, Corto de S Eulalia y Corto de S Eulalia Toronjo

Cluster 5 Huelva 11 y Huelva Corto

Cluster 6 Alosno Valiente 2, Huelva 14, Huelva Valiente 1, Pepe Sanz y Rojita

Cluster 7 Alosno Antiguo 2, Alosno Antiguo 3, El Cerro Quintos y Zalamea 1

Figura 4.35: Dendograma de fandangos DEMC+CSPA

Cluster 8 Aldeano, Alosno Antonio Abad, Huelva 13 y Noche de los pinos

Cluster 9 Cortegana, Repicao de Las Veredas y Riotinto

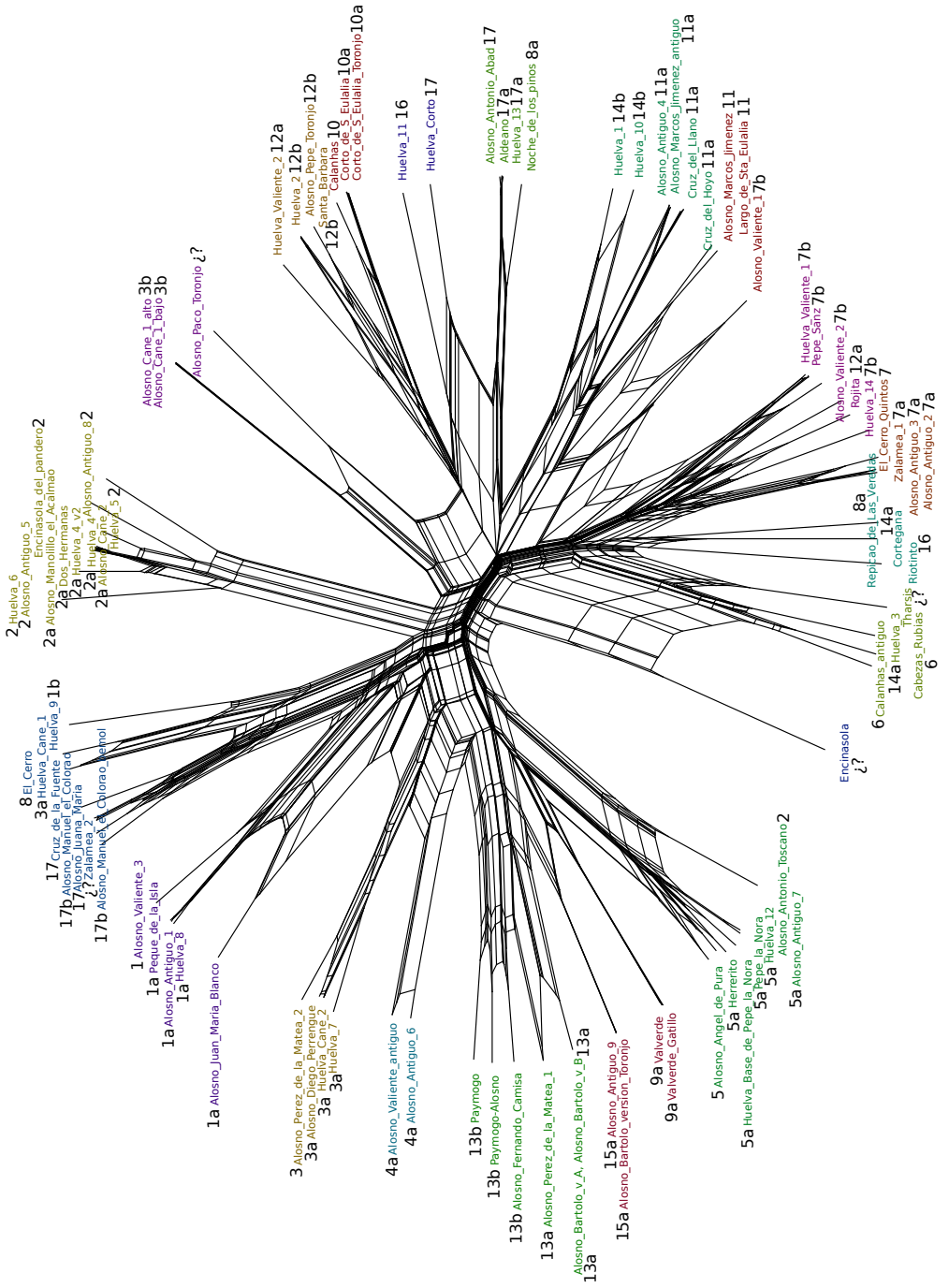


Figura 4.36: Árbol filogenético de fandangos DEMC + CSPA

Cluster 10 Alosno Marcos Jimenez, Alosno Valiente 1 y Largo de Sta Eulalia

Cluster 11 Alosno Antiguo 4, Alosno Marcos Jimenez antiguo, Cruz del Hoyo, Cruz del Llano, Huelva 1 y Huelva 10

Cluster 12 Encinasola

Cluster 13 Cabezas Rubias, Calanhas antiguo, Huelva 3 y Tharsis

Cluster 14 Alosno Bartolo v A, Alosno Bartolo v B, Alosno Fernando Camisa, Alosno Perez de la Matea 1, Paymogo-Alosno y Paymogo

Cluster 15 Alosno Angel de Pura, Alosno Antiguo 7, Alosno Antonio Toscano, Herrerito, Huelva 12, Huelva Base de Pepe la Nora y Pepe la Nora

Cluster 16 Alosno Antiguo 9, Alosno Bartolo version Toronjo, Valverde y Valverde Gatillo

Cluster 17 Alosno Antiguo 5, Alosno Antiguo 8, Alosno Cane 2, Alosno Manolillo el Acalmao, Dos Hermanas, Encinasola del pandero, Huelva 4, Huelva 4 v2, Huelva 5 y Huelva 6

Cluster 18 Alosno Antiguo 1, Alosno Juan Maria Blanco, Alosno Valiente 3, Huelva 8 y Peque de la Isla

Cluster 19 Alosno Antiguo 6 y Alosno Valiente antiguo

Cluster 20 Alosno Diego Perrengue, Alosno Perez de la Matea 2, Huelva 7 y Huelva Cane 2

Antes de evaluar esta agrupación, comparándola con la clasificación JMR, es importante recordar las siguientes consideraciones:

Respecto al contenido del árbol, es importante recordar las siguientes consideraciones:

– La propuesta JMR se basa en una herramienta de análisis que requiere conocimientos específicos difíciles de adquirir. Dado que no existen otras herramientas similares de análisis (exceptuando la que se propone en este trabajo), los resultados que proporciona están pendientes de validación.

  Aun así, al no existir otros estudios de clasificación de fandangos por criterios musicales (si existen por otros criterios como los históricos o los geográficos), se utilizará ésta clasificación como elemento de comparación.

– Los grupos propuestos en este trabajo son independientes de aquella propuesta y no son fruto de ninguna técnica de entrenamiento supervisado que intente imitar esta clasificación; sino de la definición de una métrica apropiada a este tipo de información. La propuesta de este trabajo es independiente de cualquier otra clasificación previa (incluida la de Mora Roche) y sólo depende de los datos (las notas) de las piezas analizadas.

Al no tener disponible una clasificación canónica, no tiene sentido hablar de errores de clasificación tal y como se hizo cuando se hablaba de las tonás. Sin embargo, aun es posible determinar índices de correlación entre las dos clasificaciones mostradas que nos indiquen si los grupos propuestos por cada método son o no coherentes entre sí.

Para ello, se ha calculado el índice V de Cramér (ϕ_c) [42]. En la **tabla 4.33** se muestra el valor del índice y el valor de p de la X^2 asociada. El valor asociado indica una fuerte correlación entre las agrupaciones. Dado que ambas han sido obtenido de forma independiente, el resultado apoya mutuamente las técnicas comparadas.

[42] El índice V de Cramér proporciona una medida de relación entre variables nominales. En la práctica, es una X^2 de Pearson reescalada para tener valores entre 0 y 1. (Cramér, 2016)

k	ϕ_c	p value
20	0.892	< 0.0005

Tabla 4.33: Índice ϕ_c: JMR vs DEMC+CSPA

4.4.3 *Extracción de descriptores diferenciadores*

Al igual que se hizo con las tonás, se han identificado los descriptores capaces discriminar a cada categorías considerada. Nuevamente, el número de descriptores capaces de discriminar los grupos es tan numeroso que se ha preferido dejar el listado completo en un anexo[43] y dejar en esta sección la discusión de los detalles más relevantes.

[43] Un listado completo de los descriptores puede verse en el apéndice en la **sección C.2**.

En función de los resultados obtenidos, vamos a dividir los descriptores con capacidad de diferenciar un grupo en tres categorías: variaciones de `PClass2_T`, de `PClass1_T` y el resto de descriptores.

Variaciones sobre PClass2_T

Todos los clusters analizados presentan al menos un descriptor derivado de `PClass2_T` que tiene valores específicos para dicho cluster. Estos descriptores indican los intervalos musicales de aparición más frecuentes en un rango de notas dada y en un orden de búsqueda determinado.

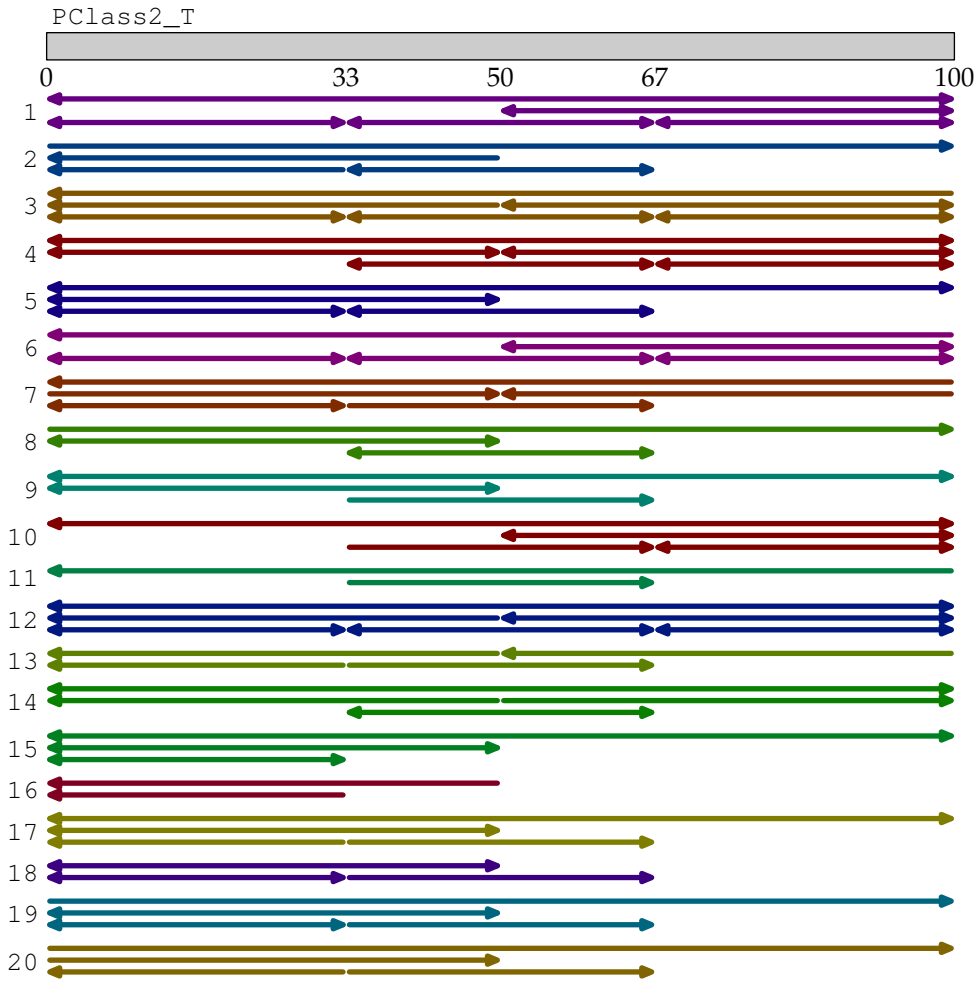


Figura 4.37: Rangos de discriminación de descriptores derivados de `PClass2_T`

La **figura 4.37** muestra un resumen de estas variaciones. Cada flecha representa una variación de `PClass2_T` en el que el color y el número a la izquierda de la flecha representa el grupo que es capaz de identificar, la posición y longitud de la flecha el rango de las notas consideradas y la punta de flecha el orden de búsqueda usado. Examinando el gráfico puede observarse como el cluster 16 puede ser identificado por los descriptores `PClass2_T_rev_0_a_50` y `PClass2_T_rev_0_a_33`, esto es: los intervalos más frecuentes en la primera mitad y el primer tercio de la pieza dando más prioridad a los que aparezcan al final de dichas regiones.

Como puede verse por la distribución de las flechas, cada cluster tiene zonas diferenciales situadas en regiones distintas a los

demás. Por ejemplo, el cluster 16 tiene los valores más característicos en la parte inicial del *tercio* cuando el cluster 10 los tiene, sobre todo, en la parte final del mismo.

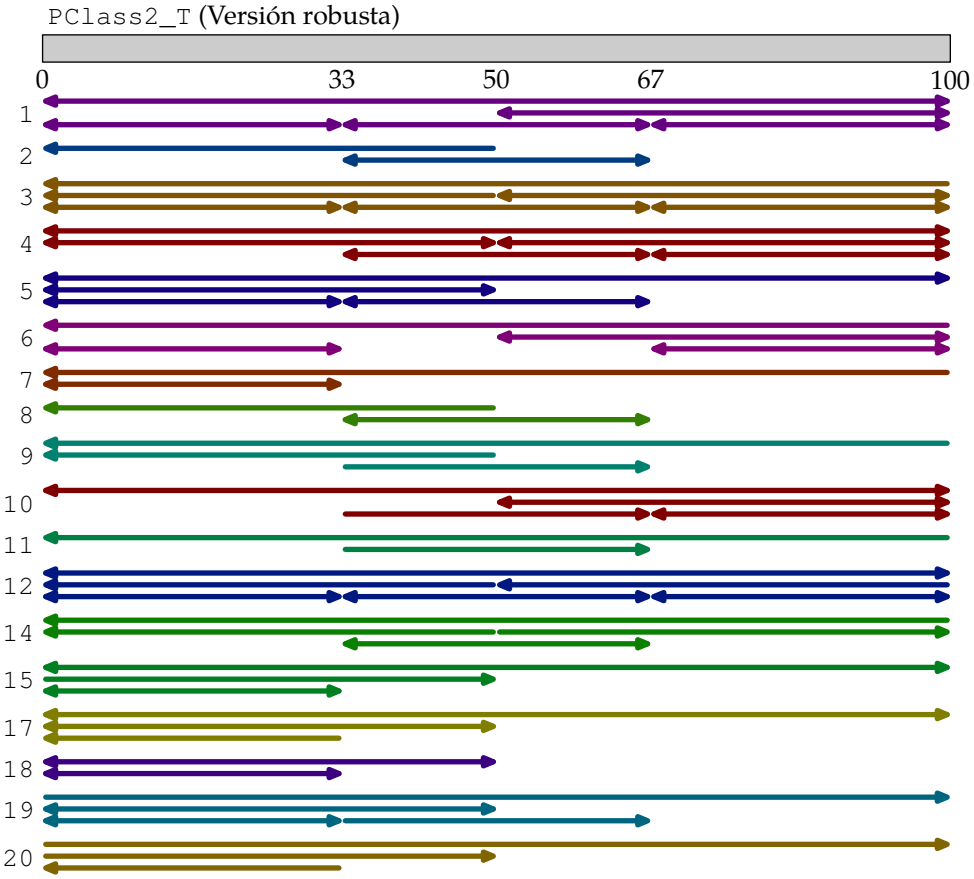


Figura 4.38: Rangos de discriminación de descriptores robustos derivados de `PClass2`

El descriptor `PClass2_T` tiene su versión robusta en la que los intervalos se suministran sin referencia a cuál intervalo es el más frecuente, cuál el segundo y cuál el tercero más frecuente. La **figura 4.38** muestra los descriptores en versión robusta de `PClass2_T`. En esta ocasión, los grupos `13` y `16` no pueden ser identificados correctamente usando esta versión robusta del descriptor.

Es interesante destacar que, en contra de lo comúnmente aceptado en los estudios musicales (como por ejemplo (Purwins, 2005)) en los que el *Pitch Class* se considera exclusivamente para composiciones enteras, el presente trabajo aboga por usar rangos que permitan calcular el *Pitch Class* en zonas específicas de la pieza ya que así lo confirma las regiones útiles para la identificación de las piezas.

Variaciones sobre PClass1_T

El segundo grupo de descriptores es el compuesto por la variaciones del descriptor `PClass1_T` (las notas más frecuentes en una región). Este grupo es muy interesante ya que a las personas es más fácil identificar notas sueltas que intervalos. Lamentablemente, no todos los clusters pueden ser identificados por medio de este descriptor.

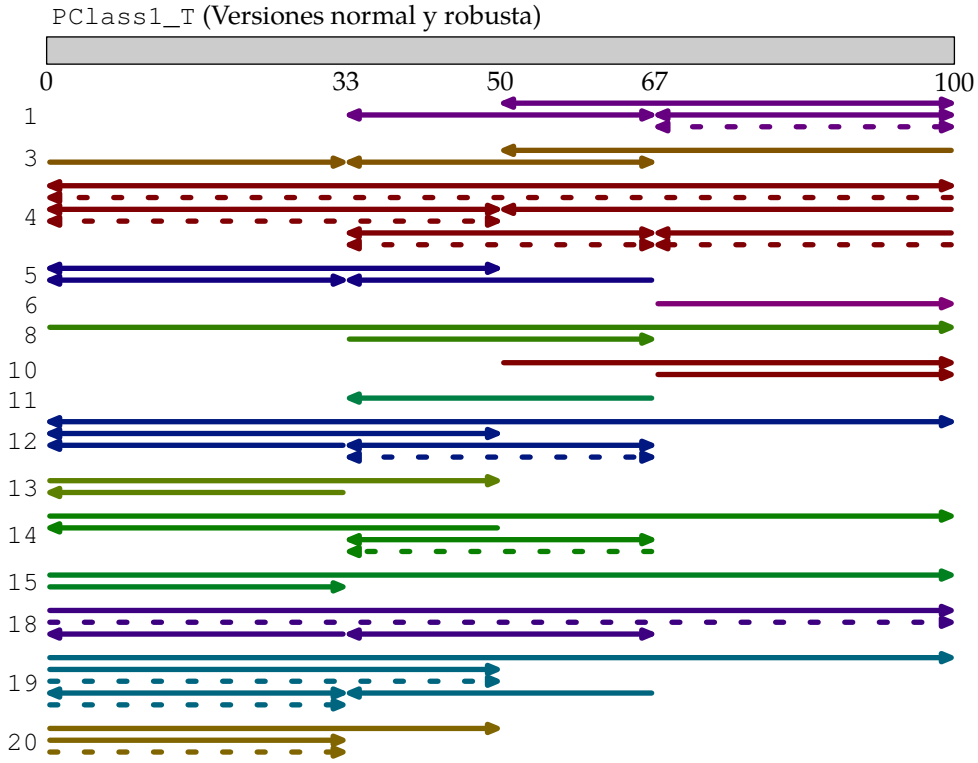


Figura 4.39: Rangos de discriminación de descriptores robustos derivados de `PClass1`

En la **figura 4.39** se muestran las variaciones a partir de `PClass1_T`. En esta ocasión, en el mismo diagrama se muestra los descriptores normales (en flechas de línea contínua) y las versiones robustas de los descriptores en líneas discontinuas. De la figura se aprecia la menor capacidad de clasificación de estos descriptores: hay menos descriptores con capacidad de identificar un grupo y hay menos grupos identificados (concretamente los grupos `2`, `7`, `9`, `16` y `17` no son identificados con estos descriptores).

Otros descriptores

Finalmente, los clusters pueden tener características específicas reflejadas en otros descriptores. Algunos de estos son:

Cluster 1 Presenta una pendiente de regresión característica en el primer (`PendienteRegresion_rango_0_a_33`) y segundo (`PendienteRegresion_rango_33_a_67`) tercio de la pieza.

Cluster 5 Nota media elevada en todo el *tercio* (`MPitch`), la primera mitad (`MPitch_rango_0_a_50`) y el primer tercio (`MPitch_rango_0_a_33`).

Cluster 12 La suavidad en la segunda mitad del *tercio* (`Suavidad_rango_50_a_100` y `Suavidad2_rango_50_a_100`) tiene un valor elevado[44].

[4.44] Un valor elevado equivale a poca suavidad

Cluster 19 Las piezas de este grupo tienen un salto de mi a do (`m_0_a_8_rango_0_a_33`) al inicio del *tercio*.

4.4.4 Extracción de arcos melódicos

Al igual que ocurrió con las tonás, se ha recopilado un diccionario de arcos melódicos que aparecen en los fandangos, y que permiten construir un grafo.

La **figura 4.40** muestra un grafo completo de los arcos específicos de los fandangos donde se aprecian acumulaciones de líneas de color que sugieren agrupaciones. Para favorecer su visualización, también se acompaña de una versión podada[45] del mismo (**figura 4.41**) en el que se puede apreciar mejor las relaciones entre las distintas categorías (identificadas por el color del arco). Dos categorías que comparten un nodo son categorías que comparten submotivos musicales, por lo que cuanto mayor sea la cantidad de nodos frontera comunes, mayor será la relación entre categorías.

[4.45] La poda ha consistido en la eliminación de motivos cortos que presentan solapamiento con otros motivos de tamaño superior.

En esta ocasión, la longitud del primer tercio de los fandangos[46] es bastante más reducida al de las tonas por lo que se aprecian las siguiente diferencias cualitativas respecto a los arcos obtenidos en las tonás:

[4.46] En media, tienen una duración de 8.8 notas frente a la duración media de 38.86 de las tonás.

- La longitud de los motivos identificados (y por tanto de los arcos) es menor que en las tonás ya que la máxima longitud de todo motivo está limitado por la longitud del tercio.
- La especificidad de los arcos encontrados es menor. Al ser los arcos más cortos, es más probable encontrarlos en fandangos de cualquier categoría[47].
- El número posible de arcos también es menor. Tal y como se calculó en la **expresión 3.25**, el número de arcos posibles es

[4.47] Es mucho menos específica la cadena «en» (debido a su longitud) que la cadena «en un lugar de la mancha» que, lógicamente, aporta más información. La primera cadena es, por tanto, más fácil de encontrar en cualquier tipo de texto que la segunda.

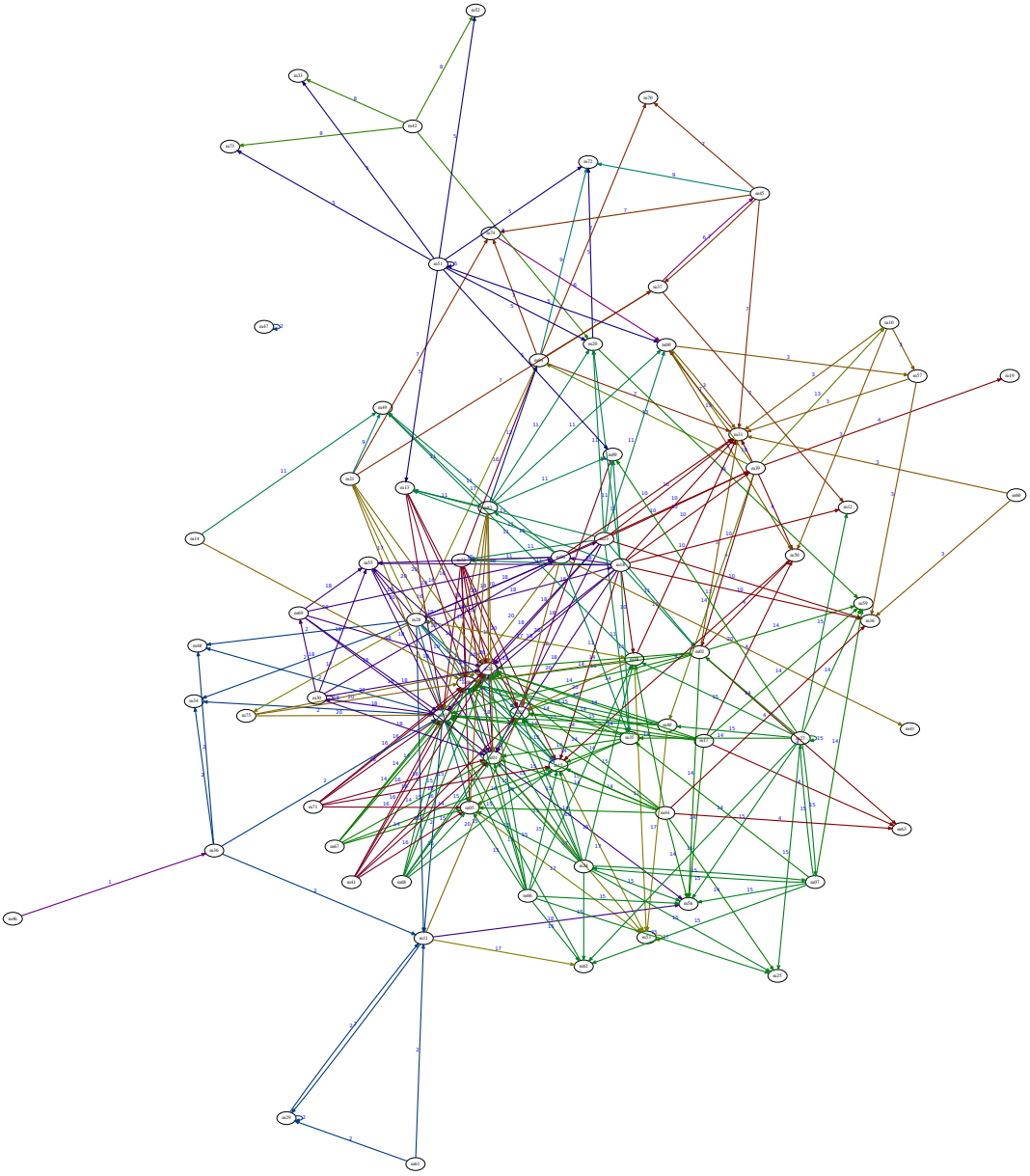


Figura 4.40: Grafo de fandangos

proporcional a la cuarta potencia de la longitud de la cadena examinada.

La reducida cardinalidad de los arcos, permite trabajar con todos ellos sin necesidad de establecer filtros de frecuencia o longitud mínima.

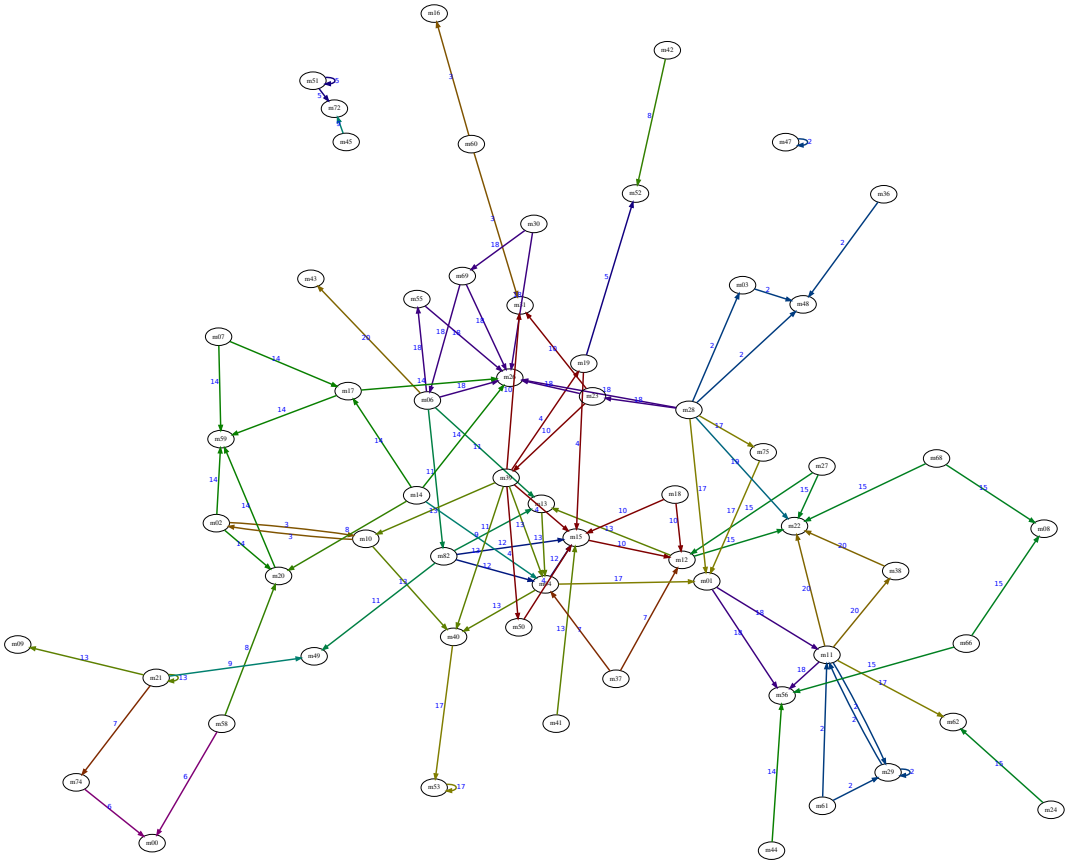


Figura 4.41: Grafo de fandangos. Podado que filtra los motivos más cortos en caso de solapamiento

A modo de ejemplo se muestran los fandangos del cluster `17`. En la **figura 4.42** se muestran todos los arcos melódicos identificados en la categoría. Si ahora consideramos los arcos específicos de la categoría (aquellos que solo están presentes en ésta) nos queda el grupo tal y como se representa en la **figura 4.43**. Tal y como se enunciaba, los tercios más pequeños contienen arcos cortos que presentan una mayor probabilidad de coincidir con arcos en otras categorías, por lo que estas piezas no incorporan ningún arco específico.

La escasez de arcos de cierta longitud también dificulta la identificación de partes con función más estructural de las que no lo son. En los fandangos que tienen el primer tercio más largo (como son Alosno Antiguo 8, Encinasola del pandero o Huelva 6, sí que permite estimar dichas partes [48] con mayor facilidad. Aun así, las arcadas que aparecían en las tonás ya no son reconocibles en los fandangos.

4.48 Quizás sea más preciso decir que se puede observar notas que no están nunca en ningún motivo y de ahí deducir las zonas estructurales.

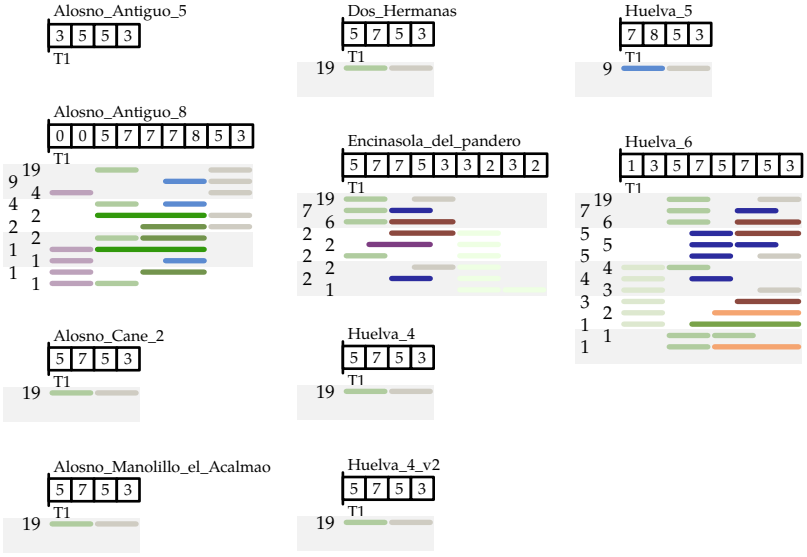


Figura 4.42: Arcos melódicos del grupo `17`

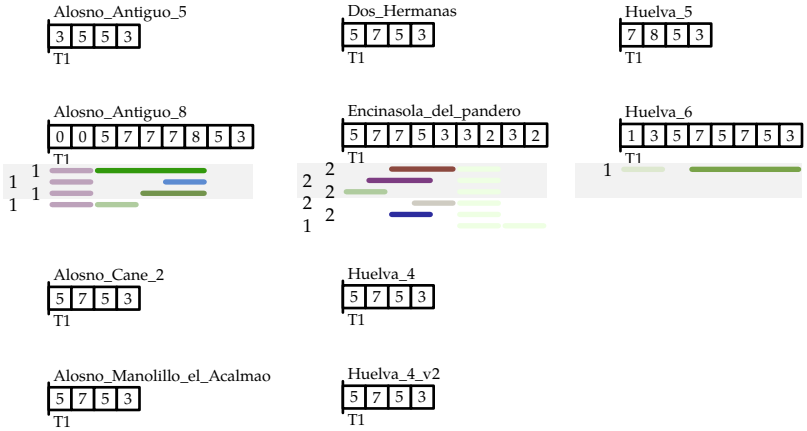


Figura 4.43: Arcos melódicos específicos del grupo `17`

En contra de este grupo, el cluster `18` (**figura 4.44**) está formado por fandangos con el primer tercio más largo por lo que todos los fandangos de este grupo presentan numerosos arcos específicos de esta categoría (**figura 4.45**).

El hecho de que los fandangos fueron transcritos eliminando aspectos ornamentales hace que las transcripciones sean de naturaleza fundamentalmente estructural lo que motiva que los arcos identificados tengan con asiduidad una distancia nula entre los motivos que la forman. Aun así, siguen existiendo límites (quizás en la pieza Huelva 8 sea donde se aprecie más claramente) que permiten identificar regiones relevantes dentro del tercio.

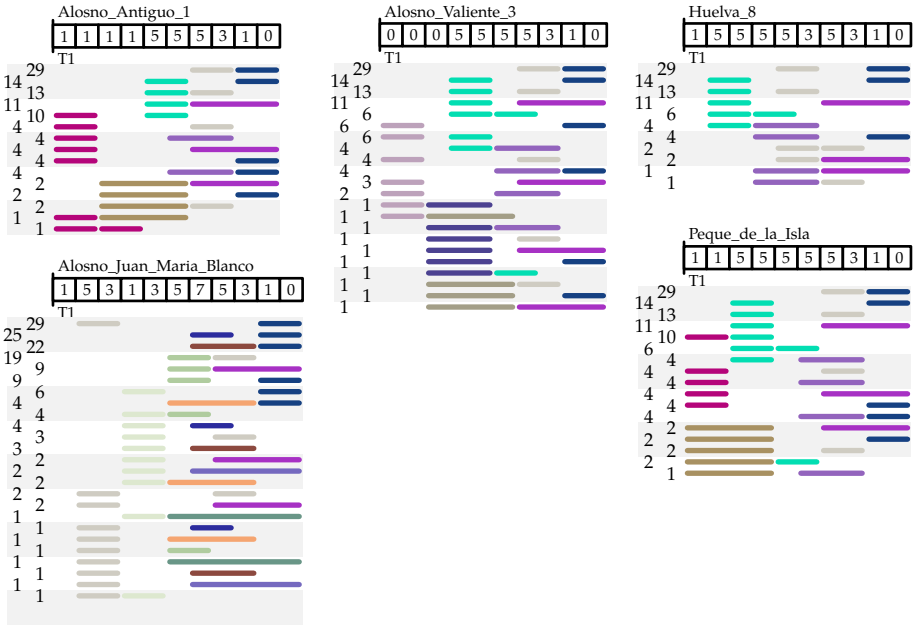


Figura 4.44: Arcos melódicos del grupo `18`

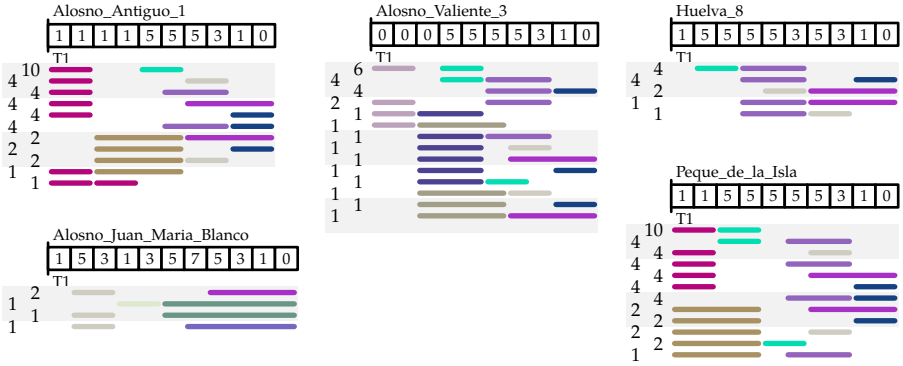


Figura 4.45: Arcos melódicos específicos del grupo `18`

Uso de arcos como herramienta de valoración de una clasificación.

Los arcos permiten valorar la idoneidad de una clasificación por criterios melódicos. El corpus ideal de clasificación sería aquél en el que cada grupo fuera absolutamente independiente (melódicamente hablando) de los demás. En cuyo caso, ningún arco aparecería simultáneamente en dos o más categorías[49]. Melódicamente hablando, las categorías serían absolutamente disjuntas. Así, es posible estimar la independencia musical entre categorías analizando la distribución de arcos comunes en dos o más de estas.

Cuanta mayor relación exista entre las categorías, mayor número de arcos aparecerán comunes entre las categorías. De ahí que podemos afirmar que una buena clasificación es aquella que

[4.49] Por economía de lenguaje, utilizaremos a partir de ahora el término «arcos en común» para indicar arcos presentes simultáneamente en dos o más categorías.

mantiene juntos elementos relacionados y separados a los que no lo son, por lo que debe minimizar el número de arcos comunes.

Conviene recordar que la identificación y etiquetado de los arcos melódicos en categorías es una operación de suma cero. Los arcos se obtienen por un procedimiento independiente y posteriormente son etiquetados en categorías. Si se contabilizan los arcos comunes, el resto de los arcos estará asignado exclusivamente en una categoría específica. Este hecho nos permite establecer un primer criterio de calidad de clasificaciones usando arcos:

Criterio de cantidad de arcos comunes Ante dos clasificaciones efectuadas sobre el mismo corpus, aquella que presente menos arcos en común será una mejor clasificación.

Por otro lado, ya se ha mencionado la relación entre la longitud de los arcos y su especificidad. Cuanto más largo es un arco, más información incorpora y su presencia mejor define una supuesta categoría. Lo que nos lleva a un segundo criterio de calidad de clasificaciones:

Criterio de la longitud de arcos comunes Ante dos clasificaciones efectuadas sobre el mismo corpus, aquella que presente los arcos en común más cortos será una mejor clasificación.

Ambos criterios de calidad pueden verificarse a partir de un histograma de los arcos comunes en función de la longitud de los mismos. El área del histograma corresponde con la cantidad total de arcos comunes y la longitud media (o la máxima) dar una estimación para el criterio de longitud.

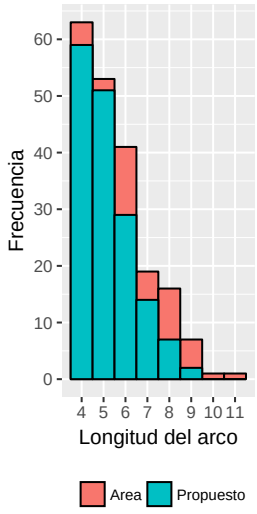


Figura 4.46: Histograma de longitudes de arcos comunes. Comparación clasificación fandangos

La **figura 4.46** muestra el histograma de los arcos comunes en función de la longitud para la clasificación tradicional por área geográfica (en rojo) frente a la clasificación melódica propuesta (en verde). Cuando se emplea la división geográfica, el número de arcos comunes (el área del histograma) es 201 frente a los 162 de la clasificación propuesta. Si consideramos las longitudes de los arcos, nuevamente la clasificación territorial se comporta peor que la propuesta, sin importar si se considera como índice la longitud máxima de cada distribución (11 *vs* 9) o el valor medio (5.512 *vs* 5.167) de las mismas.

La aplicación de los dos criterios es unánime: el desempeño de la clasificación propuestas es superior[50] al de la clasificación clásica territorial.

4.50 Siempre hablando desde el punto de vista del análisis de melodías.

De los arcos comunes identificados (de la clasificación propuesta), los más largos son de 9 elementos (3+6) en la que algunos de estos elementos pueden desaparecer. La **figura 4.47** muestra una familia de estos arcos comunes. La primera parte del arco (o motivo origen) está formado por conjuntos de tres notas que en ocasiones la primera nota puede o no estar presente. El segundo motivo (el destino del arco) es en todos estos casos el mismo: una expresión descendentes terminando en la, así llamada, «cadencia andaluza» (`['5','3','1','0']`o, con usando los nombres de nota que representan: `['la','sol','fa','mi']`). Al igual que pasa con el motivo inicial del arco, la primera nota de la cadencia presentada puede aparecer o no dando una formación total de 5 o 6 notas.

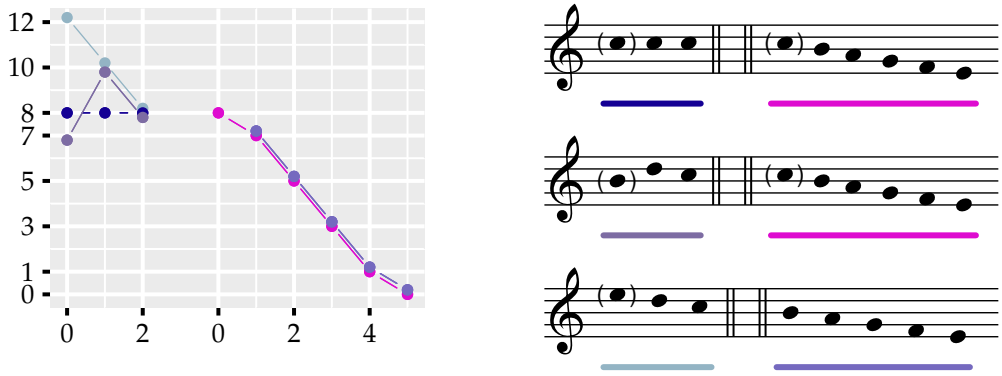


Figura 4.47: Familia de arcos en común de fandangos

Una segunda familia de arcos comunes que incluye términos más largos (hasta 9 elementos) se muestra en la **figura 4.48**.

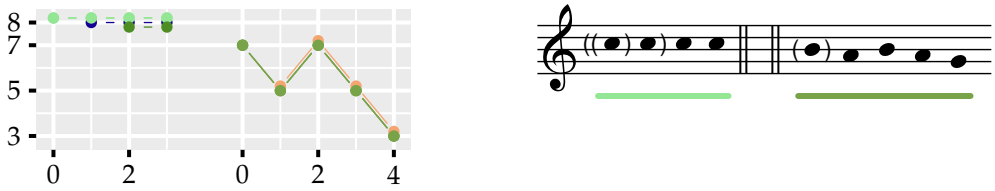


Figura 4.48: Familia de arcos en común de fandangos (bis)

Análogamente a lo realizado con los arcos comunes, es posible representar los arcos específicos a cada categoría identificadas. Estas categorías pueden compartir contornos melódicos de algún motivo musical (especialmente en los motivos que incluyen la cadencia final del tercio); pero el conjunto del arco (la suma de motivo origen y el motivo destino) siempre es único para dicha categoría. Lo que implica que no hay dos gráficos de arcos (arcos melódicos, al fin y al cabo) que coincidan entre distintas figuras.

Los arcos mostrados representan a los arcos más largos, y por tanto los más selectivos, dentro de cada categoría. Para un listado completo, debe consultarse la tabla situada en el **anexo C.3**.

Cluster 1

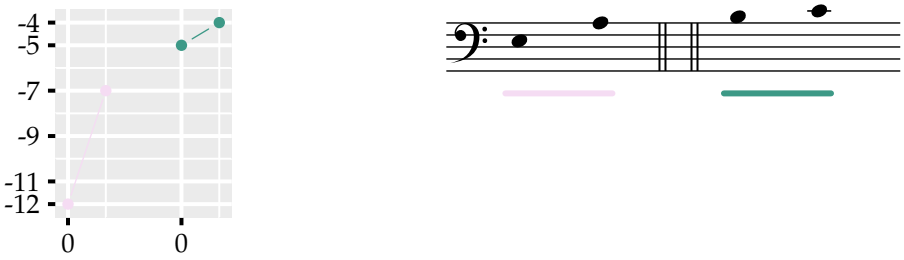


Figura 4.49: Selección de arcos específicos cluster 1

Cluster 2

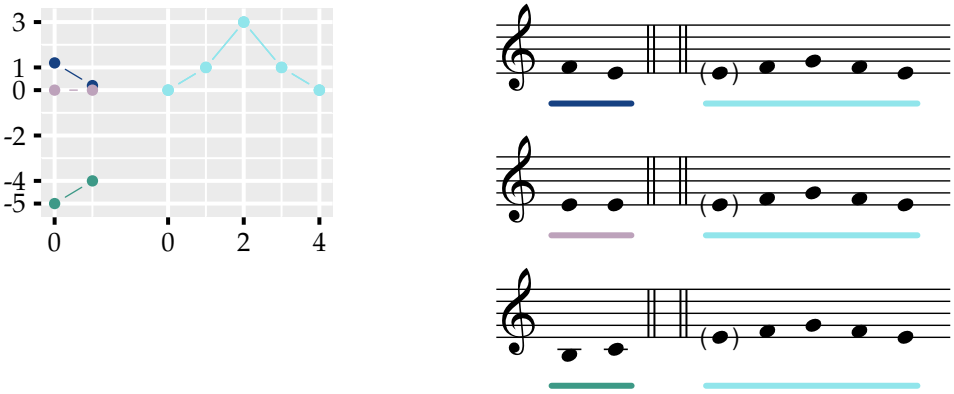


Figura 4.50: Selección de arcos específicos cluster 2

Cluster 3

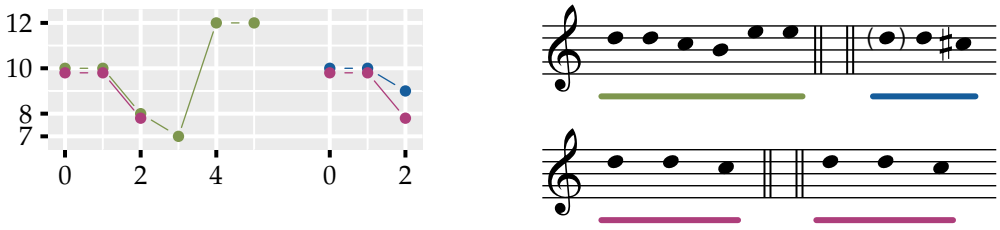


Figura 4.51: Selección de arcos específicos cluster 3

Cluster 4

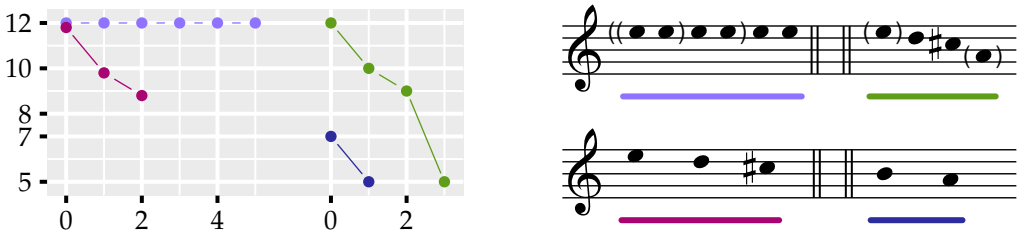


Figura 4.52: Selección de arcos específicos cluster 4

Cluster 5

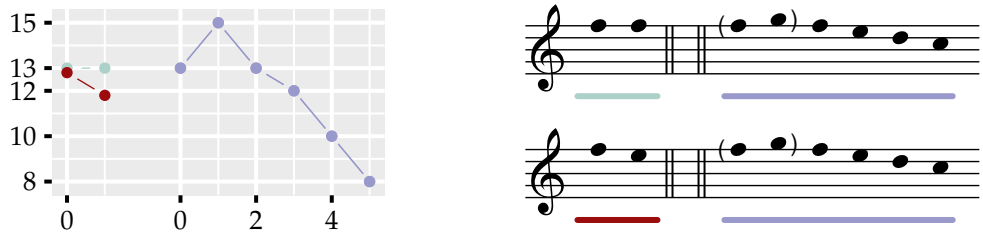


Figura 4.53: Selección de arcos específicos cluster 5

Cluster 6

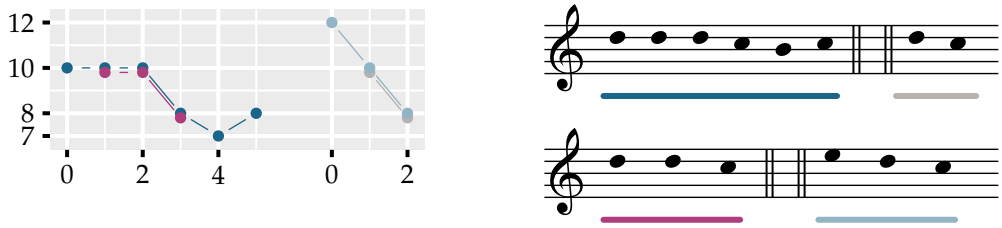


Figura 4.54: Selección de arcos específicos cluster 6

Cluster 7

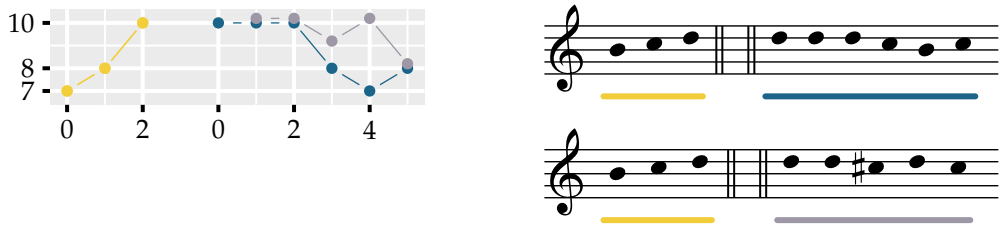


Figura 4.55: Selección de arcos específicos cluster 7

Cluster 8

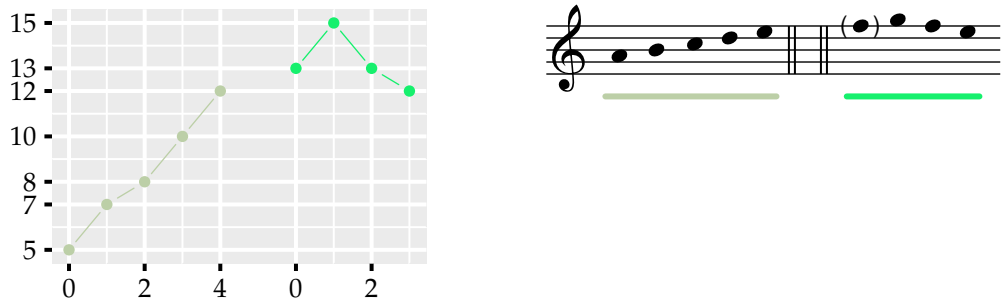


Figura 4.56: Selección de arcos específicos cluster 8

Cluster 9

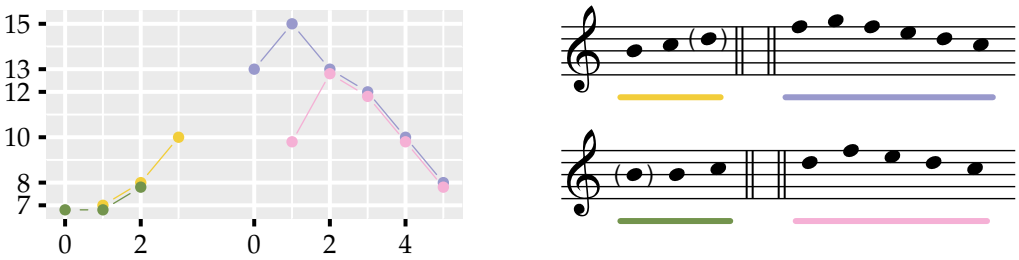


Figura 4.57: Selección de arcos específicos cluster 9

Cluster 10

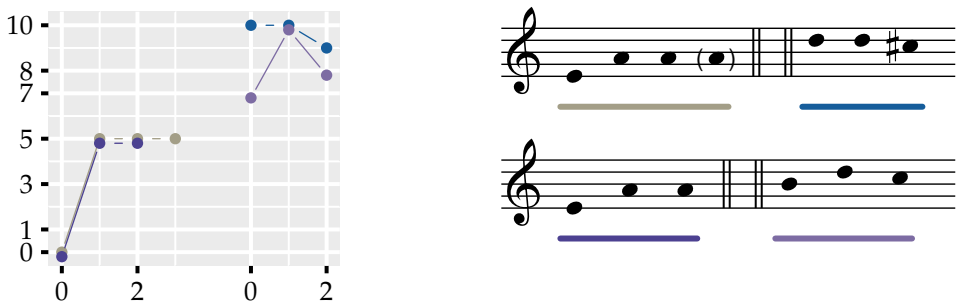


Figura 4.58: Selección de arcos específicos cluster 10

Cluster 11

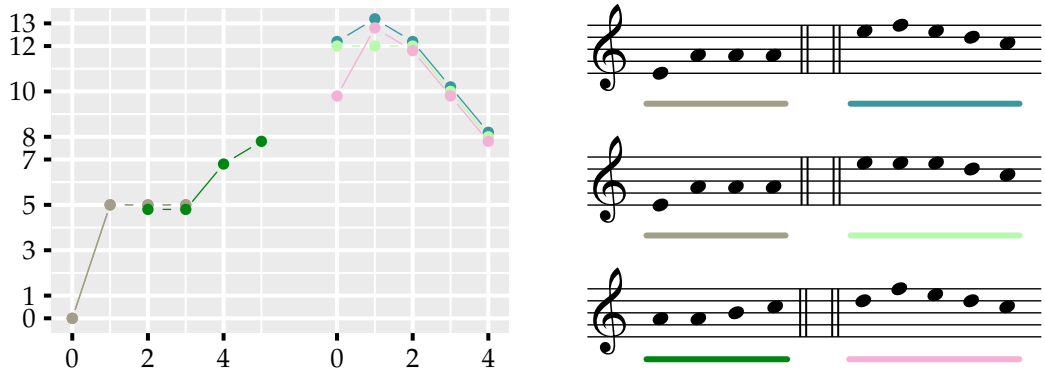


Figura 4.59: Selección de arcos específicos cluster 11

Cluster 12

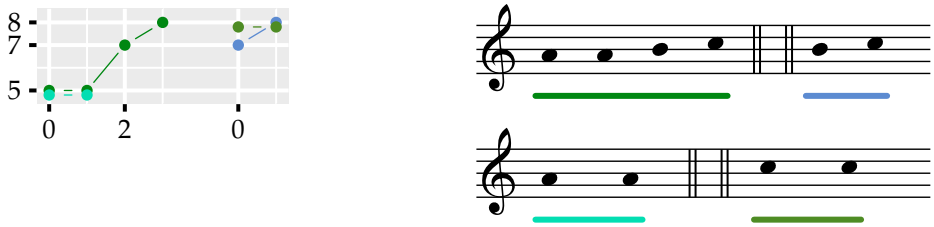


Figura 4.60: Selección de arcos específicos cluster 12

Cluster 13

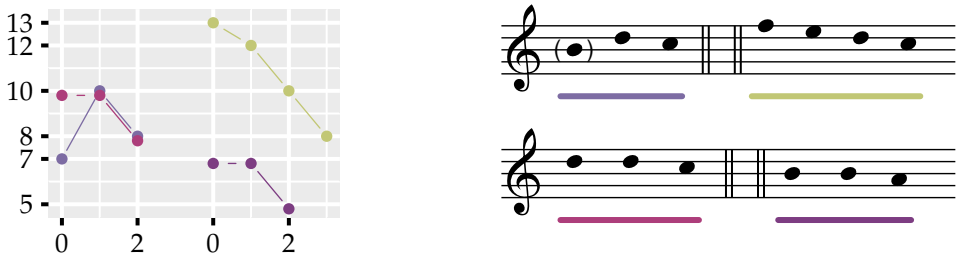


Figura 4.61: Selección de arcos específicos cluster 13

Cluster 14

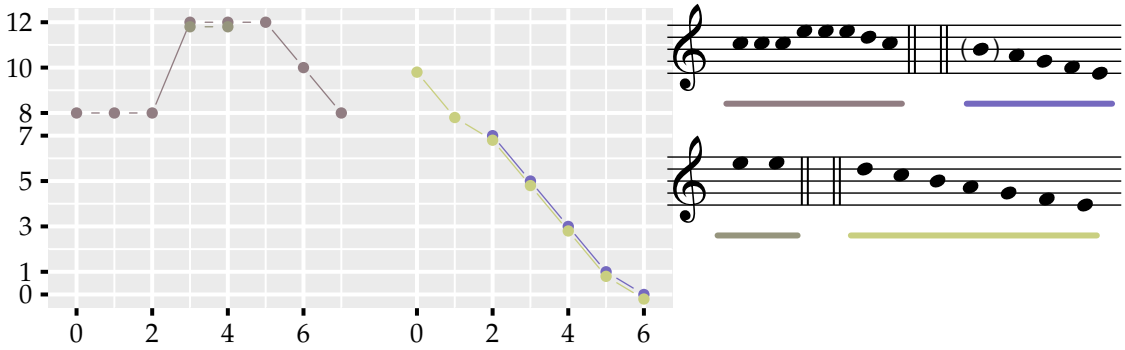


Figura 4.62: Selección de arcos específicos cluster 14

Cluster 15

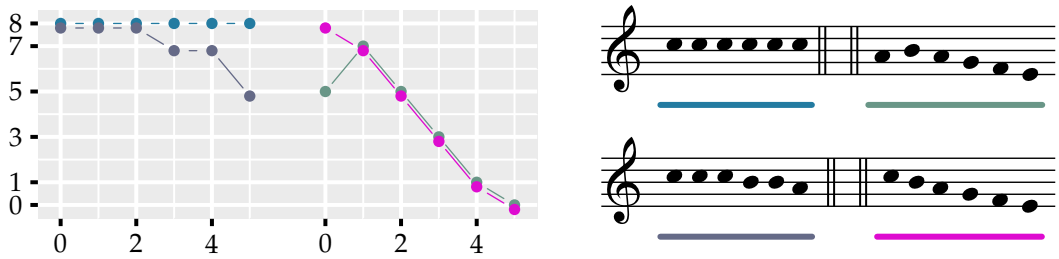


Figura 4.63: Selección de arcos específicos cluster 15

Cluster 16

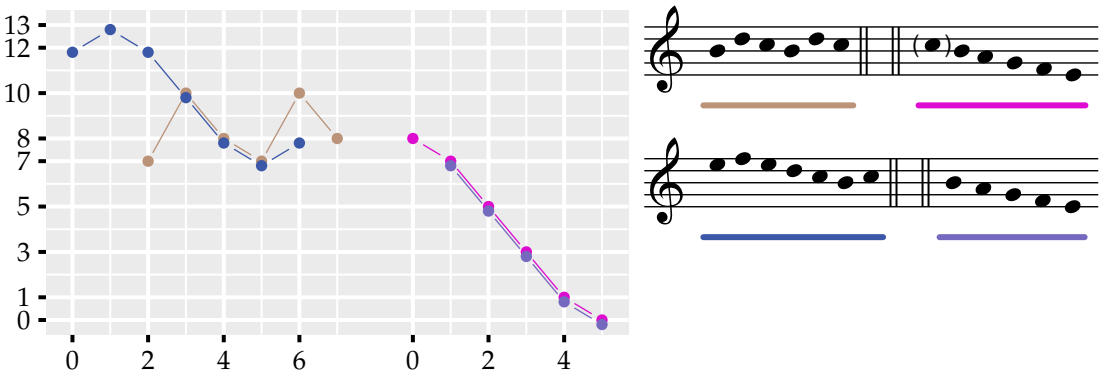


Figura 4.64: Selección de arcos específicos cluster 16

Cluster 17

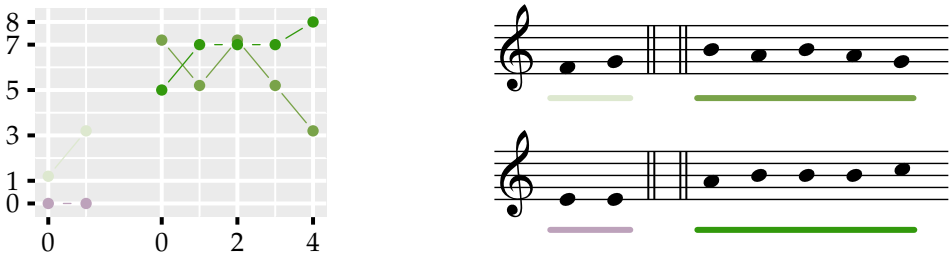


Figura 4.65: Selección de arcos específicos cluster 17

Cluster 18

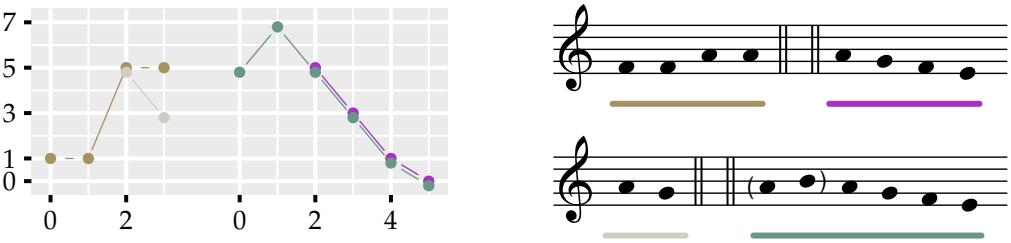


Figura 4.66: Selección de arcos específicos cluster 18

Cluster 19

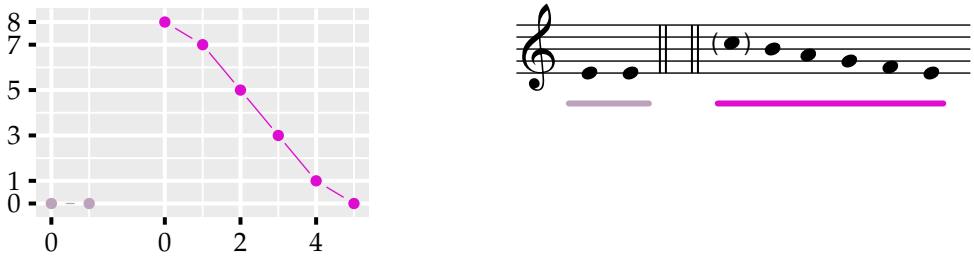


Figura 4.67: Selección de arcos específicos cluster 19

Cluster 20

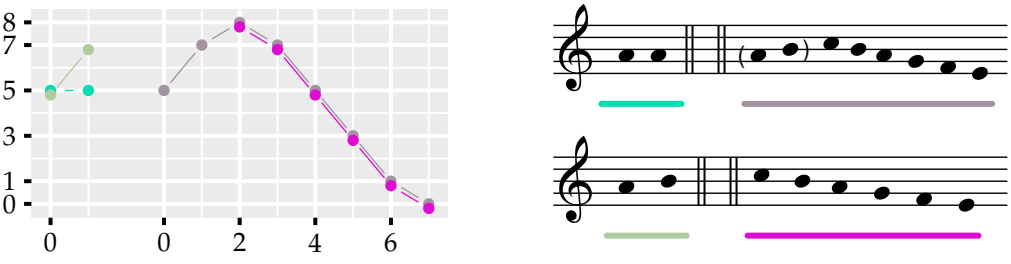


Figura 4.68: Selección de arcos específicos cluster 20

Aunque los arcos melódicos no contienen toda la melodía de las piezas, su construcción se han basado en los elementos comunes de las melodías dentro de una misma clase. A pesar de cierta similitud entre los arcos de distintas categorías, sobretodo ocasionada por el final en cadencia andaluza la, sol, fa, mi (o numéricamente 5, 3, 1, 0), al considerar el conjunto del arco dan motivos musicales claramente diferenciados.

La combinación de las herramientas propuestas efectúan una clasificación por criterios de similitud musical y, posteriormente, identifican los fragmentos relevantes que justifican dicha similitud. Esta nueva organización en grupos permite explorar la naturaleza musical de los fandangos en futuros trabajos.

A partir de este tipo de herramientas puede llegarse a una clasificación de los fandangos de Huelva a partir de un criterio independiente del geográfico, que era el empleado por la flamencología clásica. Entendemos que nuestro trabajo ha posibilitado un análisis musicológico inexistente hasta el momento de muchas posibilidades futuras y de aplicación a otros palos flamencos y en general otros estilos musicales.

5 Conclusiones y futuras líneas de trabajo

Tras el desarrollo teórico y la, posterior, aplicación práctica en cantes flamencos, este capítulo presenta un resumen de las actividades realizadas, un listado de las aportaciones originales efectuadas a lo largo de la tesis y, finalmente, enumera posibles líneas de investigación para continuar trabajando con las herramientas propuestas.

5.1 Resumen

El cante flamenco es un conjunto de estilos cuyas reglas de formación difieren enormemente de la música occidental (ya sea popular o clásica). Su carácter improvisatorio, su ausencia de ritmo o la aplicación de éste con cierta flexibilidad y su modo tonal hacen que se resista a ser analizado con las herramientas empleadas en la música convencional.

Al principio de la memoria se establecieron dos hipótesis relacionadas con el cante flamenco: que los distintos estilos o palos proceden de unas formas originarias o modelos actualmente desconocidos y que los procesos de derivación desde dichos modelos a las formas actuales se producen, fundamentalmente, en zonas localizadas.

A partir de dichas hipótesis y solo de estas hipótesis, se han construido una serie de herramientas robustas para el análisis de melodías flamencas. El carácter general de estas herramientas, debido a la no especificidad de las mismas hacia los cantes ensayados, permiten su uso en otros palos flamencos o incluso en otras áreas de conocimiento a los que se le puedan aplicar las hipótesis planteadas.

El objeto de las herramientas desarrolladas buscaba responder a tres cuestiones sobre los cantes analizados.

¿Qué piezas derivan de un mismo modelo?

Para responder a esta pregunta se ha desarrollado una nueva métrica que hace hincapié en las partes comunes a conjuntos de piezas (las más probables a pertenecer al modelo original).

Esta métrica ha sido ensayada sobre dos conjuntos de cantes: las tonás y los fandangos de Huelva. El primer conjunto es un conjunto de validación que es conocido y está presente en la literatura de la etnomusicología computacional. El análisis de este caso, daba un F_1 mínimo agregado de 0.986, siendo este significativamente superior al resultado de referencia tomado de la literatura (cuyo valor agregado máximo[1] es de 0.833).

5.1 Se ha tomado el peor resultado de F_1 agregado de la métrica propuesta y se ha comparado con el mejor resultado agregado de la métrica de referencia. Por lo que la diferencia mostrada de 0.153 es la diferencia mínima entre las F_1 agregadas.

El mismo procedimiento fue aplicado a los fandangos de Huelva. En este caso, dado que para este corpus no existe un consenso de clasificación melódica, no tiene sentido hablar del indicador de calidad F_1. En cambio, si es posible comparar la clasificación propuesta con otras clasificaciones estimando el nivel de independencia entre las clasificaciones. Para ello se tomó la clasificación (no publicada) del Dr. Mora Roche obtenida siguiendo criterios exclusivamente musicales.

La V de Cramér calculada entre ambas clasificaciones (0.892) muestra una fuerte correlación entre ambas clasificaciones. La fuerte correlación entre ambas implica una validación mutua ya que han sido calculadas de forma independiente empleando herramientas de naturaleza distinta. Este resultado es importante ya que la clasificación aquí propuesta emplea métodos estrictamente computacionales (que son fácilmente reproducibles) frente a la clasificación Mora Roche que requiere de conocimientos musicales y de entrenamiento para poder aplicarla con éxito.

¿Qué caracteriza a las piezas que derivan de un modelo dado?

Y, por ende, cuáles son las características del modelo desconocido.

La existencia de un gran repertorio de herramienta capaces de caracterizar las piezas provenientes de un modelo o estilo dado, hizo que el trabajo desarrollado se centrase en proveer a estas herramientas de un conjunto de descriptores suficientemente rico[2].

5.2 La precisión de estos algoritmos depende, en gran medida, del número de descriptores independientes disponibles.

Para obtener un conjunto de descriptores relevante se ha procedido a trabajar en dos fases: la primera se ha realizado una recopilación de descriptores genéricos aplicables a cadenas y una

segunda de construcción de variantes sobre estos. En total, por cada pieza se han calculado 4650 descriptores.

Si consideramos el corpus de tonás, el descriptor `PClass_2_T`[3] fue capaz de discriminar entre las deblas, los martinetes 1 y 2.

5.3 Los intervalos más frecuentes en el rango considerado.

En el caso de los fandangos de Huelva, el descriptor `PClass_2_T` sigue siendo el descriptor más relevante para distinguir entre las 20 categorías identificadas; pero los fandangos tienen formas más ricas que hacen que algunas categorías sean identificables por el inicio del fragmento analizado, mientras que otras lo sean por la parte media o final de este.

¿Qué partes de las cadenas son más probables que procedan del modelo?

La búsqueda de subcadenas relevantes, considerando que estas son las más probables de los modelos originales, ha derivado en el concepto de arcos melódicos (una construcción de motivos que incorpora en su interior un hueco para las expansiones). Los arcos melódicos son un concepto novedoso fruto de esta investigación.

Los arcos melódicos han resaltado estructuras profundas en las tonás en los que claramente se han apreciado las zonas de expansión intercaladas con submotivos comunes entre piezas.

En el conjunto de los fandangos no se aprecia con tanta claridad las zonas de expansión, presumiblemente debido a una menor longitud de las melodías del primer tercio de los fandangos. Sin embargo, siguen existiendo intervalos que funcionan como unión de submotivos y así se ha resaltado en su momento.

Los arcos abren la posibilidad de usar técnicas de análisis semántico y de textos en el análisis musical, ya que los motivos que forman un arco funcionan como palabras que aportan sentido musical a la pieza y los arcos funcionan como colocaciones lexicográficas de estas palabras.

Consideraciones finales

Aunque el conjunto de herramientas propuestas se han ensayado con éxito en el análisis de piezas de flamenco, no contienen en su desarrollo ninguna consideración especial hacia el flamenco ni hacia la música. Es, por tanto, de esperar que dichas técnicas funcionen sobre otros campos de trabajo.

5.2 *Aportaciones*

A continuación se describen las aportaciones originales realizadas a lo largo de la tesis. Primeramente se describirán las específicas a los algoritmos propuestos

- **División del proceso de aprendizaje entre adecuación de los datos y aplicación del algoritmo.** Donde la adecuación son las operaciones que hay que efectuar a los datos de un tipo determinado y la aplicación es el algoritmo de aprendizaje en sí.
- **Creación de la meta-distancia media al centroide en cadenas** que determina una medida de dispersión de un grupo de cadenas y que se basa en otra métrica subyacente.
- **Sistematización de descriptores para cadenas** en función de los requisitos exigibles a las cadenas analizadas: genéricos, diferenciales y específicos. Y un **listado de descriptores aplicables**.
- **Formulación de mecanismos de construcción de variantes sobre descriptores** como la prioridad de elección de valores o el rango de aplicación.
- **Definición de arco** como elemento de unión de motivos remotos.
- Desarrollo de un lenguaje gráfico para **visualización compacta de subcadenas**. Estos gráficos han permitido, entre otros, analizar el centroide calculado de varias cadenas o localizar la situación de los arcos.
- **Construcción de un criterio de comparación de clasificaciones empleando arcos** que valora como mejor clasificación aquella que presenta menores elementos comunes entre categorías.

Por otra parte, a continuación se detallan las aportaciones efectuadas en el campo del análisis del cante flamenco.

- Se propone un **sistema de agrupación de cantes flamencos por similitud melódica** empleando DEMC + CSPA que se ha demostrado eficaz tanto para las tonás como para los fandangos.
- Igualmente, se ha demostrado la **utilidad del descriptor `PClass_2_T` y sus variantes** para identificar estilos en cantes flamencos.

Además, se ha **diseñado un gráfico de interpretación de valores de `PClass_2_T`**.

- En el análisis del flamenco **es necesario calcular los descriptores por rangos**, para detectar estilos con inicios característicos o finales característicos.

- **Construcción de un diccionario de motivos flamencos** en las tonás y fandangos. Este diccionario permite la realización de análisis de textos aplicado a la música.

- Se ha **identificado una estructura profunda** en las tonas.

5.3 Futuras líneas de trabajo

A lo largo del trabajo realizado en la presente tesis doctoral, no se han explorado todos los caminos que han ido surgiendo. A consecuencia de esto, es posible continuar el trabajo aquí desarrollado profundizando en algunas de estas vías. Entre otras, se considera interesante trabajar en las siguientes tareas:

- Probar la distancia media al centroide usando nuevas métricas. La distancia media al centroide se ha evaluado para dos de las métricas de cadenas más usadas (distancia de edición y n-gramas). Sin embargo existen otras métricas aplicables que no han sido consideradas, como las métricas basadas en alineamiento.

  Estas métricas, más sofisticadas, permiten incorporar conocimiento a la medida de la distancia.

- Optimización de algoritmos. El algoritmo de búsqueda de centroide usado es muy ineficiente, por lo que se requiere de otras técnicas de cálculo del mismo.

- Ampliación del corpus de estudio. El corpus utilizado no deja de ser una muestra reducida de cantes. Una vez probada la efectividad de los algoritmos, es posible ensayarlo en otros tipos de cantes, así como en corpus mixtos (como por ejemplo, ver el efecto de mezclar las tonás con los fandangos de Huelva).

  Un corpus de piezas más numeroso permite construir clasificadores y estimar el rendimiento por medio de análisis de validación.

A parte de estas líneas de continuación, las herramientas presentadas abren, también, nuevos caminos. Se ha insistido en numerosas ocasiones que las técnicas presentadas son genéricas e independientes de las melodías en el flamenco y en el planteamiento modular del binomio adecuación/aplicación. Las siguientes líneas explotan nuevas posibilidades de estudio, basándose en el trabajo aquí desarrollado:

- Uso de las técnicas en campos nuevo de trabajo tales como señales biomédicas, análisis de consumo de *commodities*, o análisis de textos entre otros.

- Investigación de otros algoritmos de clasificación. Una vez demostrada la utilidad de los descriptores propuestos, es posible aplicar otras técnicas de clasificación como las máquinas de vector soporte (SVM *Support Vector Machine*) o técnicas de aprendizaje profundo (*Deep Learning*).

- Igualmente se puede probar la eficacia de la adecuación presentada sobre otros algoritmos de aprendizaje automático como puede ser la regresión o los razonamientos basados en casos.

- Profundizar en el análisis de los arcos empleando técnicas de análisis de textos como el tf-idf (*Term Frecuency - Inverse Document Frecuency*) o el LDA (*Latent Dirichlet Allocation*).

Apéndices

A Sintaxis de descripción de elementos de un algoritmo

A lo largo del texto se ha empleado una descripción simplificada de funciones y algoritmos que indican la relación entre los parametros de entrada que requiere y lo que tras su ejecución se obtiene. El formato empleado para estas descripciones está derivado de la descripción de tipos empleada en el lenguaje *Haskel* (Peterson, et al., 1997) que se ve más adecuada para describir algoritmos que la descripción matemática de funciones que simplemente indican dominio y codominio ($f : \mathbb{R} \rightarrow \mathbb{R}$).

Un «tipo» en Haskell es una descripción del tipo de información que se almacena en una variable. La descripción del tipo se efectúa en una expresión con la siguiente forma:

variable funcion o tipo descrito :: *descripción del tipo*

(TIPO A.1)

donde *descripcion del tipo* puede ser o un tipo básico predefinido (como *Bool*, *Char*, *Int*, *Float*), otro tipo previamente definido ó una combinación de tipos[1].

Los tipos se denotan con identificadores que empiezan en mayúsculas mientras que las variables, las funciones y las variables de tipo[2] se denotan con identificadores que comienzan en minúsculas.

En Haskell, las funciones también pueden ser asignadas a variables en las mismas condiciones que otros tipos de información (esta condición se suele indicar como que las funciones son tipos de primer orden).

Para expresar una función se emplea el operador flecha «→» indicando que toma un valor del tipo situado a la izquierda de la flecha y devuelve un valor de tipo situado a la derecha de la flecha.

A.1 En Haskell existe un mecanismo más de construcción de tipos: los tipos polimórficos o genéricos que se escapan de la mera descripción de tipos y algoritmos que en este trabajo se hace. En los casos en los que sea útil el empleo de un tipo polimórfico se presentará como un operador más de construcción de tipos derivados sin entrar en la verdadera naturaleza de los mismos

A.2 Una variable almacena un dato, mientras que una variable de tipo almacena un tipo. Estas son usadas para expresar tipos indeterminados.

$$funcion :: tipo\_argumento \rightarrow tipo\_salida \quad \text{(TIPO A.2)}$$

El operador flecha es asociativo por la derecha, lo que significa que en una expresión en la que haya más de una flecha, deben interpretarse primero las de la derecha. Así, los tipos

$$f :: a \rightarrow b \rightarrow c \quad \text{(TIPO A.3)}$$

$$f :: a \rightarrow (b \rightarrow c) \quad \text{(TIPO A.4)}$$

son idénticos. En la práctica, a efectos de interpretar tipos, la última flecha que aparezca indica el tipo de salida de la función y todos los demás tipos indicados entre las otras flechas expresan parámetros de entrada. En las funciones descritas en **A.2** y **A.4**, ambas expresan funciones de dos argumentos de entrada (de tipo a el primero y tipo b el segundo) cuyo valor de salida es del tipo c.

Entre los mecanismos de creación de nuevos tipos:

A.3 Esta variación no pertenece a la descripción de Haskell

Rango Un rango[3] es una restricción a los valores posibles de un tipo y su formulación imita las normas del lenguaje matemático: si un extremo está delimitado con corchetes el extremo está incluido en el rango y si está delimitado por paréntesis no está incluido.

$$Decena :: Int \in [0,10) \quad \text{(TIPO A.5)}$$

Para que un rango tenga sentido, el tipo que se restringe debe ser ordenable.

***n*-tupla** Una tupla es una estructura de datos rígida representando el producto de tipos en el que se almacena simultáneamente dos o más datos de los tipos indicados (en el caso del ejemplo **A.6**, dos elementos: el primero de tipo a y el segundo de tipo b, siempre en ese orden).

$$Tupla :: (a,b) \quad \text{(TIPO A.6)}$$

Lista Una lista es una secuencia de elementos de tamaño desconocido (entre 0 e infinito) en el que todos los elementos son del mismo tipo.

$$Lista :: [a] \quad \text{(TIPO A.7)}$$

En ocasiones, será útil forzar un tamaño determinado en la lista[4]. En cuyo caso, se indicará el tamaño de la lista empleando un subíndice a los corchetes:

A.4 Esta variación no pertenece a la descripción de Haskell

$$Lista\_fija :: [a]_n \quad \text{(TIPO A.8)}$$

en el que la lista tendrá un tamaño definido de n elementos.

Opcional El tipo *Opcional* permite, además de almacenar un valor, no almacenar ningún valor.

$$Opcional :: Maybe\ a \qquad \text{(TIPO A.9)}$$

Esta estructura (derivada de la suma de tipos) permite modelar resultados de procesos que no siempre sean calculables. Por ejemplo, si queremos definir un tipo específico para almacenar el logaritmo de números reales:

$$Tipo\_log :: Maybe\ Real \qquad \text{(TIPO A.10)}$$

Si el número real (almacenado en un float) es positivo, al calcular su logaritmo, tendremos un número real. Si el número real es negativo, no estará definido su logaritmo y la variable de salida no tendrá ningún valor asociado. De esta forma, con un sólo un tipo se cubren todas las opciones.

Opción Al igual que *Maybe*, *Either* es una combinación de suma de tipos en el que el valor almacenado es o un valor de tipo a o un valor de tipo b.

$$Opción :: Either\ a\ b \qquad \text{(TIPO A.11)}$$

El objetivo es, nuevamente, almacenar tipos de valores distintos en función del proceso aplicado. Si queremos calcular la raíz cuadrada de un número, un enfoque podría ser:

$$Raíz :: Either\ Real\ Complex \qquad \text{(TIPO A.12)}$$

En el que la raíz cuadrada puede ser o un número Real (si el argumento era un número positivo) o un número Complejo (si el argumento era un número negativo).

En lenguajes tipo C y derivados, existe la instrucción `union` que permite crear tipos derivados que pueden ser, simultáneamente dos o más tipos. Esta es la diferencia entre `union` y *Either*: en el primero en la misma variable están definidos simultáneamente los dos tipos, mientras que en el segundo los tipos son exclusivos o un tipo o el otro.

O dicho de otra forma, con `union` uno puede introducir un valor de un tipo y extraer un valor de otro tipo. Con *Either*, sin embargo, uno solo puede extraer de una variable el valor del mismo tipo que se introdujo en ella.

B `JuceTranscripcion` Manual de Usuario

B.1 Introducción y principios rectores de diseño

El proceso de transcripción de audio con melodía a partitura o a otros formatos procesables es una tarea ardua que requiere conocimientos y habilidades musicales que los sistemas de procesado automático no son capaces de suplir satisfactoriamente[1]. Así, dentro de las tareas desarrolladas a lo largo de este trabajo de tesis, está la de un software (denominado `JuceTranscripcion`) de asistencia a la transcripción y etiquetado de melodías que ha sido usado (en mayor o menor medida) sobre los corpus analizados.

Esta aplicación se ha desarrollado en C++ estándar usando el *framework* multimedia `Juce` (Storer, 2016). Gracias a ello, es posible compilar y ejecutar dicho programa en todas las plataformas en los que `Juce` este soportado[2].

La aplicación `JuceTranscripcion` se encarga de asistir al usuario en el proceso de transcripción de una grabación de audio para obtener un fichero MIDI etiquetado. Actualmente el programa es funcional incluyendo todas necesidades para efectuar la transcripción de un fragmento musical monofónico.

Al igual que `Tony` (Mauch, et al., 2015), este no requiere que su usuario posea conocimientos avanzados de música, sólo tener un cierto sentido musical.

Ambos sistemas de transcripción permiten trabajar con estructuras musicales no occidentales en las que las duraciones de las notas no están fijadas por una nota patrón y que permiten, al transcriptor, seleccionar el nivel de detalle de la transcripción desde el mero esqueleto melódico a partituras más detalladas con adornos y melismas.

Originariamente, se planteó el uso de la aplicación `Tony` para efectuar las transcripciones; pero el interfaz de usuario no era

B.1 El encuentro MIREX (dentro del congreso ISMIR) hace un *ranking* de los mejores algoritmos hasta la fecha y la precisión de los mejores algoritmos ronda el 71% de acierto (ISMIR. The 17th International Society for Music Information Retrieval Conference. New York, 2016).

B.2 iOS, Android, Windows, MaxOS y Linux. Obviamente para que el software tenga utilidad la plataforma deberá permitir el uso de teclado y ratón de dos botones

el más adecuado para procesar un conjunto elevado de piezas. El interfase de `Tony` establece dinámicas de trabajo que requiere alternar entre el uso del teclado y el ratón lo que aumenta el cansancio al trabajar centrando la vista en la pantalla, buscando las combinaciones del teclado y la situación del ratón. `JuceTranscripcion` se ha construido desde el primer momento para evitar la fatiga en su uso y aumentar, por tanto, su eficiencia.

Principios de diseño

Se consideraron los siguientes principios de diseño en la creación de `JuceTranscripcion`:

- Las operaciones habituales de transcripción se efectúa a dos manos: una sobre el teclado y la otra sobre el ratón. La mano del teclado no tiene que moverse[3] por lo que se reduce la fatiga visual (al no ser necesario buscar la posición de la siguiente tecla a pulsar) y en el cuello.

  El programa se maneja con las dos manos interaccionando simultáneamente. La mano derecha siempre sobre el ratón y la izquierda sobre el teclado modificando las acciones del ratón de forma similar a la estenotipia.

- Las interacciones que requieran precisión (como seleccionar el inicio de una nota) se efectuarán haciendo click con el ratón (interacción *point-and-click*). Otras interacciones que no la requieran (como activar la barra de desplazamiento, o variar el nivel de zoom) podrán hacerse arrastrando el ratón (*drag-and-drop*) o usando la rueda.

  Estudios como (Inkpen, 2001; MacKenzie, Sellen, & Buxton, 1991) demuestran la mayor eficiencia de las operaciones de apuntar y click frente a operaciones de arrastre con el ratón.

- En vez de trabajar con un cursor de edición, como la mayoría de los programas, se trabaja con una región delimitada por 2 cursores (llamados cursor derecho e izquierdo). Los cursores se mueven de forma independiente uno del otro y no pueden desactivarse.

  Existen atajos para mover los cursores a posiciones concretas como pueden ser el inicio o final de otras notas ya identificadas en la transcripción o el inicio y final de la grabación.

- Al igual que el programa de edición musical `Finale` (MakeMusic, 2014), la zona de edición tiene áreas calientes que modifican la función ejecutada en función de dónde se hace el click. A diferencia de Finale, las zonas calientes son áreas

B.3 Por ejemplo, al solicitar que se guarde un fichero, se reutiliza automáticamente el nombre del original añadiéndole un *timestamp* ahorrando el tener que escribir o confirmar un nombre.

grandes y visualmente delimitadas, por lo que no se requiere gran precisión a la hora de marcar en ellas.

Hay cuatro tipos de regiones diferenciadas por dos variables: dentro de la región marcada por los cursores o fuera de ella y dentro de una nota o fuera de ella. En la **figura B.3** (del **manual de JuceTranscripcion B**) se muestran las acciones que se ejecutan en función de las zonas donde se hace click y los modificadores de teclado usados en la pulsación del ratón.

– Las acciones se han buscado para que sean lo más regulares posibles[4] y así facilitar su aprendizaje y accionamiento.

– Como puede verse en la **figura B.3**, no existe una función de seleccionar o editar una nota. En caso de querer modificar una nota, hay que borrarla y crear una nota nueva (operaciones que se efectúan rápidamente).

  No hay pérdida de funcionalidad ya que los cursores pueden mantener las características de la nota borrada haciendo la creación de una nueva nota, con alguna característica modificada, inmediata.

– Estos cambios requieren de un entrenamiento específico debido a las diferencias del interfase respecto a lo habitual.

  Tras el entrenamiento, que no suele ser muy largo, la transcripción con el programa se efectúa muy rápidamente y con poco esfuerzo.

[4] Siempre que sean acciones de movimiento de los cursores el botón izquierdo moverá el cursor izquierdo y el derecho el derecho. Otro ejemplo, si marcar sobre una nota con mayúsculas mueve el cursor izquierdo al inicio de la nota, hacerlo fuera de una nota mueve el cursor izquierdo al inicio de la pieza.

B.2 Descripción de los componentes del programa

B.2.1 Identificación visual de los componentes del programa

La **figura B.1** muestra la pantalla de trabajo de `JuceTranscripcion`. La partes del programa (citadas de arriba a abajo) son:

Barra de menú Desde los menús, se pueden acceder a acciones de relacionadas con los ficheros de trabajo, Vistas y Configuración. Estas opciones se describirán con mayor detalle en los siguientes apartados. En los casos que aparezca una combinación de teclas al lado de una opción, esta representará un

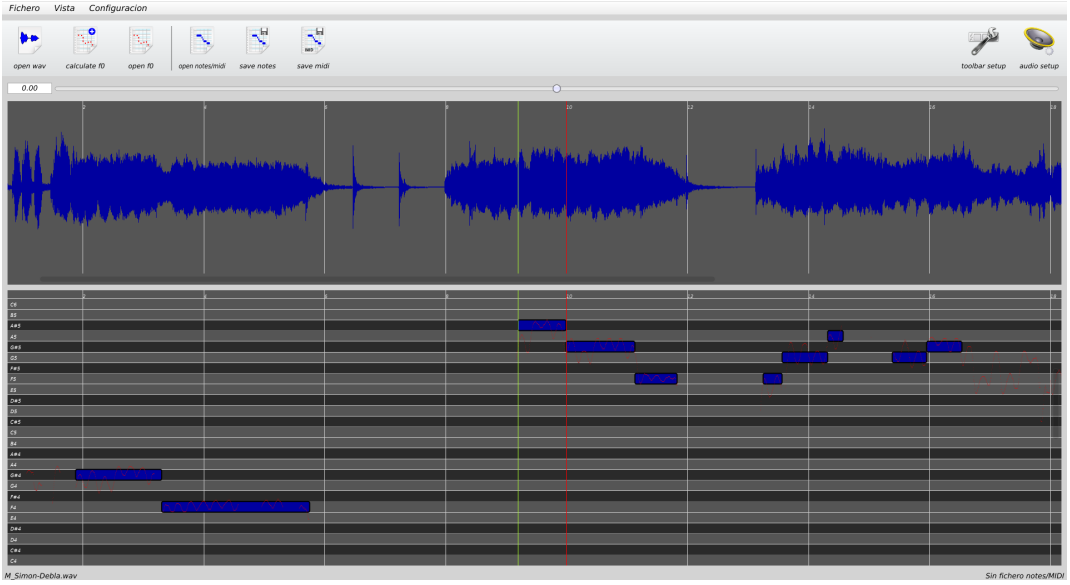


Figura B.1: Pantalla de trabajo de `JuceTranscripcion`

atajo de teclas que tambien se puede usar para disparar la misma acción que en el menú.

Existen algunas opciones que son dependientes de otras. En el caso de querer ejecutar una sin haber ejecutado la precursora, un mensaje de avisó nos recordará que no es posible.

Barra de Botones La barra de botones, nos facilita acceder al flujo de operaciones básicas en la transcripción. Si una opción no es posible ejecutarla en un momento dado, el icono estará desactivado.

Además de los botones del flujo de los ficheros, en el extremo derecho hay dos botones de configuración: el botón de configuración de la barra (que nos permite reordenar los botones existentes y elegir cuáles queremos ver). y el botón de configuración del sonido. Que se comenta en el apartado de **configuración B.2.2**.

Control de Volumen El programa tiene dos fuentes de sonido: el fichero de audio original y las notas de la transcripción. El control deslizante indica los volúmenes relativos de las dos fuentes siendo el -1 sólo la fuente de audio original y el 1 sólo las notas de la transcripción.

Para mover el control del volumen, también se pueden usar el teclado. Las teclas entre el 1 y el 9 moverá la posición del deslizador a intervalos regulares.

Visualización del audio original En esta ventana se ve los valores de la onda original. Si la señal es estéreo, se convierte a mono primeramente.

Piano-Roll Cada franja horizontal representa una nota musical (que puede verse en la zona izquierda del componente en notación anglosajona[5]). El sombreado permite localizar más fácilmente las notas siguiendo el esquema del teclado de un piano (sombra gris: notas naturales, sombra negra: notas alteradas).

Además en esta zona, se pueden ver unos puntos rojos que representan el *F*0 calculado y las notas (en azul) finales transcritas.

La vista del audio original y la del *piano roll* están sincronizadas en el tiempo, por lo que el audio mostrado siempre corresponderá con las notas marcadas en el *piano roll*.

Barra de estado En la barra de estado se muestra el nombre de los ficheros con los que se está trabajando. El nombre de la izquierda es el nombre del fichero de audio original que se está utilizando y el nombre de la derecha es el nombre del fichero MIDI con el que se trabaja.

A parte de las zonas descritas, tienen importancia capital los cursores:

Cursores A parte de las zonas indicadas, existen dos líneas verticales (verde limón y rojo) que representan respectivamente el cursor izquierdo y el derecho.

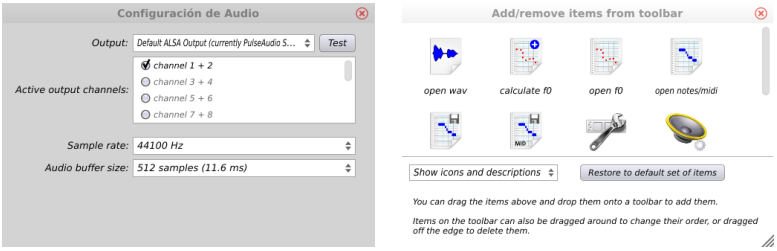
Estas líneas se pueden ver tanto en la zona del audio original como en la del *piano roll*.

B.2.2 Configuración

El *framework* `Juce` proporciona dos herramientas de configuración que se han incorporado al programa: la configuración de audio y la configuración de la barra de tareas.

El cuadro de configuración de audio (**figura B.2a**), permite seleccionar el driver de audio que se usará, la frecuencia de muestreo y el tamaño del *buffer* de audio empleado en la reproducción. La calidad de la reproducción de audios en un equipo depende de las opciones escogidas en este cuadro de diálogo. Lamentablemente, los valores más apropiados dependen del sistema operativo en el que se ejecute el programa así como del hardware. Así que se recomienda jugar con las opciones hasta encontrar valores satisfactorios.

[5] C es Do, D es Re, E es Mi, F es F, G es Sol, A es La y B es Si



a: Configuración de audio b: Configuración de *toolbar*

Figura B.2: Ventanas de configuraciónde `JuceTranscripcion`

Por otro lado, la ventana de configuración del *toolbar* (**figura B.2b**), permite reordenar y ocultar los iconos de la barra de botones por medio de arrastar los mismos. En la ventana de configuración están las instrucciones de configuración.

B.2.3 Los cursores y acciones del ratón

Quizás el aspecto del programa más difícil de asimilar y acostumbrarse es la gestión y manejo de los cursores. Los cursores se basan en dos principios:

- El cursor izquierdo (línea vertical color verde) siempre estará a la izquierda del cursor derecho (linea verde).
- Toda acción de creación o borrado de notas se efectúa en la región comprendida entre ambos cursores. En el caso de creación de notas, la duración de la nota creada siempre corresponderá al tamaño de la región marcada entre los dos cursores.

Existen tres tipos de acciones relacionadas con los cursores y el ratón: opciones de movimiento, de creación de notas y de borrado de notas. En la **figura B.3** se resumen las acciones posibles que se comentan a continuación.

Movimiento de los cursores

El movimiento de los cursores puede hacerse de forma libre (situándolo donde se desee) o situarlo usando de referencia otras notas ya situadas.

Click izquierdo (derecho) sobre el fondo Mueve el cursor izquierdo (derecho) donde se haya marcado. Se puede marcar sobre el fondo del *piano roll* o sobre el audio original.

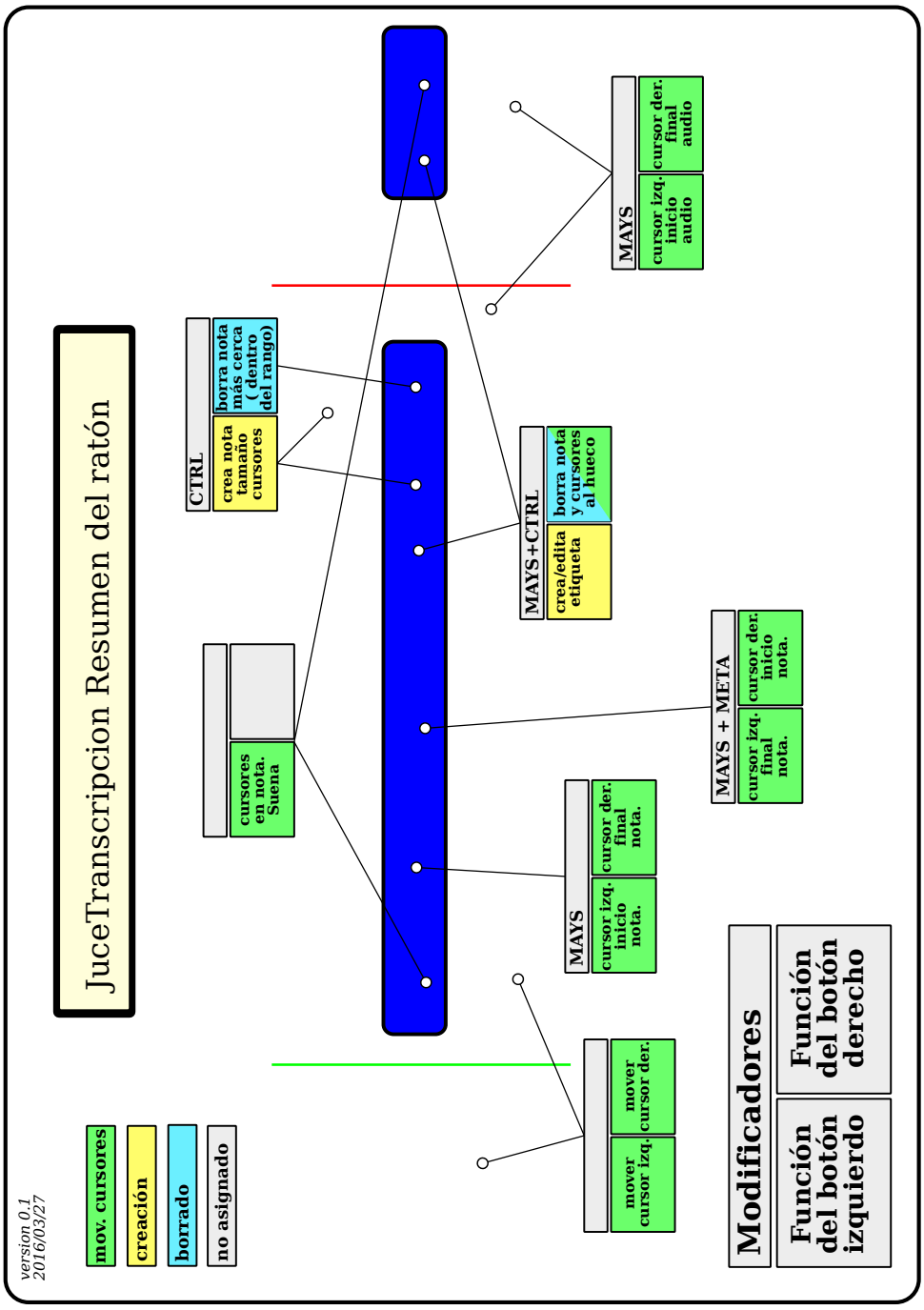


Figura B.3: `JuceTranscripcion`: Referencia del ratón

Click izquierdo sobre una nota Mueve los dos cursores a los extremos de la nota seleccionada. Y simultáneamente suena las notas en la región seleccionada.

MAYS + Click izquierdo (derecho) sobre una nota Mueve el cursor izquierdo (derecho) al inicio (final) de la nota seleccionada.

MAYS + META + Click izquierdo (derecho) sobre una nota Mueve el cursor izquierdo (derecho) al final (inicio) de la nota seleccionada. La tecla `META` depende del sistema operativo empleado, correspondiendo a la tecla `ALT` en windows y linux.

MAYS + Click izquierdo (derecho) sobre fondo Mueve el cursor izquierdo (derecho) al inicio (final) de la pieza. Se puede marcar sobre el fondo del *piano roll* o sobre el audio original.

Si en algún movimiento efectuado se produce una inversión en los cursores (esto es, que el resultado de la acción pretendida deje el cursor derecho a la izquierda del cursor izquierdo) estos intercambiarán sus posiciones para mantener la coherencia de la posición relativa entre cursores.

En el apartado de **casos de uso B.3.1** de la **sección B.3** (concretamente en la transcripción de varias notas en cadena) de descripción del proceso de transcripción se muestra la utilidad de este comportamiento.

Inicialmente, los cursores están situados al inicio y al final de la grabación de audio.

Creación de notas

CTRL + Click derecho en región Crea una nota con inicio en el cursor izquierdo y final en el derecho. La frecuencia de la nota depende de la altura en la que se hace click dentro del *piano roll*.

Borrado de notas

CTRL + Click derecho en región sobre nota Borra la nota marcada. Si en el punto que se ha hecho click hay varias notas superpuestas, borra la que su centro está más cercano del punto de click.

Existe una operación mixta de movimiento de cursor y borrado que no requiere que la nota borrada esté entre los cursores.

MAYS + CTRL + Click derecho sobre nota Mueve los cursores sobre una nota y lo borra. Lo que permite crear una nueva nota en el hueco borrado.

B.2.4 Vista y nivel de zoom

Las vistas del audio original y el *piano roll* están sincronizadas en el tiempo. Estas vistas pueden modificarse para poder apreciar mejor los detalles. Las operaciones posibles son:

Zoom en el tiempo Con la rueda del ratón (cuando el cursor se encuentra en la vista de audio original) se puede variar el nivel de detalle en la visualización

Desplazamiento horizontal en el tiempo Arrastrando horizontalmente sobre la vista de audio o sobre el *piano roll* se efectúa el desplazamiento por el tiempo.

La vista de audio presenta también una barra de desplazamiento en la parte inferior que permite hacerse una idea del tamaño del audio visible y efectuar desplazamientos rápidos en el tiempo.

Esta operación no altera la escala de visualización.

Desplazamiento de tesitura El *piano roll* siempre muestra dos escalas completas (suficiente para la mayoría de los casos), es posible modificar qué dos escalas se ven usando la rueda del ratón en la vista del *piano roll*.

Además `Jucetranscripcion` presenta una serie de vistas predeterminadas para facilitar su rápido acceso:

Vista completa Muestra toda la grabación de audio. Se accede usando el menú de Vista o con `CTRL`+o.

Vista de cursores Cambia la escala adecuadamente para que se vea la región entre los cursores y un poco de margen a cada lado. Esto permite cambiar el nivel de zoom cómodamente al trabajar con una frase o motivo sin tener que buscar el punto con interminables desplazamientos en la vista de audio. Para ello, se selecciona el motivo completo, y se llama a esta vista (`CTRL`+i).

La vista no sigue a los cursores si estos cambian. Esta permanecerá en la escala establecida hasta que se cambie a otra visualización.

Vista anterior La escala escogida es la misma que en la vista de cursores, pero en esta ocasión dejará al cursor izquierdo a la derecha de la vista, para poder ver con detalle qué hay antes de la zona seleccionada. (`CTRL`+u)

Vista posterior Análogo a la vista anterior, pero en este caso se muestra la zona posterior a los cursores. (`CTRL`+p)

B.2.5 El teclado

Se añade, a modo de resumen, un diagrama (**figura B.4**) con todos los atajos de teclado que el programa reconoce.

B.2.6 Otras funcionalidades

Cálculo del F0

El programa `JuceTranscripcion` es capaz de estimar el valor de *F*0 por medio del algoritmo MELODIA (Salamon & Gómez, 2012) en su implementación en el *framework* Essentia (Bogdanov, et al., 2013) desarrollada por el *Music Technology Group* de la Universidad Pompeu Fabra de Barcelona.

Una vez cargado el fichero de audio en el programa, activando la opción de calcular el *F*0, borrará cualquier pista de *F*0 cargada previamente y la sustituirá por la nueva pista calculada.

*Variando la afinación del F*0

Dado que el audio original que se quiere transcribir no tiene porque estar afinado con la nota La a 440 Hz, es posible que el *F*0 resultante quede entre dos notas. Usando los cursores del teclado (arriba y abajo) se puede desplazar el *F*0 para mejorar la identificación de las notas.

Lamentablemente, solo cambia la visualización de *F*0 y no la generación del sonido de las notas transcritas. Si la variación de afinación es muy acusada, la reproducción del audio simultáneamente con las notas puede producir una disonancia entre ambas que dificulte la verificación de la transcripción.

Etiquetar notas o motivos

Es posible etiquetar cada nota de la transcripción. El propósito es poder marcar el inicio de cada motivo o sección relevante de

Figura B.4: `JuceTranscripcion`: Referencia del teclado

la grabación. Y, aunque no ha sido diseñado para esto, también podría usarse para anotar la sílaba cantada en esa nota.

Las etiquetas se transforman en eventos de letra en el fichero midi, por lo que es información recuperable posteriormente con programas que procesen ficheros MIDI.

El combinación `MAYS+CTRL`+click izquierdo sobre la nota a marcar abre el cuadro de diálogo de edición de la etiqueta. En caso de querer borrar una etiqueta, abrir el cuadro de diálogo de edición de la misma y borrar el texto que aparezca.

B.3 Proceso de Transcripción

Flujo de trabajo propuesto para realizar una transcripción se muestra en la **figura B.5**.

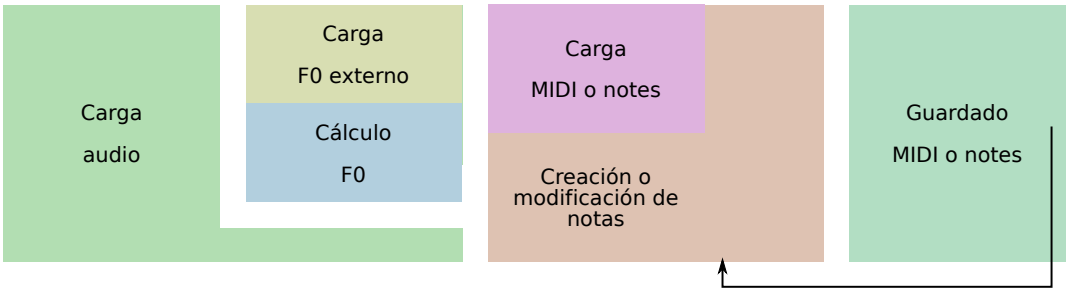


Figura B.5: Flujo del proceso de transcripción de una pieza

1. Se carga el fichero de audio a transcribir. Soporta los formatos `wav`, `mp3` y `ogg`.
2. (Opcional) Se carga o se calcula el $F0$ a partir del audio. La posibilidad de cargar un fichero $F0$[6]. permite usar la estimación realizada por otras aplicaciones (y posiblemente con otros algoritmos).
3. (Opcional) Carga un fichero «notes» o MIDI que contengan un trabajo de transcripción inacabado o que se quiera modificar.
4. Se crean o modifican las notas existentes.
5. Se guarda el fichero notes o MIDI generados (repitiendo los últimos dos pasos las veces que sea necesario).

[6] El formato del fichero es un fichero de texto en el que cada fila tiene dos números `double` separados por un espacio. El primer número representa el tiempo (en segundos) en el que se ha calculado el $F0$ y el segundo la frecuencia de este.

B.3.1 Acciones habituales de transcripción

En el proceso de creación y modificación de notas (el grueso del trabajo de la transcripción), existen una serie de acciones típicas

que se producen frecuentemente. Se ve conveniente documentar estos casos de uso habituales con las acciones más adecuadas para completar dichas tareas.

A continuación se enumeran estos casos de uso.

Crear una nota Se marca el inicio y final de la nota con los cursores y CTRL + click dentro de la región seleccionada.

Crear un motivo musical o varias notas sucesivas en el que el final de una nota coincida con el inicio de la siguiente. Para ello se parte de una nota previamente creada y con los cursores en sus extremos (en caso de no tener los cursores en posición, hacer click sobre esta nota).

1. Se marca el final de la nota a crear con el cursor izquierdo[7].
2. (Los cursores ahora marcan el hueco de la nueva nota). Se crea la nota (CTRL+click) en el tono adecuado.
3. Se repite las veces necesarias hasta terminar el motivo.

Corregir el tono de una nota Como se ha mencionado, no existe la edición de notas. El procedimiento es:

1. Selección y borrado de nota errónea (MAYS + CTRL + click derecho).
2. (Los cursores marcan el hueco de la nota borrada). Se crea nota en el tono adecuado

Modificar el instante de inicio de una nota

1. Selección de la nota errónea (click izquierdo sobre nota).
2. Mover el cursor izquierdo
3. Borrado de la nota errónea (CTRL+click derecho)
4. Creado de la nueva nota (CTRL+click izquierdo)[8].

Esta operación puede efectuarse más rápidamente haciendo selección y borrado de la nota (MAYS + CTRL + click derecho), moviendo el extremo izquierdo y creando la nueva nota; pero se pierde la referencia visual de la nota antigua.

Modificar el instante final de una nota

B.7 Si el final de la nueva nota se hubiera marcado con el cursor derecho, los cursores estarían entre el inicio de la nota antigua y el final de la nueva. Marcando con el cursor izquierdo hace que se fuerce el intercambio de cursores y que finalmente quede el cursor izquierdo donde originalmente estaba el derecho y el derecho al final de la nota que queremos crear.

B.8 Entre los últimos dos pasos no es necesario mover el ratón. Manteniendo pulsado la tecla CTRL se puede hacer una pulsación del botón derecho y luego del izquierdo

1. Selección de la nota errónea (click izquierdo sobre nota).
2. Mover el cursor derecho
3. Borrado de la nota errónea (CTRL+click derecho)
4. Creado de la nueva nota (CTRL+click izquierdo).

Modificar la transición entre dos notas de un motivo En este caso se tienen dos notas seguidas y se quiere cambiar el instante de transición entre las notas. Es posible usar los procedimientos ya descritos de modificación del final de la primera nota y el inicio de la segunda; pero este procedimiento garantiza que ambos puntos coinciden en el tiempo.

1. Selección de la primera nota errónea (click izquierdo sobre nota).
2. Mover el cursor derecho a la nueva posición
3. Borrado de la nota (CTRL+click derecho) y
4. Creación en la nueva nota (CTRL+click izquierdo).
5. Movimiento del cursor izquierdo al final de la segunda nota (MAYS+META+click izquierdo)
6. Borramos la segunda nota y creamos la nueva nota (CTRL+ click derecho y CTRL+click izquierdo).

Copiar y pegar notas El copiado y pegado de notas solo tendría sentido en el caso de la repetición fuera idéntica. En caso contrario, habría que editar y modificar todas las notas pegadas siendo más costoso que hacer la transcripción desde el principio.

Esta función no se ha implementado.

B.3.2 Guardando ficheros, nombrando los ficheros

El guardado de ficheros está diseñado para no ser destructivo y hacer perder el menor tiempo posible. Es por ello que el programa, cada vez que se ordena guardar el fichero de salida, genera el nombre del fichero a guardar de forma automática.

El nombre del fichero se forma a partir de dos elementos: un nombre base tomado del fichero MIDI que se abrió (si las notas se han creado de cero, el programa pregunta por un nombre) y una marca de tiempo. Cada vez que se guarde el progreso de

la transcripción, todas las grabaciones usarán el mismo nombre base que permite identificar la sesión o el fichero de audio con el que se está trabajando y una marca de tiempo diferente que indicará la versión del trabajo. La marca de tiempo es una versión compacta del estándar RFC3339 (Klyne & Newman, 2002) en el que primero se mencionan las unidades temporales de menor precisión: año, mes, día, horas, segundos.

Este esquema temporal, agrupa las distintas sesiones de trabajo y las ordena por antigüedad. Una vez terminado el trabajo, y fuera de `JuceTranscripcion`, se pueden borrar todas las copias de trabajo intermedias que se han ido guardando (lo que es fácil de hacer porque tienen la fecha y hora del momento de la grabación y aparecen ordenados en el tiempo).

C Tablas de resultados

C.1 Listado de descriptores específicos de las tonás

Debla

Descriptor		Valores
PClass2_T	=	BAsAsBGsG, BCCBAA, BCCBAGs, BCCBBA, BCCBGsA, CBAGsGsA, CBBCAGs, CBBCCA, CBGsGGGs, GsGGGsBAs
PClass2_T_o	=	AABCCB, AGsBCCB, AGsCBGsA, AsBBAsGsG, BABCCB, BAsGGsGsG, BCCACB, BCCBGsA, CBGGsGsG
Salto	<	-1.5
Pendiente	<	-0.01973684
Direccion	=	descendente
PendienteRegresion	<	-0.005840428
PClass1_T_rango_0_a_50	=	AsBC, BAsGs, BCA, CBA, CBAs, CBD
PClass1_T_rev_rango_0_a_50	=	BAsGs, BCA, CBA, CBAs, CBCs, CBD, GsAsB
PClass2_1_rango_0_a_50	=	AsB, BAs, BC, CB
PClass2_T_rango_0_a_50	=	AsBBAsAsC, BAsAsBGsG, BCCBAGs,

Descriptor		Valores
		BCCBBA, BCCBCC, BCCBCD, CBBCAGs, CBBCBA, CBBCCC, CBBCCD, CBBCCsC
PClass2_T_o_rango_0_a_50	=	AGsBCCB, AsBAsCBAs, AsBBAsGsG, BABCCB, BCCBCC, BCCBCD, BCCBCsC
PClass2_T_rev_rango_0_a_50	=	AsBBAsAsC, BCCBAB, BCCBCC, BCCBCsC, CBBCAB, CBBCBB, CBBCCC, CBBCDC, CBBCGsA, GGsGsGAsB
PClass2_T_rev_o_rango_0_a_50	=	ABBCCB, AsBAsCBAs, AsBGGsGsG, BBBCCB, BCCBCC, BCCBCsC, BCCBDC, BCCBGsA
PClass_C_rango_0_a_33	>=	0.1388235
PClass1_T_rango_0_a_33	=	AsBC, BAsC, BCA, BCD, CAsB, CBA, CBD
PClass1_T_o_rango_0_a_33	=	CAB, CAsB, CDB
PClass1_T_rev_rango_0_a_33	=	AsBC, AsBGs, BCA, BCGs, CAsB, CBA, CBD
PClass1_T_rev_o_rango_0_a_33	=	CAB, CAsB, CDB, CGsB, GsAsB
PClass2_1_rango_0_a_33	=	BAs, BC, CAs, CB
PClass2_T_rango_0_a_33	=	BAsAsBAsC, BAsCCAsB, BCCBBA, BCCBCC, CAsCBBC, CBBCAGs, CBBCBA, CBBCCC, CBBCCD
PClass2_T_o_rango_0_a_33	=	AGsBCCB, AsBAsCBAs, AsBBAsCC, BABCCB, BCCAsCB, BCCBCC, BCCBCD

Descriptor		Valores
PClass2_1_rev_rango_0_a_33	=	BAs, BC, CB
PClass2_T_rev_rango_0_a_33	=	BAsAsBAsC, BAsAsBCB, BAsAsBGGs, BCCBAB, BCCBCC, BCCBCD, CBBCAsGs, CBBCBA, CBBCBB, CBBCCC, CBBCDC, CBBCGA, CBGsAAGs
PClass2_T_rev_o_rango_0_a_33	=	ABBCCB, AGsCBGsA, AsBAsCBAs, AsBBAsCB, AsBBAsGGs, AsGsBCCB, BABCCB, BBBCCB, BCCBCC, BCCBCD, BCCBDC, BCCBGA
PClass2_T_rango_33_a_67	=	AsBBAsGsAs, BAABAGs, BAsAsBAsGs, BCCBAB, BCCBBA, BCCBGsA, CBAGsBC, CBAGsGsA, CBBCAGs, CBBCBA, CBBCCCs, CBBCCsD
PClass2_T_rev_rango_33_a_67	=	ABBACB, BAsAsBAsGs, BCCBAB, BCCBGsA, CBAGsBC, CBBCAB, CBBCBA, CBBCCsC, CBBCDC, CBBCGsA, CBBCGsGs, GGsGsGAsGs
PClass2_T_rev_o_rango_33_a_67	=	ABBACB, ABBCCB, AGsBCCB, AsBAsGsBAs, AsGsGGsGsG, BABCCB, BCCBCsC, BCCBDC,

Descriptor	Valores
	BCCBGsA, BCCBGsGs

Martinete 1

Descriptor	Valores
PClass1_T	= ABFs, AFsGs, AGFs, AGsE, AGsG, AsAG, EAG, FsGGs, GGsA, GGsE, GGsFs, GsAE, GsAFs, GsAG, GsEA, GsGA
PClass2_T	= AAABBA, AAEEGsA, AAsAsAGA, AGsGsGGGs, EEAAAGs, EEGsAAA, EEGsAAGs, EEGsGGGs, FsGsGsAAA, FsGsGsAAGs, GGsGsGEFs, GGsGsGFsG, GGsGsGGA, GGsGsGGsGs, GsAAGsAAs, GsAAGsEE, GsAAGsFsGs, GsAAGsGA, GsAAGsGsG, GsAAGsGsGs, GsAFsGsAB, GsAGFsEE, GsAGGsAGs, GsGAGsEE, GsGEEGGs
PClass2_T_rango_0_a_50	= AAABBA, AAsAsAAsC, AGsGGsCE, EEBEEFs, EECEEFs, EEEDsDsE, EEFsGGG, EEFsGsGsA,

Descriptor	Valores
	EEGGsGsA,
	EEGGsGsGs, EEGsAAA,
	EEGsAAGs, EEGsADsE,
	EEGsAEFs, EEGsGsBE,
	EEGsGsGsA,
	EFsFsEGG,
	FsGsGsAAGs,
	FsGsGsAAsDs,
	FsGsGsAEE,
	GGsGsGEE, GsAAGsEE,
	GsAAGsEF, GsAEEAGs,
	GsGsEEGGs
PClass2_T_o_rango_0_a_50	= AAABBA, AAEEGsA,
	AAsAsAAsC,
	AGsCEGGs, AGsEEGsA,
	AGsEFGsA,
	AGsFsGsGsA,
	AsDsFsGsGsA,
	BEEEEFs, BEEEGsGs,
	CEEEEFs, DsEEDsEE,
	DsEEEGsA, EEEFsGsA,
	EEFsGGG, EEFsGsGsA,
	EEGGsGsA, EEGGsGsG,
	EEGGsGsGs,
	EEGsAGsGs, EFsFsEGG
PClass2_T_rev_rango_0_a_50	= AABAAB, AGsGsAEE,
	AGsGsAFsGs,
	AGsGsAGsGs,
	AsAAAsAG, EEAGsAA,
	EEAGsGsA,
	EEFsGsAGs, EEGsAAA,
	EEGsAAGs, EEGsAGGs,
	EEGsGAGs,
	EEGsGGsGs,
	EEGsGsGsG,

Descriptor	Valores
	EFsGsGGGs,
	FsGGsGGGs,
	GGsAGsDsG,
	GGsAGsGsA,
	GsAAAGsGs,
	GsAAGsEE,
	GsAAGsGGs, GsAEEAA,
	GsAEEAB, GsAEEAsA,
	GsAFsGsGsFs,
	GsAFsGsGsG,
	GsGGGsEE,
	GsGsAGsGsA,
	GsGsGGsEE
PClass1_T_rango_50_a_100	= AFsG, AGAs, AGFs,
	AGGs, AGsFs, AGsG,
	GAGs, GFsA, GFsGs,
	GGsA, GGsB, GGsF,
	GGsFs, GsAAs,
	GsAFs, GsAG, GsGA,
	GsGFs
PClass2_T_rango_0_a_33	= AAABBA, AAsAsAAsC,
	CEEFsFsG, EEAAAE,
	EEAsEEFs, EEBEEDs,
	EEBEEFs, EECEEFs,
	EEDsEGsA, EEEDsDsE,
	EEFsGsGsA, EEGGGGs,
	EEGGsAsE, EEGGsBE,
	EEGsAAA, EEGsAAE,
	EEGsAAGs, EEGsAAsE,
	EEGsABE, EEGsACE,
	EEGsGsBE, EFFEGsGs,
	EFsFsEFsG,
	FsGsAsDsDsE
PClass2_T_o_rango_0_a_33	= AAABBA, AAAEEE,
	AAEEGsA, AAsAsAAsC,

Descriptor	Valores
	AEEEGsA, AGsEEGsA,
	AsDsDsEFsGs,
	AsEEEEFs, AsEEEGGs,
	AsEEEGsA, BEEDsEE,
	BEEEEFs, BEEEGGs,
	BEEEGsA, BEEEGsGs,
	CEEEEFs, CEEEGsA,
	CEEFsFsG, DsEEDsEE,
	DsEEEGsA,
	EEFsGsGsA, EEGGGGs,
	EFFEGsGs, EFsFsEFsG
PClass2_T_rev_rango_0_a_33 =	AABAAB, AAsAsACAs,
	AGsGsAEE,
	AGsGsAFsGs,
	EEAAGsA, EEAGsAA,
	EEAGsGsA,
	EEAsEGsAs,
	EEDsEEDs, EEDsEGsA,
	EEEFsBE, EEGGsGsG,
	EEGsAAA, EEGsAAB,
	EEGsAAGs, EEGsAAsA,
	EEGsACB, EEGsAGsGs,
	EEGsGsFsGs,
	EFsFsEAsB,
	FsGsGFsAG,
	GGsEEGsG, GGsGGEE,
	GsAEEAGs,
	GsAEEGsGs,
	GsAFsGsEE,
	GsAGGsGsG,
	GsAGsGsFE, GsGsEEGGs
PClass2_T_rev_o_rango_0_a_33 =	AAABBA, AAAGsEE,
	AAEEGsA, AAsAsACAs,
	ABEEGsA, AGFsGsGFs,
	AGsEEGsA,

Descriptor	Valores
	AGsFsGsGsA,
	AsAEEGsA,
	AsBEFsFsE,
	AsEEEGsAs, BEEEEFs,
	CBEEGsA, DsEEDsEE,
	DsEEEGsA,
	EEFsGsGsA,
	EEFsGsGsGs,
	EEGGGGs, EEGGsGsG,
	EEGGsGsGs,
	EEGsAGsGs,
	FEGsAGsGs, GGsGsAGsG

Martinete 2

Descriptor	Valores
PClass2_T	= AAAGsAB, AAAGsEA,
	AAAGsGsA, AADDCsD,
	AAGsAAGs,
	AAsAsAAsAs,
	AAsAsAAsC,
	AGsAAGsA, AGsGsAAA,
	AGsGsAAB,
	AGsGsAGsGs,
	AsAAAsAA, EFFEAA,
	FsGGAAG, GsAAAAGs,
	GsAAGsAA, GsAAGsAB
PClass2_T_rango_0_a_50	= AAACsCsD, AAAGsGsA,
	AAEAAAs, AAEAAGs,
	AAFsAAFs, AAGsAAGs,
	AAsAsAAA,
	AAsAsAsAsA,
	AGsAAGsA, AGsGsAAA,
	AGsGsGsGsA,
	FsGGGFsFs,
	GsAAAAGs, GsAAGsAA

Descriptor	Valores
PClass2_T_o_rango_0_a_50	= AAAAsAsA, AAAAsEA, AAACsCsD, AAAFsFsA, AAAGsEA, AAAGsGsA, AAsAsAAsAs, AGsGsAGsGs, FsFsFsGGG
PClass2_T_rev_rango_0_a_50	= AAABGsA, AAAsAAAs, AABACB, AADCsDD, AAEFDE, AAGsAAGs, AAsAsAAsAs, AAsAsAAsC, AGsAAGsA, AGsGsAAA, AsAAAsAA, GGFsGAG, GsAAAAGs, GsAAGsAB, GsAAGsBA, GsAAGsGsGs
PClass2_T_rango_50_a_100	= AAAGsBA, ABBAAGs, ABBBBC, ACCBBB, AGsAsAGsA, AGsGsAAB, AGsGsABA, AGsGsABC, AGsGsGGGs, AsAAGsCD, BAAGsCD, BCBAAGs, BCCBAB, CAsAsAAAs, CCCAsAsA, DCsDDCsD, FEEFEE, GAAGGFs
PClass2_T_rango_0_a_33	= AAAAsAsA, AAACsNA, AAAGsEA, AAAGsGsA, AAEAAAs, AAEAAG, AAEAAGs, AAFsAAFs, AAGsAAB, AAsAsAAA, AAsAsAsAsA, AGsGsAAA, FsGAAsAsA, GsAAAAGs, GsGsAGsGsA

Descriptor	Valores
PClass2_T_o_rango_0_a_33	= AAAAsAsA, AAAAsEA, AAABGsA, AAACs, AAAFsFsA, AAAGEA, AAAGsEA, AAAGsGsA, AAsAsAAsAs, AAsAsAFsG, AGsGsAGsGs
PClass2_T_rev_rango_0_a_33	= AAABGsA, AAACsNA, AAAGsEA, AAAGsGsGs, AAAsAAAs, AACCsBC, AADDCD, AAGsAAGs, AAGsAFsGs, AAsAsAsAsA, AGsAAGsA, AsAAAsAA, GGFsGAsA, GsAAAAsGs, GsAAABA, GsAAGsAA, GsGsGsAAGs
PClass2_T_rev_o_rango_0_a_33	= AAAAsAsA, AAABGsA, AAACs, AAAGsEA, AAAGsGsA, AAAGsGsGs, AAAsGsGsA, AABAGsA, AABCCCs, AACDDD, AAFsGsGsA, AAsAsAAsAs, AGsGsAGsGs, AsAFsGGG
PClass2_T_rango_67_a_100	= ABBBBC, AGGFsFsG, AGsAAGsA, AGsBCCB, AGsGsABC, AGsGsGGGs, AsAAGsDCs, BAAAAB, BAAGsBB, BBBCCD, BCCBAGs, BCCBBA, BCCDDCs, CAsAsAAAs, CBDCBC, CCsCsCCC,

Descriptor	Valores
	CDDCCB, DDEFFE, FEFsFEF

C.2 Listado de descriptores específicos del fandango de Huelva

Cluster 1

Descriptor	Valores
PClass2_T	= EAABBC, GEEENA
PClass2_T_o	= ABBCEA, EEGE
PClass2_T_rev	= BCABEA, EEGENA
PClass2_T_rev_o	= ABBCEA, EEGE
Pendiente	>= 1.8
PendienteRegresion	>= 2.35
PendienteRegresion_rango_0_a_50	>= 3.5
PClass1_T_rango_50_a_100	= ABC, ENULLNULL
PClass1_T_rev_rango_50_a_100	= CBA, ENULLNULL
PClass2_T_rango_50_a_100	= ABBCNA, EENANA
PClass2_T_o_rango_50_a_100	= ABBC, EE
PClass2_T_rev_rango_50_a_100	= BCABNA, EENANA
PClass2_T_rev_o_rango_50_a_100	= ABBC, EE
PClass2_T_rango_0_a_33	= EANANA, GENANA
PClass2_T_o_rango_0_a_33	= EA, GE
PClass2_1_rev_rango_0_a_33	= EA, GE
PClass2_T_rev_rango_0_a_33	= EANANA, GENANA
PClass2_T_rev_o_rango_0_a_33	= EA, GE
PendienteRegresion_rango_0_a_33	>= 4.25
PClass1_T_rango_33_a_67	= EAB, EGNULL
PClass1_T_rev_rango_33_a_67	= CBA, EGNULL
PClass2_1_rango_33_a_67	= EA, GE
PClass2_T_rango_33_a_67	= EAABBC, GEEENA
PClass2_T_o_rango_33_a_67	= ABBCEA, EEGE
PClass2_T_rev_rango_33_a_67	= BCABEA, EEGENA
PClass2_T_rev_o_rango_33_a_67	= ABBCEA, EEGE

Descriptor	Valores
Pendiente_rango_33_a_67	>= 1.833333
PendienteRegresion_rango_33_a_67	>= 2.55
PClass1_T_rango_67_a_100	= BCNULL, ENULLNULL
PClass1_T_o_rango_67_a_100	= CB, E
PClass1_T_rev_rango_67_a_100	= CBNULL, ENULLNULL
PClass1_T_rev_o_rango_67_a_100	= CB, E
PClass2_T_rango_67_a_100	= BCNANA, EENANA
PClass2_T_o_rango_67_a_100	= BC, EE
PClass2_T_rev_rango_67_a_100	= BCNANA, EENANA
PClass2_T_rev_o_rango_67_a_100	= BC, EE

Cluster 2

Descriptor	Valores
PClass2_T	= BCCBBA, DsFCsCsCsDs, DsFFDsFG, EDDEEG, EEEFFE, FGGFFE, FGGGsGsAs
PClass2_T_rev_rango_0_a_50	= AEBACB, AsGsGsAsGGs, DsFFDsCsDs, EEFEEF, GFEGDE, GFFGNA, GFGsGGGs
PClass2_T_rev_o_rango_0_a_50	= AEBACB, AsGsGGsGsAs, CsDsDsFFDs, DEEGGF, EEEFFE, FGGF, GFGGsGsG
PClass2_T_rev_rango_0_a_33	= BACBBC, EGDEED, FDsDsFCsDs, FEEFEE, FGNANA, GsAsGGsFG, GsGGGsFG
PClass2_T_rango_33_a_67	= BAAEEF, EGGFFD, FDsDsFFG, FEEEEF, FGGFFE, GsAsAsGsGsF, GsGGFFDs
PClass2_T_o_rango_33_a_67	= AEBAEF, AsGsGsAsGsF, DsFFDsFG, EEEFFE, EGFDGF, FDsGFGsG, FEFGGF

Descriptor	Valores
PClass2_T_rev_rango_33_a_67	= DEFDGF, DsFFDsGF, FEGFFG, FGEFAE, FGEFEE, FGGsFAsGs, GDsFGDsF
PClass2_T_rev_o_rango_33_a_67	= AEEFFG, AsGsFGGsF, DEFDGF, DsFFDsGF, DsFFGGDs, EEEFFG, FEFGGF

Cluster 3

Descriptor	Valores
PClass2_T_rev	= DCDDED, DCsEDEE, DCsEDFsE, DDDCsED
PClass2_T_rev_o	= DCDDED, DCsDDED, DCsEDEE, DCsEDFsE
PClass2_T_rev_rango_0_a_50	= BECBDC, EECEDC, EFsDEDD
PClass2_T_rev_o_rango_0_a_50	= BECBDC, CEDCEE, DDDEEFs
PClass1_T_rev_rango_50_a_100	= DEC, DECs, ECsD
PClass2_T_rango_50_a_100	= BEEEED, EEEDDD, EFsFsEED
PClass2_T_o_rango_50_a_100	= BEEDEE, DDEDEE, EDEFsFsE
PClass2_T_rev_rango_50_a_100	= DCDDED, DCsDDED, DCsEDEE, DCsEDFsE
PClass2_T_rev_o_rango_50_a_100	= DCDDED, DCsDDED, DCsEDEE, DCsEDFsE
PClass1_T_rango_0_a_33	= DCB, DCE, DCsE
PClass2_T_rango_0_a_33	= CsDDDDE, DDDCCB, DDDCCE
PClass2_T_o_rango_0_a_33	= CBDCDD, CEDCDD, CsDDDDE
PClass2_T_rev_rango_0_a_33	= CBDCDD, CEDCDD, DEDDCsD
PClass2_T_rev_o_rango_0_a_33	= CBDCDD, CEDCDD, CsDDDDE
PClass1_T_rango_33_a_67	= ECB, ECD, EDFs

Descriptor	Valores
PClass1_T_rev_rango_ 33_a_67	= EBC, EDB, EDC, EFsD
PClass2_T_rango_33_a_67	= CBBEEE, CEEEED, DEEFsFsE
PClass2_T_o_rango_33_a_67	= BECBEE, CEEDEE, DEEFsFsE
PClass2_T_rev_rango_ 33_a_67	= EDEEBE, EDEECE, EEBECB, FsEEFsDE
PClass2_T_rev_o_rango_ 33_a_67	= BECBEE, BEEDEE, CEEDEE, DEEFsFsE
PClass2_T_rango_67_a_100	= EDDDDC, EDDDDCs, EEEDDCs, FsEEDDCs
PClass2_T_o_rango_67_a_ 100	= DCDDED, DCsDDED, DCsEDEE, DCsEDFsE
PClass2_T_rev_rango_67_a_ 100	= DCDDED, DCsDDED, DCsEDEE, DCsEDFsE
PClass2_T_rev_o_rango_ 67_a_100	= DCDDED, DCsDDED, DCsEDEE, DCsEDFsE

Cluster 4

Descriptor	Valores
PClass1_T	= EBCs, EDCs
PClass1_T_rev	= EACs, EBCs
PClass1_T_rev_o	= CsEA, CsEB
PClass2_T	= EEBCsCsD, EEEDDCs
PClass2_T_o	= BCsCsDEE, DCsEDEE
PClass2_T_rev	= EEBACsB, EECsADCs
PClass2_T_rev_o	= BACsBEE, CsADCsEE
PClass1_T_rango_0_a_50	= EBCs, ENULLNULL
PClass1_T_o_rango_0_a_50	= CsEB, E
PClass1_T_rev_rango_0_a_50	= EDCs, ENULLNULL
PClass1_T_rev_o_rango_0_a_50	= CsDE, E
PClass2_T_rango_0_a_50	= EEBCsCsD, EENANA
PClass2_T_o_rango_0_a_50	= BCsCsDEE, EE
PClass2_T_rev_rango_0_a_50	= EEDECsD, EENANA
PClass2_T_rev_o_rango_0_a_50	= CsDDEEE, EE
PClass1_T_rev_rango_50_a_100	= EAB, EACs

Descriptor	Valores
PClass2_T_rango_50_a_100	= EEEDDCs
PClass2_T_o_rango_50_a_100	= DCsEDEE
PClass2_T_rev_rango_50_a_100	= BACsBDCs, CsADCsED
PClass2_T_rev_o_rango_50_a_100	= BACsBDCs, CsADCsED
PClass1_T_rango_33_a_67	= EDCs, EDNULL
PClass1_T_o_rango_33_a_67	= CsDE, DE
PClass1_T_rev_rango_33_a_67	= EDCs, EDNULL
PClass1_T_rev_o_rango_33_a_67	= CsDE, DE
PClass2_T_rango_33_a_67	= EEDEED, EEEDNA
PClass2_T_o_rango_33_a_67	= DEEDEE, EDEE
PClass2_T_rev_rango_33_a_67	= EEDCsED, EEEDNA
PClass2_T_rev_o_rango_33_a_67	= DCsEDEE, EDEE
PClass1_T_rev_rango_67_a_100	= ABCs, ACsD
PClass1_T_rev_o_rango_67_a_100	= CsAB, CsDA
PClass2_T_rango_67_a_100	= DCsCsBBA, EDDCsCsA
PClass2_T_o_rango_67_a_100	= BACsBDCs, CsADCsED
PClass2_T_rev_rango_67_a_100	= BACsBDCs, CsADCsED
PClass2_T_rev_o_rango_67_a_100	= BACsBDCs, CsADCsED

Cluster 5

Descriptor	Valores
MPitch	>= 75.73636
PClass2_T	= DEEGGF, FFFEEF
PClass2_T_o	= DEEGGF, EFFEFF
PClass2_T_rev	= FEGFEG, FFFEDC
PClass2_T_rev_o	= DCFEFF, EGFEGF
MPitch_rango_0_a_50	>= 76.16667
PClass1_T_rango_0_a_50	= DEG, FENULL
PClass1_T_rev_rango_0_a_50	= FENULL, GED
PClass2_T_rango_0_a_50	= DEEGNA, FFFEEF
PClass2_T_o_rango_0_a_50	= DEEG, EFFEFF
PClass2_T_rev_rango_0_a_50	= EGDENA, FFEFFE
PClass2_T_rev_o_rango_0_a_50	= DEEG, EFFEFF
MPitch_rango_0_a_33	>= 76.26667
PClass1_T_rango_0_a_33	= DEG, FENULL

Descriptor		Valores
PClass1_T_rev_rango_0_a_33	=	FENULL, GED
PClass2_T_rango_0_a_33	=	DEEGNA, FFFENA
PClass2_T_o_rango_0_a_33	=	DEEG, FEFF
PClass2_T_rev_rango_0_a_33	=	EGDENA, FFFENA
PClass2_T_rev_o_rango_0_a_33	=	DEEG, FEFF
PClass1_T_rev_rango_33_a_67	=	FGE
PClass2_T_rango_33_a_67	=	EGGFNA, FEEFFG
PClass2_T_o_rango_33_a_67	=	EFFEFG, EGGF
PClass2_T_rev_rango_33_a_67	=	GFEGNA, GFFGEF
PClass2_T_rev_o_rango_33_a_67	=	EFFGGF, EGGF

Cluster 6

Descriptor	Valores
PClass2_T_rev	= CCDCDD, DCCCED, DCDDCD, DCDDNA
PClass2_T_rev_o	= CCDCDD, CCDCED, CDDCDD, DCDD
PClass2_T_rango_50_a_100	= CBBCCD, DCCCNA, DCNANA, DEEDDC
PClass2_T_o_rango_50_a_100	= BCCBCD, CCDC, DC, DCDEED
PClass2_T_rev_rango_50_a_100	= CCDCED, CCDCNA, DCCDBC, DCNANA
PClass2_T_rev_o_rango_50_a_100	= BCCDDC, CCDC, CCDCED, DC
PClass2_T_rango_0_a_33	= DDDCCD, DDDCNA, DDNANA
PClass2_T_o_rango_0_a_33	= CDDCDD, DCDD, DD
PClass2_T_rev_rango_0_a_33	= CDDCDD, DDDCNA, DDNANA
PClass2_T_rev_o_rango_0_a_33	= CDDCDD, DCDD, DD
PClass2_T_rango_33_a_67	= CDDEED, DCCBBC, DDDCCC, DDDCNA
PClass2_T_rev_rango_33_a_67	= BCCBDC, CCDCDD, DCDDNA, EDDECD

Descriptor	Valores
PClass1_T_rango_67_a_100	= CBD, CED, CNULLNULL, DCNULL
PClass2_T_rango_67_a_100	= BCCDDC, CCNANA, DCNANA, EDDCCC
PClass2_T_o_rango_67_a_100	= BCCDDC, CC, CCDCED, DC
PClass2_T_rev_rango_67_a_100	= CCDCED, CCNANA, DCCDBC, DCNANA
PClass2_T_rev_o_rango_67_a_100	= BCCDDC, CC, CCDCED, DC

Cluster 7

Descriptor	Valores
PClass2_T_rev	= CBDCDD, DDBCCB, DDDCCsD
PClass2_T_rev_o	= BCCBDD, CBDCDD, CsDDCDD
PClass2_T_rango_0_a_50	= DDBBBC, DDBCCD, DDCDDC
PClass2_T_rev_rango_50_a_100	= CBBCDC, DCCsDDCs, DDBCCB
PClass2_T_rango_0_a_33	= BBBCCD, BCCDDD, DDCDNA
PClass2_T_o_rango_0_a_33	= BBBCCD, BCCDDD, CDDD
PClass2_T_rev_rango_0_a_33	= CDBCBB, DDCDBC, DDCDNA
PClass2_T_rev_o_rango_0_a_33	= BBBCCD, BCCDDD, CDDD
PClass2_T_rango_33_a_67	= DCDDCB, DDCDDC, DDDCsNA

Cluster 8

Descriptor	Valores
PClass1_T	= AEF, CDE, EFA
PClass2_T	= ABBCCD, CDABBC
PClass2_T_rango_0_a_50	= ABBCCD

Descriptor	Valores
PClass2_T_rev_rango_0_a_50	= DCCDBC, DECDBC, EFDECD
PClass2_T_rev_o_rango_0_a_50	= BCCDDC, BCCDDE, CDDEEF
PClass1_T_rango_33_a_67	= CDE
PClass2_T_rango_33_a_67	= CDDCDE, CDDEEF
PClass2_T_o_rango_33_a_67	= CDDCDE, CDDEEF
PClass2_T_rev_rango_33_a_67	= AGFAEF, CDDEDC, FGEFDE
PClass2_T_rev_o_rango_33_a_67	= AGEFFA, CDDCDE, DEEFFG

Cluster 9

Descriptor	Valores
PClass2_T	= BBBCCD, CDBCAB, GFGBBC
PClass2_T_rev	= BBDCED, CDBCDE, GFDCED
PClass2_T_rev_o	= BBDCED, BCCDDE, DCEDGF
PClass2_T_rango_0_a_50	= BBBCCD, BCABCD, GBBCCD
PClass2_T_rev_rango_0_a_50	= BBCDBC, BCDBCD, GFDGCD
PClass2_T_rev_o_rango_0_a_50	= BBBCCD, BCCDDB, CDDGGF
PClass2_T_rango_33_a_67	= BBBCCD, CDDBBC, GFDGFG
PClass2_T_o_rango_33_a_67	= BBBCCD, BCCDDB, DGFGGF

Cluster 10

Descriptor	Valores
PClass2_T	= AAEAAB, EAAAAC, EAABBD

Descriptor	Valores
PClass2_T_o	= AAABEA, AAACEA, ABBDEA
PClass2_T_rev	= AADCsDD, DCBDAB
PClass2_T_rev_o	= AADCsDD, ABBDDC
PClass1_T_rango_50_a_100	= BAD, BDC, CsDB
PClass2_T_rango_50_a_100	= BAABBD, BCsCsDDD, BDDCNA
PClass2_T_o_rango_50_a_100	= ABBABD, BCsCsDDD, BDDC
PClass2_T_rev_rango_50_a_100	= DCBDAB, DCBDNA, DCsDDCsD
PClass2_T_rev_o_rango_50_a_100	= ABBDDC, BDDC, CsDDCsDD
PClass2_T_rango_33_a_67	= AAABBCs, ABBDNA, ACCBBA
PClass2_T_o_rango_33_a_67	= AAABBCs, ABBD, ACBACB
PClass1_T_rango_67_a_100	= ABD, BDC, CsDNULL
PClass2_T_rango_67_a_100	= ABBDDC, BDDCNA, CsDDDDCs
PClass2_T_o_rango_67_a_100	= ABBDDC, BDDC, CsDDCsDD
PClass2_T_rev_rango_67_a_100	= DCBDAB, DCBDNA, DCsDDCsD
PClass2_T_rev_o_rango_67_a_100	= ABBDDC, BDDC, CsDDCsDD

Cluster 11

Descriptor	Valores
PClass2_T_rev	= AADCED, EEAADC
PClass2_T_rev_o	= AADCED, AADCEE
PClass1_T_rev_rango_33_a_67	= AEF, EAF, EANULL, EFD
PClass2_T_rango_33_a_67	= AAABBC, AAAEEF, ABBCCD, EEAAAE
PClass2_T_o_rango_33_a_67	= AAABBC, AAAEEE, AAAEEF, ABBCCD

Cluster 12

Descriptor		Valores
PClass1_T	=	ABC
PClass1_T_rev	=	ABC
PClass2_T	=	BCCBAA
PClass2_T_o	=	AABCCB
PClass2_T_rev	=	CBBCBA
PClass2_T_rev_o	=	BABCCB
PClass1_T_rango_0_a_50	=	ACB
PClass1_T_rev_rango_0_a_50	=	CAB
PClass2_T_rev_rango_0_a_50	=	CCBCAB
PClass2_T_rev_o_rango_0_a_50	=	ABBCCC
PClass2_T_rango_50_a_100	=	CBBCBA
PClass2_T_rev_rango_50_a_100	=	CBBABC
PClass2_T_rev_o_rango_50_a_100	=	BABCCB
Suavidad_rango_50_a_100	>=	9.025
Suavidad2_rango_50_a_100	>=	11.70167
PClass1_T_rev_rango_0_a_33	=	ACB
PClass2_T_rango_0_a_33	=	AAABBC
PClass2_T_o_rango_0_a_33	=	AAABBC
PClass2_T_rev_rango_0_a_33	=	BCABAA
PClass2_T_rev_o_rango_0_a_33	=	AAABBC
PClass1_T_rango_33_a_67	=	CBNULL
PClass1_T_o_rango_33_a_67	=	CB
PClass1_T_rev_rango_33_a_67	=	CBNULL
PClass1_T_rev_o_rango_33_a_67	=	CB
PClass2_T_rango_33_a_67	=	BCCCCB
PClass2_T_o_rango_33_a_67	=	BCCBCC
PClass2_T_rev_rango_33_a_67	=	BCCBCC
PClass2_T_rev_o_rango_33_a_67	=	BCCBCC
PClass2_T_rango_67_a_100	=	BCCBBA
PClass2_T_o_rango_67_a_100	=	BABCCB
PClass2_T_rev_rango_67_a_100	=	BACBBC
PClass2_T_rev_o_rango_67_a_100	=	BABCCB

Cluster 13

Descriptor	Valores
PClass1_T_rango_0_a_50	= BCNULL, BDC, BDCs, BEGs
PClass2_T_rev_rango_0_a_50	= BCBBCB, CBDCBD, CBDCDD, FEBFCB
PClass2_T_rev_rango_50_a_100	= BCCBDC, CBBABB, CBBADC, CBDCED
PClass1_T_rev_rango_0_a_33	= BCD, BCNULL, DBGs, DCsB
PClass2_T_rev_rango_0_a_33	= BDGsBEGs, CBBCBB, CBDCBD, DDCsDBCs
PClass2_T_rango_33_a_67	= BDDCCB, CBBBBC, CBBFFE, CBDDDC

Cluster 14

Descriptor	Valores
PClass1_T	= CED, EAC, ECD, EFD
PClass2_T	= CCEECE, EEABBC, EECCCE, EEECCA, EEFEEF
PClass2_T_rev	= EECCFE, EEFEGF, FEEEGF
PClass2_T_rev_o	= CCEEFE, EEFEGF
PClass1_T_rev_rango_0_a_50	= ECB, ECD, ECNULL
PClass2_T_rev_rango_0_a_50	= EECCED, EECEBC, EEDCED, EEECNA, EEEDCE
PClass2_T_rev_o_rango_0_a_50	= BCCEEE, CCEDEE, CEEDEE, DCEDEE, ECEE
PClass2_T_rango_50_a_100	= CAABBA, DCCBBA, EEECCA
PClass2_T_o_rango_50_a_100	= ABBACA, BACBDC, CAECEE
PClass1_T_rango_33_a_67	= EAC, ECA, EDC
PClass1_T_rev_rango_33_a_67	= EAB, EAC
PClass1_T_rev_o_rango_33_a_67	= CEA, EAB
PClass2_T_rango_33_a_67	= EEECCA, EEEDDC
PClass2_T_o_rango_33_a_67	= CAECEE, DCEDEE

Descriptor	Valores
PClass2_T_rev_rango_33_a_67	= BACBDC, EEBAAB, EEBACB, EECAEC
PClass2_T_rev_o_rango_33_a_67	= ABBAEE, BACBDC, BACBEE, CAECEE

Cluster 15

Descriptor	Valores
PClass1_T	= CBA, CBD
PClass2_T	= CCBACB, CCCBBA, CCCBBD
PClass2_T_o	= BACBCC, BDCBCC
PClass2_T_rev	= BACBCC, BACCFE, CCBAAG, CCBAFE, CCCBFE, CCFEGF
PClass2_T_rev_o	= AGBACC, BACBCC, BACCFE, CBCCFE, CCFEGF
PClass_C_rango_0_a_50	>= 0.4642857
PClass2_T_rango_0_a_50	= CCCBBB, CCCBBD, CCCBNA, CCNANA
PClass2_T_o_rango_0_a_50	= BBCBCC, BDCBCC, CBCC, CC
PClass2_T_rev_rango_0_a_50	= CCACBA, CCBABB, CCCBNA, CCDCBD, CCNANA
PClass1_T_rango_0_a_33	= CBD, CBNULL, CNULLNULL
PClass2_T_rango_0_a_33	= CCCBBB, CCCBBD, CCNANA
PClass2_T_o_rango_0_a_33	= BBCBCC, BDCBCC, CC
PClass2_T_rev_rango_0_a_33	= CCBBCB, CCBDCB, CCNANA
PClass2_T_rev_o_rango_0_a_33	= BBCBCC, BDCBCC, CC

Cluster 16

Descriptor	Valores
PClass2_T_rev_rango_0_a_50	= BCCBDC, DCBDCC

Descriptor	Valores
PClass2_T_rev_rango_0_a_33 =	BDCBDC, DCEDFE

Cluster 17

Descriptor	Valores
PClass2_T =	ABBAAG, ABBAFG, BBEEEA, BCCAAG, GAAAAG, GFsABBB
PClass2_T_o =	AAAGGA, ABAGBA, ABBAFG, ABBBGFs, AGBCCA, BBEAEE
PClass2_T_rev =	AGAAGA, AGBAAB, AGCABC, BAABAG, BBAGCA, GFsFsGGG
PClass2_T_rev_o =	AAAGGA, ABAGBA, AGBBCA, AGBCCA, FsGGFsGG
PClass2_T_rango_0_a_50 =	ABBANA, ABBBBA, BCCANA, EEEAAB, FGGAAB, GAAANA
PClass2_T_o_rango_0_a_50 =	AAGA, ABBA, ABBABB, ABEAEE, ABFGGA, BCCA
PClass2_T_rev_rango_0_a_50 =	AAGANA, AGBABB, BAABGA, BAABNA, BBABEA, CABCNA
PClass2_T_rev_o_rango_0_a_50 =	AAGA, ABBA, ABBAGA, ABBBEA, AGBABB, BCCA
PClass2_T_rev_rango_0_a_33 =	ABEAEE, ABGAFG, ABNANA, BABBAB, BCNANA, GANANA
PClass2_T_rev_o_rango_0_a_33 =	AB, ABBABB, ABEAEE, ABFGGA, BC, GA
PClass2_T_rango_33_a_67 =	ABBAAG, ABBANA, BAAGGG, BBABBC, BCCAAG, GAAAAG

Cluster 18

Descriptor	Valores
PClass1_T =	AFG, EAG, FAG

Descriptor	Valores
PClass1_T_o	= EGA, FGA
PClass2_T_rango_0_a_50	= AAFAAG, AAFFFA, EEAAEA, FAAGGF, FFFAAA
PClass2_T_o_rango_0_a_50	= AAAGFA, AAEAEE, AAFAFF, AGFAGF
PClass2_T_rev_rango_0_a_50	= AAAGFA, AAEEEA, AAFAFF, FFAAFA, GAFGGF
PClass2_T_rev_o_rango_0_a_50	= AAAGFA, AAEAEE, AAFAFF, FGGAGF
PClass1_T_rev_rango_0_a_33	= AFNULL, EANULL, FNULLNULL, GFA
PClass2_T_rango_0_a_33	= AAFANA, EEEANA, FAAGGF, FFFAAA, FFNANA
PClass2_T_o_rango_0_a_33	= AAFA, AAFAFF, AGFAGF, EAEE, FF
PClass2_T_rev_rango_0_a_33	= AAFAFF, AAFANA, EEEANA, FFNANA, FGGFAG
PClass2_T_rev_o_rango_0_a_33	= AAFA, AAFAFF, AGFGGF, EAEE, FF
PClass1_T_rango_33_a_67	= AEG, AFG, AGNULL
PClass1_T_rev_rango_33_a_67	= ABG, AFG, AGE, AGNULL
PClass2_T_rango_33_a_67	= AAAGGA, AAAGNA, AAEAAG, AAFFFA, FGGAAB

Cluster 19

Descriptor	Valores
PClass1_T	= ECB
PClass2_T	= ECCBBA, EEECCB
PClass2_T_o	= BACBEC, CBECEE
m_0_a_8	>= 0.1666667
PClass1_T_rango_0_a_50	= ECB
PClass1_T_o_rango_0_a_50	= CEB
PClass2_T_rango_0_a_50	= ECCBBA, EEECCB
PClass2_T_o_rango_0_a_50	= BACBEC, CBECEE

Descriptor	Valores
PClass2_T_rev_rango_0_a_50	= BACBEC
PClass2_T_rev_o_rango_0_a_50	= BACBEC
m_0_a_8_rango_0_a_50	>= 0.25
PClass1_T_rango_0_a_33	= ECB
PClass1_T_o_rango_0_a_33	= CEB
PClass1_T_rev_rango_0_a_33	= BCE, EBC
PClass2_T_rango_0_a_33	= ECCBNA, EEECCB
PClass2_T_o_rango_0_a_33	= CBEC, CBECEE
PClass2_T_rev_rango_0_a_33	= CBECEE, CBECNA
PClass2_T_rev_o_rango_0_a_33	= CBEC, CBECEE
m_0_a_8_rango_0_a_33	>= 0.25
PClass1_T_rev_rango_33_a_67	= FGA, GAB
PClass2_T_rango_33_a_67	= CBBAAG
PClass2_T_o_rango_33_a_67	= AGBACB
Pendiente_rango_33_a_67	< -1.208333
PendienteRegresion_rango_33_a_67	< -1.6

Cluster 20

Descriptor	Valores
PClass2_T	= AAGAAB, BBFGGA, GAABBC
PClass2_T_o	= AAABGA, ABBCGA, BBFGGA
PClass1_T_rango_0_a_50	= AGB, BFG, BGA
PClass2_T_rango_0_a_50	= AAGAAB, BBFGGA, GAABBC
PClass2_T_o_rango_0_a_50	= AAABGA, ABBCGA, BBFGGA
PClass1_T_rango_0_a_33	= AGB, BFG, GAB
PClass1_T_o_rango_0_a_33	= FGB, GAB
PClass2_T_rev_rango_0_a_33	= AAABGA, BBABGA, BCABGA
PClass2_T_rev_o_rango_0_a_33	= AAABGA, ABBBGA, ABBCGA
PClass2_T_rango_33_a_67	= ABBCCB, BBBCCB, BCCBBA

C.3 Arcos por grupo del fandango de Huelva (algoritmo pair)

Arcos comunes a dos o más clusters

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m27.m22	8	9	[8,8,8]	[8,7,5,3,1,0]
m12.m22	5	9	[7,10,8]	[8,7,5,3,1,0]
m07.m22	15	8	[8,8]	[8,7,5,3,1,0]
m27.m08	15	8	[8,8,8]	[7,5,3,1,0]
m00.m22	7	8	[10,8]	[8,7,5,3,1,0]
m09.m08	6	8	[12,10,8]	[7,5,3,1,0]
m12.m08	5	8	[7,10,8]	[7,5,3,1,0]
m20.m22	3	8	[13,12]	[8,7,5,3,1,0]
m41.m15	3	8	[7,10,8,7,10,8]	[7,5]
m07.m08	27	7	[8,8]	[7,5,3,1,0]
m27.m26	15	7	[8,8,8]	[5,3,1,0]
m00.m08	11	7	[10,8]	[7,5,3,1,0]
m04.m08	6	7	[7,8]	[7,5,3,1,0]
m09.m26	6	7	[12,10,8]	[5,3,1,0]
m12.m26	5	7	[7,10,8]	[5,3,1,0]
m14.m26	4	7	[5,7,8]	[5,3,1,0]
m20.m08	3	7	[13,12]	[7,5,3,1,0]
m45.m13	3	7	[7,8,10]	[13,12,10,8]
m45.m73	3	7	[7,8,10]	[13,15,13,12]
m04.m49	2	7	[7,8]	[10,13,12,10,8]
m11.m08	2	7	[1,3]	[7,5,3,1,0]
m14.m58	2	7	[5,7,8]	[10,8,10,12]
m75.m01	2	7	[5,7,7,7,8]	[5,3]
m07.m26	27	6	[8,8]	[5,3,1,0]
m27.m05	17	6	[8,8,8]	[7,5,3]
m00.m26	11	6	[10,8]	[5,3,1,0]
m06.m26	11	6	[5,5]	[5,3,1,0]
m38.m26	9	6	[5,7]	[5,3,1,0]
m04.m26	7	6	[7,8]	[5,3,1,0]
m09.m05	6	6	[12,10,8]	[7,5,3]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m12.m05	5	6	[7,10,8]	[7,5,3]
m04.m13	4	6	[7,8]	[13,12,10,8]
m25.m03	4	6	[5,7,5,3]	[1,0]
m45.m52	4	6	[7,8,10]	[15,13,12]
m02.m17	3	6	[12,12]	[12,12,12,12]
m04.m58	3	6	[7,8]	[10,8,10,12]
m04.m73	3	6	[7,8]	[13,15,13,12]
m12.m12	3	6	[7,10,8]	[7,10,8]
m13.m04	3	6	[13,12,10,8]	[7,8]
m17.m02	3	6	[12,12,12,12]	[12,12]
m20.m26	3	6	[13,12]	[5,3,1,0]
m21.m09	3	6	[7,7,8]	[12,10,8]
m28.m26	3	6	[0,0]	[5,3,1,0]
m37.m04	3	6	[10,10,10,8]	[7,8]
m45.m09	3	6	[7,8,10]	[12,10,8]
m11.m25	2	6	[1,3]	[5,7,5,3]
m11.m26	2	6	[1,3]	[5,3,1,0]
m33.m13	2	6	[13,15]	[13,12,10,8]
m37.m00	2	6	[10,10,10,8]	[10,8]
m38.m58	2	6	[5,7]	[10,8,10,12]
m73.m00	2	6	[13,15,13,12]	[10,8]
m82.m15	2	6	[5,5,7,8]	[7,5]
m07.m05	30	5	[8,8]	[7,5,3]
m27.m15	25	5	[8,8,8]	[7,5]
m05.m03	22	5	[7,5,3]	[1,0]
m27.m01	17	5	[8,8,8]	[5,3]
m27.m03	15	5	[8,8,8]	[1,0]
m00.m05	11	5	[10,8]	[7,5,3]
m02.m09	7	5	[12,12]	[12,10,8]
m12.m15	7	5	[7,10,8]	[7,5]
m45.m00	7	5	[7,8,10]	[10,8]
m45.m20	7	5	[7,8,10]	[13,12]
m04.m05	6	5	[7,8]	[7,5,3]
m04.m09	6	5	[7,8]	[12,10,8]
m09.m01	6	5	[12,10,8]	[5,3]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m09.m03	6	5	[12,10,8]	[1,0]
m09.m15	6	5	[12,10,8]	[7,5]
m14.m20	6	5	[5,7,8]	[13,12]
m38.m05	6	5	[5,7]	[7,5,3]
m12.m01	5	5	[7,10,8]	[5,3]
m12.m03	5	5	[7,10,8]	[1,0]
m14.m19	5	5	[5,7,8]	[10,12]
m15.m05	5	5	[7,5]	[7,5,3]
m18.m00	5	5	[0,5,5]	[10,8]
m00.m12	4	5	[10,8]	[7,10,8]
m04.m52	4	5	[7,8]	[15,13,12]
m06.m14	4	5	[5,5]	[5,7,8]
m10.m00	4	5	[10,10,8]	[10,8]
m10.m04	4	5	[10,10,8]	[7,8]
m12.m00	4	5	[7,10,8]	[10,8]
m14.m00	4	5	[5,7,8]	[10,8]
m14.m01	4	5	[5,7,8]	[5,3]
m14.m03	4	5	[5,7,8]	[1,0]
m14.m15	4	5	[5,7,8]	[7,5]
m19.m52	4	5	[10,12]	[15,13,12]
m21.m00	4	5	[7,7,8]	[10,8]
m09.m04	3	5	[12,10,8]	[7,8]
m11.m05	3	5	[1,3]	[7,5,3]
m20.m05	3	5	[13,12]	[7,5,3]
m27.m00	3	5	[8,8,8]	[10,8]
m45.m19	3	5	[7,8,10]	[10,12]
m45.m33	3	5	[7,8,10]	[13,15]
m00.m09	2	5	[10,8]	[12,10,8]
m12.m04	2	5	[7,10,8]	[7,8]
m14.m04	2	5	[5,7,8]	[7,8]
m19.m50	2	5	[10,12]	[12,10,9]
m20.m09	2	5	[13,12]	[12,10,8]
m21.m01	2	5	[7,7,8]	[5,3]
m21.m04	2	5	[7,7,8]	[7,8]
m33.m09	2	5	[13,15]	[12,10,8]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m38.m21	2	5	[5,7]	[7,7,8]
m45.m04	2	5	[7,8,10]	[7,8]
m52.m00	2	5	[15,13,12]	[10,8]
m07.m15	43	4	[8,8]	[7,5]
m07.m01	30	4	[8,8]	[5,3]
m01.m03	29	4	[5,3]	[1,0]
m07.m03	27	4	[8,8]	[1,0]
m15.m03	25	4	[7,5]	[1,0]
m02.m02	21	4	[12,12]	[12,12]
m38.m01	19	4	[5,7]	[5,3]
m02.m15	16	4	[12,12]	[7,5]
m00.m15	14	4	[10,8]	[7,5]
m02.m00	14	4	[12,12]	[10,8]
m02.m16	14	4	[12,12]	[10,9]
m06.m00	14	4	[5,5]	[10,8]
m06.m03	14	4	[5,5]	[1,0]
m04.m00	13	4	[7,8]	[10,8]
m06.m01	13	4	[5,5]	[5,3]
m20.m00	12	4	[13,12]	[10,8]
m00.m01	11	4	[10,8]	[5,3]
m00.m03	11	4	[10,8]	[1,0]
m00.m04	9	4	[10,8]	[7,8]
m04.m01	9	4	[7,8]	[5,3]
m04.m15	9	4	[7,8]	[7,5]
m06.m04	9	4	[5,5]	[7,8]
m38.m03	9	4	[5,7]	[1,0]
m00.m00	8	4	[10,8]	[10,8]
m04.m20	8	4	[7,8]	[13,12]
m04.m03	7	4	[7,8]	[1,0]
m07.m00	7	4	[8,8]	[10,8]
m38.m15	7	4	[5,7]	[7,5]
m04.m19	6	4	[7,8]	[10,12]
m06.m06	6	4	[5,5]	[5,5]
m06.m38	6	4	[5,5]	[5,7]
m11.m03	6	4	[1,3]	[1,0]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m28.m03	6	4	[0,0]	[1,0]
m38.m00	6	4	[5,7]	[10,8]
m38.m20	6	4	[5,7]	[13,12]
m15.m01	5	4	[7,5]	[5,3]
m15.m15	5	4	[7,5]	[7,5]
m19.m20	5	4	[10,12]	[13,12]
m38.m19	5	4	[5,7]	[10,12]
m04.m04	4	4	[7,8]	[7,8]
m06.m15	4	4	[5,5]	[7,5]
m11.m15	4	4	[1,3]	[7,5]
m11.m38	4	4	[1,3]	[5,7]
m28.m01	4	4	[0,0]	[5,3]
m33.m20	4	4	[13,15]	[13,12]
m38.m04	4	4	[5,7]	[7,8]
m00.m07	3	4	[10,8]	[8,8]
m00.m19	3	4	[10,8]	[10,12]
m04.m33	3	4	[7,8]	[13,15]
m11.m01	3	4	[1,3]	[5,3]
m20.m01	3	4	[13,12]	[5,3]
m20.m03	3	4	[13,12]	[1,0]
m20.m04	3	4	[13,12]	[7,8]
m20.m15	3	4	[13,12]	[7,5]
m00.m20	2	4	[10,8]	[13,12]
m19.m16	2	4	[10,12]	[10,9]
m33.m00	2	4	[13,15]	[10,8]
m39.m15	2	4	[7,9]	[7,5]
m39.m16	2	4	[7,9]	[10,9]

Arcos específicos del cluster 1

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m46.m36	2	4	[-12,-7]	[-5,-4]

Arcos específicos del cluster 2

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m03.m48	1	7	[1,0]	[0,1,3,1,0]
m28.m48	1	7	[0,0]	[0,1,3,1,0]
m36.m48	1	7	[-5,-4]	[0,1,3,1,0]
m28.m34	2	6	[0,0]	[1,3,1,0]
m03.m34	1	6	[1,0]	[1,3,1,0]
m36.m34	1	6	[-5,-4]	[1,3,1,0]
m61.m29	3	5	[1,3,4]	[1,-1]
m61.m11	1	5	[1,3,4]	[1,3]
m11.m29	4	4	[1,3]	[1,-1]
m28.m11	2	4	[0,0]	[1,3]
m03.m03	1	4	[1,0]	[1,0]
m03.m11	1	4	[1,0]	[1,3]
m11.m11	1	4	[1,3]	[1,3]
m28.m28	1	4	[0,0]	[0,0]
m29.m11	1	4	[1,-1]	[1,3]
m29.m29	1	4	[1,-1]	[1,-1]
m36.m03	1	4	[-5,-4]	[1,0]
m36.m11	1	4	[-5,-4]	[1,3]
m47.m47	1	4	[-2,0]	[-2,0]

Arcos específicos del cluster 3

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m60.m31	1	9	[10,10,8,7,12,12]	[10,10,9]
m60.m16	2	8	[10,10,8,7,12,12]	[10,9]
m10.m57	2	6	[10,10,8]	[7,12,12]
m10.m10	1	6	[10,10,8]	[10,10,8]
m10.m31	1	6	[10,10,8]	[10,10,9]
m10.m50	1	6	[10,10,8]	[12,10,9]
m57.m31	1	6	[7,12,12]	[10,10,9]
m10.m02	3	5	[10,10,8]	[12,12]
m00.m57	2	5	[10,8]	[7,12,12]
m10.m16	2	5	[10,10,8]	[10,9]
m57.m16	2	5	[7,12,12]	[10,9]
m00.m10	1	5	[10,8]	[10,10,8]
m00.m31	1	5	[10,8]	[10,10,9]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m00.m50	1	5	[10,8]	[12,10,9]
m02.m10	1	5	[12,12]	[10,10,8]
m02.m31	1	5	[12,12]	[10,10,9]
m00.m02	3	4	[10,8]	[12,12]
m00.m16	2	4	[10,8]	[10,9]

Arcos específicos del cluster 4

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m44.m63	2	9	[12,12,12,12,12,12]	[10,9,5]
m44.m16	2	8	[12,12,12,12,12,12]	[10,9]
m17.m63	6	7	[12,12,12,12]	[10,9,5]
m17.m50	4	7	[12,12,12,12]	[12,10,9]
m17.m16	6	6	[12,12,12,12]	[10,9]
m02.m63	10	5	[12,12]	[10,9,5]
m02.m50	9	5	[12,12]	[12,10,9]
m39.m50	1	5	[7,9]	[12,10,9]
m50.m15	1	5	[12,10,9]	[7,5]
m39.m02	2	4	[7,9]	[12,12]
m16.m15	1	4	[10,9]	[7,5]
m19.m02	1	4	[10,12]	[12,12]
m19.m15	1	4	[10,12]	[7,5]
m39.m19	1	4	[7,9]	[10,12]

Arcos específicos del cluster 5

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m51.m72	3	8	[13,13]	[13,15,13,12,10,8]
m20.m72	1	8	[13,12]	[13,15,13,12,10,8]
m51.m13	3	6	[13,13]	[13,12,10,8]
m51.m73	3	6	[13,13]	[13,15,13,12]
m20.m13	1	6	[13,12]	[13,12,10,8]
m20.m73	1	6	[13,12]	[13,15,13,12]
m51.m09	3	5	[13,13]	[12,10,8]
m51.m52	3	5	[13,13]	[15,13,12]
m20.m52	1	5	[13,12]	[15,13,12]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m51.m20	5	4	[13,13]	[13,12]
m51.m00	3	4	[13,13]	[10,8]
m51.m33	3	4	[13,13]	[13,15]
m20.m20	1	4	[13,12]	[13,12]
m20.m33	1	4	[13,12]	[13,15]
m51.m51	1	4	[13,13]	[13,13]

Arcos específicos del cluster 6

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m74.m00	1	8	[10,10,10,8,7,8]	[10,8]
m37.m45	1	7	[10,10,10,8]	[7,8,10]
m10.m09	1	6	[10,10,8]	[12,10,8]
m10.m45	1	6	[10,10,8]	[7,8,10]
m58.m00	1	6	[10,8,10,12]	[10,8]
m58.m07	1	6	[10,8,10,12]	[8,8]
m00.m45	1	5	[10,8]	[7,8,10]
m10.m07	1	5	[10,10,8]	[8,8]
m10.m19	1	5	[10,10,8]	[10,12]
m19.m00	1	4	[10,12]	[10,8]
m19.m07	1	4	[10,12]	[8,8]

Arcos específicos del cluster 7

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m21.m74	1	9	[7,7,8]	[10,10,10,8,7,8]
m45.m74	1	9	[7,8,10]	[10,10,10,8,7,8]
m45.m70	2	8	[7,8,10]	[10,10,9,10,8]
m04.m74	1	8	[7,8]	[10,10,10,8,7,8]
m04.m70	2	7	[7,8]	[10,10,9,10,8]
m21.m37	1	7	[7,7,8]	[10,10,10,8]
m37.m12	1	7	[10,10,10,8]	[7,10,8]
m45.m37	1	7	[7,8,10]	[10,10,10,8]
m45.m31	2	6	[7,8,10]	[10,10,9]
m04.m37	1	6	[7,8]	[10,10,10,8]
m10.m12	1	6	[10,10,8]	[7,10,8]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m21.m10	1	6	[7,7,8]	[10,10,8]
m45.m10	1	6	[7,8,10]	[10,10,8]
m04.m31	2	5	[7,8]	[10,10,9]
m31.m00	2	5	[10,10,9]	[10,8]
m45.m16	2	5	[7,8,10]	[10,9]
m04.m10	1	5	[7,8]	[10,10,8]
m04.m16	2	4	[7,8]	[10,9]
m16.m00	2	4	[10,9]	[10,8]

Arcos específicos del cluster 8

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m42.m73	2	9	[5,7,8,10,12]	[13,15,13,12]
m42.m52	3	8	[5,7,8,10,12]	[15,13,12]
m42.m20	3	7	[5,7,8,10,12]	[13,12]
m14.m73	2	7	[5,7,8]	[13,15,13,12]
m42.m33	2	7	[5,7,8,10,12]	[13,15]
m14.m52	3	6	[5,7,8]	[15,13,12]
m19.m73	2	6	[10,12]	[13,15,13,12]
m38.m73	2	6	[5,7]	[13,15,13,12]
m58.m20	1	6	[10,8,10,12]	[13,12]
m38.m52	3	5	[5,7]	[15,13,12]
m14.m33	2	5	[5,7,8]	[13,15]
m19.m33	2	4	[10,12]	[13,15]
m38.m33	2	4	[5,7]	[13,15]

Arcos específicos del cluster 9

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m45.m72	1	9	[7,8,10]	[13,15,13,12,10,8]
m04.m72	1	8	[7,8]	[13,15,13,12,10,8]
m21.m49	1	8	[7,7,8]	[10,13,12,10,8]
m21.m13	1	7	[7,7,8]	[13,12,10,8]
m45.m58	1	7	[7,8,10]	[10,8,10,12]
m14.m45	1	6	[5,7,8]	[7,8,10]
m45.m45	1	6	[7,8,10]	[7,8,10]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m04.m45	1	5	[7,8]	[7,8,10]
m21.m20	1	5	[7,7,8]	[13,12]
m38.m45	1	5	[5,7]	[7,8,10]

Arcos específicos del cluster 10

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m23.m31	1	7	[0,5,5,5]	[10,10,9]
m18.m12	1	6	[0,5,5]	[7,10,8]
m18.m31	1	6	[0,5,5]	[10,10,9]
m23.m16	1	6	[0,5,5,5]	[10,9]
m23.m39	1	6	[0,5,5,5]	[7,9]
m06.m31	2	5	[5,5]	[10,10,9]
m18.m38	2	5	[0,5,5]	[5,7]
m06.m12	1	5	[5,5]	[7,10,8]
m15.m12	1	5	[7,5]	[7,10,8]
m18.m15	1	5	[0,5,5]	[7,5]
m18.m16	1	5	[0,5,5]	[10,9]
m18.m39	1	5	[0,5,5]	[7,9]
m38.m31	1	5	[5,7]	[10,10,9]
m39.m31	1	5	[7,9]	[10,10,9]
m06.m16	2	4	[5,5]	[10,9]
m06.m39	2	4	[5,5]	[7,9]
m15.m00	1	4	[7,5]	[10,8]
m38.m16	1	4	[5,7]	[10,9]

Arcos específicos del cluster 11

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m23.m32	2	9	[0,5,5,5]	[12,13,12,10,8]
m23.m35	2	9	[0,5,5,5]	[12,12,12,10,8]
m82.m49	1	9	[5,5,7,8]	[10,13,12,10,8]
m18.m32	2	8	[0,5,5]	[12,13,12,10,8]
m18.m35	2	8	[0,5,5]	[12,12,12,10,8]
m23.m13	2	8	[0,5,5,5]	[13,12,10,8]
m82.m13	2	8	[5,5,7,8]	[13,12,10,8]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m14.m49	1	8	[5,7,8]	[10,13,12,10,8]
m06.m32	4	7	[5,5]	[12,13,12,10,8]
m06.m35	4	7	[5,5]	[12,12,12,10,8]
m23.m09	4	7	[0,5,5,5]	[12,10,8]
m06.m49	2	7	[5,5]	[10,13,12,10,8]
m14.m13	2	7	[5,7,8]	[13,12,10,8]
m18.m13	2	7	[0,5,5]	[13,12,10,8]
m82.m09	2	7	[5,5,7,8]	[12,10,8]
m38.m49	1	7	[5,7]	[10,13,12,10,8]
m06.m13	9	6	[5,5]	[13,12,10,8]
m18.m09	4	6	[0,5,5]	[12,10,8]
m23.m00	4	6	[0,5,5,5]	[10,8]
m23.m02	4	6	[0,5,5,5]	[12,12]
m14.m09	2	6	[5,7,8]	[12,10,8]
m23.m20	2	6	[0,5,5,5]	[13,12]
m38.m13	2	6	[5,7]	[13,12,10,8]
m82.m00	2	6	[5,5,7,8]	[10,8]
m82.m20	2	6	[5,5,7,8]	[13,12]
m06.m82	1	6	[5,5]	[5,5,7,8]
m06.m09	13	5	[5,5]	[12,10,8]
m18.m02	4	5	[0,5,5]	[12,12]
m06.m45	2	5	[5,5]	[7,8,10]
m18.m20	2	5	[0,5,5]	[13,12]
m38.m09	2	5	[5,7]	[12,10,8]
m06.m20	9	4	[5,5]	[13,12]
m06.m02	8	4	[5,5]	[12,12]

Arcos específicos del cluster 12

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m82.m04	1	6	[5,5,7,8]	[7,8]
m06.m07	1	4	[5,5]	[8,8]
m07.m04	1	4	[8,8]	[7,8]
m38.m07	1	4	[5,7]	[8,8]

Arcos específicos del cluster 13

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m12.m13	1	7	[7,10,8]	[13,12,10,8]
m00.m13	1	6	[10,8]	[13,12,10,8]
m10.m40	1	6	[10,10,8]	[7,7,5]
m12.m09	1	6	[7,10,8]	[12,10,8]
m21.m21	1	6	[7,7,8]	[7,7,8]
m00.m40	1	5	[10,8]	[7,7,5]
m04.m21	1	5	[7,8]	[7,7,8]
m04.m40	1	5	[7,8]	[7,7,5]
m10.m15	1	5	[10,10,8]	[7,5]
m12.m20	1	5	[7,10,8]	[13,12]
m39.m10	1	5	[7,9]	[10,10,8]
m39.m40	1	5	[7,9]	[7,7,5]
m39.m00	1	4	[7,9]	[10,8]
m39.m04	1	4	[7,9]	[7,8]

Arcos específicos del cluster 14

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m67.m08	2	13	[8,8,8,12,12,12,10,8]	[7,5,3,1,0]
m67.m26	2	12	[8,8,8,12,12,12,10,8]	[5,3,1,0]
m44.m56	1	12	[12,12,12,12,12,12]	[5,7,5,3,1,0]
m67.m05	2	11	[8,8,8,12,12,12,10,8]	[7,5,3]
m17.m59	1	11	[12,12,12,12]	[10,8,7,5,3,1,0]
m44.m08	1	11	[12,12,12,12,12,12]	[7,5,3,1,0]
m17.m56	3	10	[12,12,12,12]	[5,7,5,3,1,0]
m35.m08	3	10	[12,12,12,10,8]	[7,5,3,1,0]
m27.m59	2	10	[8,8,8]	[10,8,7,5,3,1,0]
m67.m01	2	10	[8,8,8,12,12,12,10,8]	[5,3]
m67.m03	2	10	[8,8,8,12,12,12,10,8]	[1,0]
m67.m15	2	10	[8,8,8,12,12,12,10,8]	[7,5]
m17.m22	1	10	[12,12,12,12]	[8,7,5,3,1,0]
m44.m25	1	10	[12,12,12,12,12,12]	[5,7,5,3]
m44.m26	1	10	[12,12,12,12,12,12]	[5,3,1,0]
m02.m59	9	9	[12,12]	[10,8,7,5,3,1,0]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m07.m59	5	9	[8,8]	[10,8,7,5,3,1,0]
m17.m08	4	9	[12,12,12,12]	[7,5,3,1,0]
m35.m26	3	9	[12,12,12,10,8]	[5,3,1,0]
m20.m59	1	9	[13,12]	[10,8,7,5,3,1,0]
m44.m05	1	9	[12,12,12,12,12,12]	[7,5,3]
m02.m22	9	8	[12,12]	[8,7,5,3,1,0]
m02.m56	5	8	[12,12]	[5,7,5,3,1,0]
m17.m26	5	8	[12,12,12,12]	[5,3,1,0]
m17.m25	3	8	[12,12,12,12]	[5,7,5,3]
m35.m05	3	8	[12,12,12,10,8]	[7,5,3]
m27.m35	2	8	[8,8,8]	[12,12,12,10,8]
m44.m01	1	8	[12,12,12,12,12,12]	[5,3]
m44.m03	1	8	[12,12,12,12,12,12]	[1,0]
m44.m15	1	8	[12,12,12,12,12,12]	[7,5]
m44.m38	1	8	[12,12,12,12,12,12]	[5,7]
m02.m08	14	7	[12,12]	[7,5,3,1,0]
m07.m35	5	7	[8,8]	[12,12,12,10,8]
m17.m05	4	7	[12,12,12,12]	[7,5,3]
m35.m01	3	7	[12,12,12,10,8]	[5,3]
m35.m03	3	7	[12,12,12,10,8]	[1,0]
m35.m15	3	7	[12,12,12,10,8]	[7,5]
m14.m17	1	7	[5,7,8]	[12,12,12,12]
m02.m26	17	6	[12,12]	[5,3,1,0]
m02.m25	5	6	[12,12]	[5,7,5,3]
m17.m01	5	6	[12,12,12,12]	[5,3]
m17.m03	5	6	[12,12,12,12]	[1,0]
m17.m15	4	6	[12,12,12,12]	[7,5]
m17.m38	3	6	[12,12,12,12]	[5,7]
m27.m09	2	6	[8,8,8]	[12,10,8]
m04.m17	1	6	[7,8]	[12,12,12,12]
m07.m17	1	6	[8,8]	[12,12,12,12]
m17.m00	1	6	[12,12,12,12]	[10,8]
m38.m17	1	6	[5,7]	[12,12,12,12]
m02.m05	14	5	[12,12]	[7,5,3]
m07.m09	5	5	[8,8]	[12,10,8]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m27.m02	4	5	[8,8,8]	[12,12]
m14.m02	3	5	[5,7,8]	[12,12]
m02.m01	17	4	[12,12]	[5,3]
m02.m03	17	4	[12,12]	[1,0]
m07.m02	11	4	[8,8]	[12,12]
m02.m38	5	4	[12,12]	[5,7]
m04.m02	3	4	[7,8]	[12,12]
m38.m02	3	4	[5,7]	[12,12]
m02.m20	1	4	[12,12]	[13,12]

Arcos específicos del cluster 15

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m66.m56	1	12	[8,8,8,8,8,8]	[5,7,5,3,1,0]
m68.m22	1	12	[8,8,8,7,7,5]	[8,7,5,3,1,0]
m66.m08	2	11	[8,8,8,8,8,8]	[7,5,3,1,0]
m68.m08	2	11	[8,8,8,7,7,5]	[7,5,3,1,0]
m66.m62	1	11	[8,8,8,8,8,8]	[7,5,7,5,3]
m24.m56	3	10	[8,8,8,8]	[5,7,5,3,1,0]
m24.m22	2	10	[8,8,8,8]	[8,7,5,3,1,0]
m66.m26	2	10	[8,8,8,8,8,8]	[5,3,1,0]
m68.m26	2	10	[8,8,8,7,7,5]	[5,3,1,0]
m66.m25	1	10	[8,8,8,8,8,8]	[5,7,5,3]
m24.m08	7	9	[8,8,8,8]	[7,5,3,1,0]
m27.m56	5	9	[8,8,8]	[5,7,5,3,1,0]
m24.m62	4	9	[8,8,8,8]	[7,5,7,5,3]
m66.m05	2	9	[8,8,8,8,8,8]	[7,5,3]
m68.m05	2	9	[8,8,8,7,7,5]	[7,5,3]
m40.m22	1	9	[7,7,5]	[8,7,5,3,1,0]
m07.m56	7	8	[8,8]	[5,7,5,3,1,0]
m24.m26	7	8	[8,8,8,8]	[5,3,1,0]
m27.m62	7	8	[8,8,8]	[7,5,7,5,3]
m24.m25	4	8	[8,8,8,8]	[5,7,5,3]
m66.m15	3	8	[8,8,8,8,8,8]	[7,5]
m40.m08	2	8	[7,7,5]	[7,5,3,1,0]
m66.m01	2	8	[8,8,8,8,8,8]	[5,3]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m66.m03	2	8	[8,8,8,8,8,8]	[1,0]
m68.m01	2	8	[8,8,8,7,7,5]	[5,3]
m68.m03	2	8	[8,8,8,7,7,5]	[1,0]
m68.m15	2	8	[8,8,8,7,7,5]	[7,5]
m15.m22	1	8	[7,5]	[8,7,5,3,1,0]
m66.m38	1	8	[8,8,8,8,8,8]	[5,7]
m07.m62	10	7	[8,8]	[7,5,7,5,3]
m24.m05	8	7	[8,8,8,8]	[7,5,3]
m27.m25	7	7	[8,8,8]	[5,7,5,3]
m15.m08	3	7	[7,5]	[7,5,3,1,0]
m40.m26	2	7	[7,7,5]	[5,3,1,0]
m62.m03	2	7	[7,5,7,5,3]	[1,0]
m24.m15	12	6	[8,8,8,8]	[7,5]
m07.m25	10	6	[8,8]	[5,7,5,3]
m24.m01	8	6	[8,8,8,8]	[5,3]
m24.m03	7	6	[8,8,8,8]	[1,0]
m24.m38	4	6	[8,8,8,8]	[5,7]
m15.m26	3	6	[7,5]	[5,3,1,0]
m07.m24	2	6	[8,8]	[8,8,8,8]
m24.m07	2	6	[8,8,8,8]	[8,8]
m27.m27	2	6	[8,8,8]	[8,8,8]
m27.m40	2	6	[8,8,8]	[7,7,5]
m40.m05	2	6	[7,7,5]	[7,5,3]
m27.m12	1	6	[8,8,8]	[7,10,8]
m27.m07	7	5	[8,8,8]	[8,8]
m27.m38	7	5	[8,8,8]	[5,7]
m07.m27	6	5	[8,8]	[8,8,8]
m07.m40	4	5	[8,8]	[7,7,5]
m07.m12	2	5	[8,8]	[7,10,8]
m40.m01	2	5	[7,7,5]	[5,3]
m40.m03	2	5	[7,7,5]	[1,0]
m40.m15	2	5	[7,7,5]	[7,5]
m07.m07	16	4	[8,8]	[8,8]
m07.m38	10	4	[8,8]	[5,7]

Arcos específicos del cluster 16

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m41.m22	2	12	[7,10,8,7,10,8]	[8,7,5,3,1,0]
m71.m08	2	12	[12,13,12,10,8,7,8]	[7,5,3,1,0]
m32.m22	2	11	[12,13,12,10,8]	[8,7,5,3,1,0]
m41.m08	2	11	[7,10,8,7,10,8]	[7,5,3,1,0]
m71.m26	2	11	[12,13,12,10,8,7,8]	[5,3,1,0]
m13.m22	2	10	[13,12,10,8]	[8,7,5,3,1,0]
m32.m08	2	10	[12,13,12,10,8]	[7,5,3,1,0]
m41.m26	2	10	[7,10,8,7,10,8]	[5,3,1,0]
m71.m05	2	10	[12,13,12,10,8,7,8]	[7,5,3]
m09.m22	2	9	[12,10,8]	[8,7,5,3,1,0]
m13.m08	2	9	[13,12,10,8]	[7,5,3,1,0]
m32.m26	2	9	[12,13,12,10,8]	[5,3,1,0]
m41.m05	2	9	[7,10,8,7,10,8]	[7,5,3]
m71.m01	2	9	[12,13,12,10,8,7,8]	[5,3]
m71.m03	2	9	[12,13,12,10,8,7,8]	[1,0]
m71.m15	2	9	[12,13,12,10,8,7,8]	[7,5]
m13.m26	2	8	[13,12,10,8]	[5,3,1,0]
m32.m05	2	8	[12,13,12,10,8]	[7,5,3]
m41.m01	2	8	[7,10,8,7,10,8]	[5,3]
m41.m03	2	8	[7,10,8,7,10,8]	[1,0]
m13.m05	2	7	[13,12,10,8]	[7,5,3]
m32.m01	2	7	[12,13,12,10,8]	[5,3]
m32.m03	2	7	[12,13,12,10,8]	[1,0]
m32.m04	2	7	[12,13,12,10,8]	[7,8]
m32.m15	2	7	[12,13,12,10,8]	[7,5]
m13.m01	2	6	[13,12,10,8]	[5,3]
m13.m03	2	6	[13,12,10,8]	[1,0]
m13.m15	2	6	[13,12,10,8]	[7,5]
m12.m07	2	5	[7,10,8]	[8,8]

Arcos específicos del cluster 17

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m11.m62	1	7	[1,3]	[7,5,7,5,3]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m28.m75	1	7	[0,0]	[5,7,7,7,8]
m38.m25	1	6	[5,7]	[5,7,5,3]
m05.m53	2	5	[7,5,3]	[3,2]
m40.m53	2	5	[7,7,5]	[3,2]
m28.m21	1	5	[0,0]	[7,7,8]
m01.m53	2	4	[5,3]	[3,2]
m15.m53	2	4	[7,5]	[3,2]
m38.m53	2	4	[5,7]	[3,2]
m28.m04	1	4	[0,0]	[7,8]
m28.m38	1	4	[0,0]	[5,7]
m38.m38	1	4	[5,7]	[5,7]
m53.m53	1	4	[3,2]	[3,2]

Arcos específicos del cluster 18

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m69.m26	2	8	[1,1,5,5]	[5,3,1,0]
m01.m56	1	8	[5,3]	[5,7,5,3,1,0]
m11.m56	1	8	[1,3]	[5,7,5,3,1,0]
m23.m26	1	8	[0,5,5,5]	[5,3,1,0]
m01.m08	1	7	[5,3]	[7,5,3,1,0]
m18.m26	1	7	[0,5,5]	[5,3,1,0]
m55.m26	1	7	[5,5,3]	[5,3,1,0]
m69.m55	1	7	[1,1,5,5]	[5,5,3]
m30.m26	4	6	[1,1]	[5,3,1,0]
m01.m26	2	6	[5,3]	[5,3,1,0]
m69.m01	2	6	[1,1,5,5]	[5,3]
m69.m03	2	6	[1,1,5,5]	[1,0]
m69.m06	2	6	[1,1,5,5]	[5,5]
m01.m25	1	6	[5,3]	[5,7,5,3]
m18.m55	1	6	[0,5,5]	[5,5,3]
m23.m01	1	6	[0,5,5,5]	[5,3]
m23.m03	1	6	[0,5,5,5]	[1,0]
m28.m23	1	6	[0,0]	[0,5,5,5]
m30.m69	1	6	[1,1]	[1,1,5,5]
m06.m55	4	5	[5,5]	[5,5,3]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m30.m55	4	5	[1,1]	[5,5,3]
m55.m03	4	5	[5,5,3]	[1,0]
m28.m55	2	5	[0,0]	[5,5,3]
m01.m05	1	5	[5,3]	[7,5,3]
m18.m01	1	5	[0,5,5]	[5,3]
m18.m03	1	5	[0,5,5]	[1,0]
m18.m06	1	5	[0,5,5]	[5,5]
m28.m18	1	5	[0,0]	[0,5,5]
m55.m01	1	5	[5,5,3]	[5,3]
m30.m06	10	4	[1,1]	[5,5]
m28.m06	6	4	[0,0]	[5,5]
m30.m01	4	4	[1,1]	[5,3]
m30.m03	4	4	[1,1]	[1,0]
m01.m01	2	4	[5,3]	[5,3]
m01.m11	1	4	[5,3]	[1,3]
m01.m15	1	4	[5,3]	[7,5]
m01.m38	1	4	[5,3]	[5,7]
m30.m30	1	4	[1,1]	[1,1]

Arcos específicos del cluster 19

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m28.m22	1	8	[0,0]	[8,7,5,3,1,0]
m28.m08	1	7	[0,0]	[7,5,3,1,0]
m28.m05	1	5	[0,0]	[7,5,3]
m28.m15	1	4	[0,0]	[7,5]

Arcos específicos del cluster 20

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m06.m43	1	10	[5,5]	[5,7,8,7,5,3,1,0]
m75.m08	1	10	[5,7,7,7,8]	[7,5,3,1,0]
m75.m26	1	9	[5,7,7,7,8]	[5,3,1,0]
m82.m08	1	9	[5,5,7,8]	[7,5,3,1,0]
m38.m22	4	8	[5,7]	[8,7,5,3,1,0]
m14.m08	3	8	[5,7,8]	[7,5,3,1,0]

id.	Frecuencia	Longitud	Notas Origen	Notas Destino
m06.m22	2	8	[5,5]	[8,7,5,3,1,0]
m11.m22	1	8	[1,3]	[8,7,5,3,1,0]
m21.m08	1	8	[7,7,8]	[7,5,3,1,0]
m75.m05	1	8	[5,7,7,7,8]	[7,5,3]
m82.m26	1	8	[5,5,7,8]	[5,3,1,0]
m38.m08	4	7	[5,7]	[7,5,3,1,0]
m06.m08	2	7	[5,5]	[7,5,3,1,0]
m11.m75	1	7	[1,3]	[5,7,7,7,8]
m21.m26	1	7	[7,7,8]	[5,3,1,0]
m75.m03	1	7	[5,7,7,7,8]	[1,0]
m75.m15	1	7	[5,7,7,7,8]	[7,5]
m82.m05	1	7	[5,5,7,8]	[7,5,3]
m14.m05	3	6	[5,7,8]	[7,5,3]
m21.m05	1	6	[7,7,8]	[7,5,3]
m82.m01	1	6	[5,5,7,8]	[5,3]
m82.m03	1	6	[5,5,7,8]	[1,0]
m06.m05	2	5	[5,5]	[7,5,3]
m11.m21	1	5	[1,3]	[7,7,8]
m21.m03	1	5	[7,7,8]	[1,0]
m21.m15	1	5	[7,7,8]	[7,5]
m11.m04	1	4	[1,3]	[7,8]

Referencias

Adalbjörnsson, S. I., Jakobsson, A., & Christensen, M. G. (2015). Multi-pitch estimation exploiting block sparsity. *Signal Processing*, *109*, 236–247.

Aizawa, A. (2003). An information-theoretic perspective of tf–idf measures. *Information Processing & Management*, *39*(1), 45–65.

Al Kork, S. K., Jaumard-Hakoun, A., Adda-Decker, M., Amelot, A., Buchman, L. C., Chawah, P., ... Roussel, P. (2014). A multi-sensor helmet to capture rare singing, an intangible cultural heritage study. *10th International Seminar on Speech Production (ISSP)*.

Ankerst, M., Breunig, M. M., Kriegel, H. -P., & Sander, J. (1999). OPTICS: ordering points to identify the clustering structure. *ACM Sigmod record*. ACM,

Antoniou, A. (2016). *Digital signal processing*. McGraw-Hill.

Antonopoulos, I., Pikrakis, A., Theodoridis, S., Cornelis, O., Moelants, D., & Leman, M. (2007). Music retrieval by rhythmic similarity applied on greek and african traditional music. *Proceedings of the 8th International Conference on Music Information Retrieval*.

Arredondo Arteaga, D. J., Gil González, W. J., Flórez, M., & José, J. (2017). Methodology for selection of attributes and operating conditions for SVM-Based fault locator's. *Tecnura*, *21*(51), 15–26.

Barlow, C. (2001). On the quantification of harmony and metre. *The Ratio Book. Cologne: Feedback Papers*, *43*, 2–23.

Bay, H., Ess, A., Tuytelaars, T., & Van Gool, L. (2008). Speeded-up robust features (SURF). *Computer vision and image understanding*, *110*(3), 346–359.

Benetos, E., Dixon, S., Giannoulis, D., Kirchhoff, H., & Klapuri, A. (2013). Automatic music transcription: challenges and future directions. *Journal of Intelligent Information Systems*, *41*(3), 407–434. doi:10.1007/s10844-013-0258-3

Benetos, E. & Holzapfel, A. (2015). Automatic transcription of Turkish microtonal music. *The Journal of the Acoustical Society of America*, *138*(4), 2118–2130.

Benning, M. S., Kapur, A., Till, B. C., & Tzanetakis, G. (2007). Multimodal Sensor Analysis of Sitar Performance: Where is the Beat?. *Multimedia Signal Processing, 2007. MMSP 2007. IEEE 9th Workshop on*. IEEE,

Benward, B. (2014). *Music in Theory and Practice Volume 1*. McGraw-Hill Higher Education.

Berkhin, P. (2006). A Survey of Clustering Data Mining Techniques. En J. Kogan, C. Nicholas, & M. Teboulle (Editores) *Grouping Multidimensional Data: Recent Advances in Clustering*. (pp. 25–71). Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/3-540-28349-8_2

Berlanga, M. Á. (2014). La originalidad musical del flamenco: el compás. *Sinfonía virtual*, (26).

Berlanga, M. Á. (2017). *El Flamenco, un Arte Musical y de la Danza*. Obtenido de http://www.ugr.es/~berlanga/index.html

Bisk, Y. & Hockenmaier, J. (2015). Probing the Linguistic Strengths and Limitations of Unsupervised Grammar Induction.. *ACL (1)*.

Bland, J. M. & Altman, D. G. (1995). Multiple significance tests: the Bonferroni method. *Bmj*, *310*(6973), 170.

Blas Vega, J. & Ríos Ruiz, M. (1988). *Diccionario enciclopédico ilustrado del flamenco*. Cinterco.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, *3*(Jan), 993–1022.

Blum, A. L. & Rivest, R. L. (1993). Training a 3-node neural network is NP-complete. *Machine learning: From theory to applications*. (pp. 9–28). Springer.

Bogdanov, D., Wack, N., Gómez, E., Gulati, S., Herrera, P., Mayor, O., ... Serra, X. (2013). Essentia: An Audio Analysis Library for Music Information Retrieval.. *ISMIR*. Citeseer,

Bond, J.. *Dalí in New York*. https://www.youtube.com/watch?v=0Q9LMBFI0I0 (Consultado el 2017.10.01).

Bozkurt, B., Yarman, O., Karaosmanoğlu, M. K., & Akkoç, C. (2009). Weighing diverse theoretical models on Turkish maqam music against pitch measurements: A comparison of peaks automatically derived from frequency histograms with proposed scale tones. *Journal of New Music Research*, *38*(1), 45–70.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). Classification and regression trees.

Bribiesca, E. (2016). A Contour-Oriented Approach to Shape Analysis via the Slope Chain Code. *International Journal of Contemporary Mathematical Sciences*, *11*(2), 65–84.

Bro, R. & Smilde, A. K. (2014). Principal component analysis. *Analytical Methods*, *6*(9), 2812–2831.

Caro Baroja, J. (1981). *Los Pueblos de España, tomo II*.

Castro Buendía, G. (2014). *Las mudanzas del cante en tiempo de silverio*. (Vol. 4). Primento.

Cerezo, A. M. (2016). *La industria cultural del flamenco: aspectos económicos y fiscales*. (Tesis doctoral).

Charrad, M., Ghazzali, N., Boiteau, V., & Niknafs, A. (2014). NbClust: An R Package for Determining the Relevant Number of Clusters in a Data Set. *Journal of Statistical Software*, *61*(3), 1-36. Obtenido de http://www.jstatsoft.org/v61/i06/.

Chomsky, N. (1959). On certain formal properties of grammars. *Information and control*, *2*(2), 137–167.

Chuse, L., ... (2015). Fandangos in Voices of Women: Enacting Tradition, Affirming Identity. *Música oral del Sur: revista internacional. Españoles, indios, africanos y gitanos. El alcance global del fandango en música, canto y danza.*, (12), 517–526.

COFLA. *Proyecto COFLA*. http://www.cofla-project.com/ (Consultado el 2017-10-16).

Cornelis, O., Lesaffre, M., Moelants, D., & Leman, M. (2010). Access to ethnic music: Advances and perspectives in content-based music information retrieval. *Signal Processing*, *90*(4), 1008–1031.

Cramér, H. (2016). *Mathematical Methods of Statistics (PMS-9)*. (Vol. 9). Princeton university press.

Cruces Roldán, C. (2003). *Antropología y flamenco (II)*. Signatura Ediciones.

Dallinga, J., Benjaminse, A., Gokeler, A., Cortes, N., Otten, E., & Lemmink, K. (2017). Innovative video feedback on jump landing improves landing technique in males. *International journal of sports medicine*, *38*(02), 150–158.

Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors. *Communications of the ACM*, *7*(3), 171–176.

Dannenberg, R. B. & Raphael, C. (2006). Music score alignment and computer accompaniment. *Communications of the ACM*, *49*(8), 38–43.

Díaz-Bañez, J. M., Farigu, G., Gómez, F., Rappaport, D., & Toussaint, G. T. (2004). El compás flamenco: a phylogenetic analysis. *Proceedings of BRIDGES: Mathematical Connections in Art, Music and Science*.

Díaz-Bañez, J. M., Farigu, G., Toussaint, G., Gómez, F., & Rappaport, D. (2005). Similaridad y evolución en la rítmica del flamenco: una incursión de la matemática computacional. *Gaceta de la Real Sociedad Matemática Española*, *8*(2), 489–509.

Díaz-Báñez, J. M. & Rizo, J. -C. (2014). An efficient DTW-based approach for melodic similarity in flamenco singing. *International Conference on Similarity Search and Applications*. Springer,

Downie, J., Ehmann, A., Bay, M., & Jones, M. (2010). The music information retrieval evaluation exchange: Some observations and insights. *Advances in music information retrieval*, 93–115.

Downie, J. S. (2008). The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research. *Acoustical Science and Technology*, *29*(4), 247–255.

Dress, A. W. & Huson, D. H. (2004). Constructing splits graphs. *IEEE/ACM transactions on Computational Biology and Bioinformatics*, *1*(3), 109–115.

Driedger, J. & Müller, M. (2016). A Review of Time-Scale Modification of Music Signals. *Applied Sciences*, *6*(2), 57.

Ducasse, S., Nierstrasz, O., Schärli, N., Wuyts, R., & Black, A. P. (2006). Traits: A mechanism for fine-grained reuse. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, *28*(2), 331–388.

Dunn, J. C. (1974). Well-separated clusters and optimal fuzzy partitions. *Journal of cybernetics*, *4*(1), 95–104.

D'Ulizia, A., Ferri, F., & Grifoni, P. (2011). A survey of grammatical inference methods for natural language learning. *Artificial Intelligence Review*, *36*(1), 1–27.

de la Higuera, C. (2005). A bibliographical study of grammatical inference. *Pattern Recognition*, *38*. doi:10.1016/j.patcog.2005.01.003

de Nebrija, A. (1492). *Grammatica Antonii Nebrissensis*.

El Nuevo Herald. *Ralph Lauren rinde homenaje a España en el desfile número 80 de su carrera*. http://www.elnuevoherald.com/entretenimiento/gente/article2017732.html (Consultado el 2012-09-13).

Esmael, B., Arnaout, A., Fruhwirth, R. K., & Thonhauser, G. (2015). A statistical feature-based approach for operations recognition in drilling time series. *International Journal of Computer Information Systems and Industrial Management Applications*, *5*, 454–461.

Ester, M., Kriegel, H. -P., Sander, J., Xu, X., ... (1996). A density-based algorithm for discovering clusters in large spatial databases with noise.. *Kdd*.

Euler, L. (1739). *Tentamen novae theoriae musicae ex certissimis harmoniae principiis dilucide expositae*. ex typographia Academiae scientiarum.

Faber, V. (1994). Clustering and the continuous k-means algorithm. *Los Alamos Science*, *22*(138144.21).

Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P., ... (1996). Knowledge Discovery and Data Mining: Towards a Unifying Framework.. *KDD*.

Flasiński, M. & Jurek, J. (2014). Fundamental methodological issues of syntactic pattern recognition. *Pattern Analysis and Applications*, *17*(3), 465–480. doi:10.1007/s10044-013-0322-1

Fréchet, M. M. (1906). Sur quelques points du calcul fonctionnel. *Rendiconti del Circolo Matematico di Palermo (1884-1940)*, *22*(1), 1–72.

Fred, A. L. & Jain, A. K. (2002). Data clustering using evidence accumulation. *Pattern Recognition, 2002. Proceedings. 16th International Conference on*. IEEE,

Fred, A. L. & Jain, A. K. (2005). Combining multiple clusterings using evidence accumulation. *IEEE transactions on pattern analysis and machine intelligence*, *27*(6), 835–850.

Frey, T. & Van Groenewoud, H. (1972). A cluster analysis of the D2 matrix of white spruce stands in Saskatchewan based on the maximum-minimum principle. *The Journal of Ecology*, 873–886.

Friedman, M. & Kandel, A. (1999). Introduction to pattern recognition: statistical, structural, neural and fuzzy logic approaches.

Fu, K. S. (1982). *Applications of pattern recognition*. CRC.

Gadd, T. (1990). PHONIX: The algorithm. *Program*, *24*(4), 363–366.

Gamboa, J. M. (2005). *Una historia del flamenco*. Espasa-Calpe.

García Lorca, F. (1922). El cante jondo: Primitivo cante andaluz.

García Lorca, F. (1931). Arquitectura del cante jondo.

García Matos, M. (1950). Cante flamenco. Algunos de sus presuntos origenes. *Anuario Musical*, *5*, 97.

Gergonne, J. (1974). The application of the method of least squares to the interpolation of sequences. *Historia Mathematica*, *1*(4), 439–447.

Ghias, A., Logan, J., Chamberlin, D., & Smith, B. C. (1995). Query by humming: musical information retrieval in an audio database. *Proceedings of the third ACM international conference on Multimedia*. ACM,

Giraldo, S. I., ... (2016). Computational modelling of expressive music performance in jazz guitar: a machine learning approach.

Giraldo, S. & Ramírez, R. (2016). A machine learning approach to ornamentation modeling and synthesis in jazz guitar. *Journal of Mathematics and Music*, *10*(2), 107–126.

Gola, D., Mahachie John, J. M., Van Steen, K., & König, I. R. (2015). A roadmap to multifactor dimensionality reduction methods. *Briefings in bioinformatics*, *17*(2), 293–308.

Gomaa, W. H. & Fahmy, A. A. (2013). A survey of text similarity approaches. *International Journal of Computer Applications*, *68*(13).

Gómez, E. & Bonada, J. (2013). Towards computer-assisted flamenco transcription: An experimental comparison of automatic transcription algorithms as applied to a cappella singing. *Computer Music Journal*, *37*(2), 73–90.

Gómez, E., Cañadas-Quesada, F. J., Salamon, J., Bonada, J., Vera-Candeas, P., & Molero, P. C. (2012). Predominant Fundamental Frequency Estimation vs Singing Voice Separation for the Automatic Transcription of Accompanied Flamenco Singing.. *ISMIR*.

Gómez, E., Herrera, P., & Gómez-Martin, F. (2013). Computational ethnomusicology: perspectives and challenges. *Journal of New Music Research*. Publicación en línea avanzada. doi:10.1080/09298215.2013.818038

Gómez, F., Pikrakis, A., Mora, J., Dıaz-Bánez, J. M., Gómez, E., & Escobar, F. (2011). Automatic detection of ornamentation in flamenco. *Fourth International Workshop on Machine Learning and Music MML*.

Gonzalez, R. C. & Thomason, M. G. (1978). Syntactic pattern recognition: An introduction.

Gotoh, O. (1982). An improved algorithm for matching biological sequences. *Journal of molecular biology*, *162*(3), 705–708.

Gu, Q., Li, Z., & Han, J. (2012). Generalized fisher score for feature selection. *arXiv preprint arXiv:1202.3725*.

Guastavino, C., Gomez, F., Toussaint, G., Marandola, F., & Gomez, E. (2009). Measuring similarity between flamenco rhythmic patterns. *Journal of New Music Research*, *38*(2), 129–138.

Guerrero, A. (2013). La técnica vocal en el cante flamenco. *II Congreso Investigación y Flamenco (INFLA)*. Universidad de Sevilla,

Guerrero, J. I., García, A., Personal, E., Luque, J., & León, C. (2017). Heterogeneous data source integration for smart grid ecosystems based on metadata mining. *Expert Systems with Applications*, *79*, 254–268.

Guyon, I. & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of machine learning research*, *3*(Mar), 1157–1182.

Guyon, I. & Elisseeff, A. (2006). An introduction to feature extraction. *Feature Extraction*. (pp. 1–25). Springer.

Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning*, *46*(1), 389–422.

Hall, M. A. & Smith, L. A. (1999). Feature Selection for Machine Learning: Comparing a Correlation-Based Filter Approach to the Wrapper.. *FLAIRS conference*.

Halmos, I., Köszegi, G., & Mandler, G. (1978). Computational Ethnomusicology in Hungary. *MI: MPublishing, University of Michigan Library*, 775–783.

Hamming, R. W. (1950). Error detecting and error correcting codes. *Bell Labs Technical Journal*, *29*(2), 147–160.

Hartigan, J. A. & Wong, M. A. (1979). Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, *28*(1), 100–108.

Hastie, T., Tibshirani, R., & Friedman, J. (2016). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. (2ª ed.). Springer.

Herrera, P., Serra, X., & Peeters, G. (1999). Audio Descriptors and Descriptor Schemes in the Context of MPEG-7.. *ICMC 1999*.

Hewlett, W. B. & Selfridge-Field, E. (1998). *Melodic similarity: Concepts, procedures, and applications*. Mit Press.

Hinton, G. E. & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *science*, *313* (-5786), 504–507.

Holzapfel, A. & Stylianou, Y. (2009). Rhythmic similarity in traditional Turkish music. *ISMIR-International Conference on Music Information Retrieval*.

Hore, P., Hall, L. O., & Goldgof, D. B. (2009). A scalable framework for cluster ensembles. *Pattern Recognition*, *42*(5), 676–688.

Horning, J. J. (1969). *A study of grammatical inference*. (Tesis doctoral).

Huang, J., Horowitz, J. L., & Ma, S. (2008). Asymptotic properties of bridge estimators in sparse high-dimensional regression models. *The Annals of Statistics*, 587–613.

Hubert, L. J. & Levin, J. R. (1976). A general statistical framework for assessing categorical clustering in free recall.. *Psychological bulletin*, *83*(6), 1072.

Hwa, R. (1999). Supervised grammar induction using training data with limited constituent information. *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*. Association for Computational Linguistics,

Inkpen, K. M. (2001). Drag-and-drop versus point-and-click mouse interaction styles for children. *ACM Transactions on Computer-Human Interaction (TOCHI)*, *8*(1), 1–33.

Internet live stats. *Internet live stats*. http://www.internetlivestats.com/ (Consultado el 2017-09-20).

ISMIR. The 17th International Society for Music Information Retrieval Conference. New York (2016).

Jaccard, P. (1912). The distribution of the flora in the alpine zone.. *New phytologist*, *11*(2), 37–50.

Jacob, L., Obozinski, G., & Vert, J. -P. (2009). Group lasso with overlap and graph lasso. *Proceedings of the 26th annual international conference on machine learning*. ACM,

Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Pattern recognition letters*, *31*(8), 651–666.

Jain, A. K. & Dubes, R. C. (1988). *Algorithms for clustering data*. Prentice-Hall, Inc..

Jang, J. -S. R., Lee, H. -R., & Yeh, C. -H. (2001). Query by tapping: A new paradigm for content-based music retrieval from acoustic input. *Pacific-Rim Conference on Multimedia*. Springer,

Jaro, M. A. (1989). Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida. *Journal of the American Statistical Association*, *84*(406), 414–420.

Jost, M.. *Flamenco with a Foreign Accent*. http://flamencoproject.com/flamenco_foreign_accent.html (Consultado el 2017.10.01).

Juan, A. & Vidal, E. (2000). On the use of normalized edit distances and an efficient k-NN search technique (k-AESA) for fast and accurate string classification. *Pattern Recognition, 2000. Proceedings. 15th International Conference on*. IEEE,

Kahraman, C., Kaymak, U., & Yazici, A. (2016). *Fuzzy Logic in Its 50th Year: new developments, directions and challenges*. (Vol. 341). Springer.

Karp, R. M. (1972). Reducibility among combinatorial problems. *Complexity of computer computations*. (pp. 85–103). Springer.

Kass, G. V. (1980). An exploratory technique for investigating large quantities of categorical data. *Applied statistics*, 119–127.

Kaufman, L. & Rousseeuw, P. J. (2009). *Finding groups in data: an introduction to cluster analysis*. (Vol. 344). John Wiley & Sons.

Kernighan, B. W. & Ritchie, D. M. (2006). *The C programming language*. Prentice Hall.

Kim, S., Unal, E., & Narayanan, S. (2008). Music fingerprint extraction for classical music cover song identification. *Multimedia and Expo, 2008 IEEE International Conference on*. IEEE,

Kim, S. & Xing, E. P. (2009). Statistical estimation of correlated genome associations to a quantitative trait network. *PLoS genetics*, *5*(8), e1000587.

King, R. D., Feng, C., & Sutherland, A. (1995). Statlog: comparison of classification algorithms on large real-world problems. *Applied Artificial Intelligence an International Journal*, *9*(3), 289–333.

Kira, K. & Rendell, L. A. (1992). The feature selection problem: Traditional methods and a new algorithm. *Aaai*.

Klyne, G. & Newman, C. (2002). *Rfc 3339: Date and time on the internet: Timestamps*. (Informe técnico).

Kohavi, R. & John, G. H. (1997). Wrappers for feature subset selection. *Artificial intelligence*, *97*(1-2), 273–324.

Kotsiantis, S. B., Zaharakis, I., & Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. En I. G. Maglogiannis (Editor) *Emerging artificial intelligence applications in computer engineering: real word AI systems with applications in eHealth, HCI, information retrieval and pervasive technologies*.

Kotsiantis, S., Kanellopoulos, D., & Pintelas, P. (2006). Data preprocessing for supervised leaning. *International Journal of Computer Science*, *1*(2), 111–117.

Kroher, N., Díaz-Báñez, J. -M., Mora, J., & Gómez, E. (2016). Corpus COFLA: a research corpus for the computational study of flamenco music. *Journal on Computing and Cultural Heritage (JOCCH)*, *9*(2), 10.

Kroher, N. & Gómez, E. (2016). Automatic transcription of flamenco singing from polyphonic music recordings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *24*(5), 901–913.

Kroher, N., Gómez, E., Guastavino, C., Gómez, F., & Bonada, J. (2014). Computational Models for Perceived Melodic Similarity in A Cappella Flamenco Singing.. *ISMIR*.

Lance, G. N. & Williams, W. T. (1966). Computer programs for hierarchical polythetic classification ("similarity analyses"). *The Computer Journal*, *9*(1), 60–64.

Lance, G. -N. & Williams, W. -T. (1967). A general theory of classification sorting strategies-Hierarchical System. *Cognitive Journal*, *9*, 373–380.

Land, A. H. & Doig, A. G. (1960). An automatic method of solving discrete programming problems. *Econometrica: Journal of the Econometric Society*, 497–520.

Larsson, N. J. & Moffat, A. (2000). Off-line dictionary-based compression. *Proceedings of the IEEE*, *88*(11), 1722–1732.

León, C., López, A., Monedero, I., & Montaño, J. C. (2001). Classification of disturbances in electrical signals using neural networks. *International Work-Conference on Artificial Neural Networks*. Springer,

León, C., Molina, F. J., Fragoso, C., & Algarín, A. (1997). Reconocimiento automático de caracteres empleando técnicas de sistemas expertos. *Novática*, (128), 50–55.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet physics doklady*.

Li, Y. & Chen, J. X. (2014). A nondeterministic approach to infer context free grammar from sequence. *Wavelet Active Media Technology and Information Processing (ICCWAMTIP), 2014 11th International Computer Conference on*. IEEE,

Littau, D. & Boley, D. (2006). *Clustering Very Large Data Sets with Principal Direction Divisive Partitioning.*.

Liu, H. & Motoda, H. (1998). *Feature extraction, construction and selection: A data mining perspective*. (Vol. 453). Springer Science & Business Media.

Liu, H. & Motoda, H. (2007). *Computational methods of feature selection*. CRC Press.

Liu, H. & Motoda, H. (2012). *Feature selection for knowledge discovery and data mining*. (Vol. 454). Springer Science & Business Media.

Liu, H. & Yu, L. (2005). Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on knowledge and data engineering*, *17*(4), 491–502.

Liu, J. & Ye, J. (2010). Moreau-Yosida regularization for grouped tree structure learning. *Advances in Neural Information Processing Systems*.

Livni, R., Shalev-Shwartz, S., & Shamir, O. (2014). On the computational efficiency of training neural networks. *Advances in Neural Information Processing Systems*.

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., & Watkins, C. (2002a). Text classification using string kernels. *Journal of Machine Learning Research*, *2*(Feb), 419–444.

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., & Watkins, C. (2002b). Text classification using string kernels. *Journal of Machine Learning Research*, *2*(Feb), 419–444.

Loh, W. -Y. (2014). Fifty years of classification and regression trees. *International Statistical Review*, *82*(3), 329–348.

Loh, W. -Y. & Shih, Y. -S. (1997). Split selection methods for classification trees. *Statistica sinica*, 815–840.

López, O. (2003). 38. La Cuenca Minera de la provincia de Huelva. Su folclor. *revista ph*, (45).

López-Gómez, C. (2013). *Criterios para la transcripción manual de la colección de TONAS*. (Informe técnico). Music Technology Group, Universitat Pompeu Fabra.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, *60*(2), 91–110.

Luque, J., Larios, D. F., Personal, E., Barbancho, J., & León, C. (2016). Evaluation of MPEG-7-Based Audio Descriptors for Animal Voice Recognition over Wireless Acoustic

Sensor Networks. *Sensors*, *16*(5), 717. doi:10.3390/s16050717

Ma, S. & Huang, J. (2008). Penalized feature selection and classification in bioinformatics. *Briefings in bioinformatics*, *9*(5), 392–403.

MacKenzie, I. S., Sellen, A., & Buxton, W. A. (1991). A comparison of input devices in element pointing and dragging tasks. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM,

Magas, M. & Proutskova, P. (2013). A location-tracking interface for ethnomusicological collections. *Journal of New Music Research*, *42*(2), 151–160.

MakeMusic (2014). *Finale*. MakeMusic. Obtenido de http://www.finalemusic.com/ (2015-01-25)

Marlow, S. (2010). *Haskell 2010 Language Report*. Haskell Committee.

Marqués Donaire, M. I. (2017). *El flamenco como vehículo de la religiosidad popular*. (Tesis doctoral). Universidad de Sevilla.

Massey Jr, F. J. (1951). The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American statistical Association*, *46*(253), 68–78.

Mastriani, M. (2016). Denoising and compression in wavelet domain via projection onto approximation coefficients. *arXiv preprint arXiv:1608.00265*.

Mathews, G. B. (1896). On the partition of numbers. *Proceedings of the London Mathematical Society*, *1*(1), 486–490.

Mauch, M., Cannam, C., Bittner, R., Fazekas, G., Salamon, J., Dai, J., ... Dixon, S. (2015). Computer-aided Melody Note Transcription Using the Tony Software: Accuracy and Efficiency. *Proceedings of the First International Conference on Technologies for Music Notation and Representation*. (accepted)

McAuley, J., Ming, J., Stewart, D., & Hanna, P. (2005). Subband correlation and robust speech recognition. *IEEE Transactions on Speech and Audio Processing*, *13*(5), 956–964.

McClain, J. O. & Rao, V. R. (1975). Clustisz: A program to test for the quality of clustering of a set of objects. *Journal of Marketing Research*, 456–460.

Mcdonald, G.. *Every Noise at Once*. http://everynoise.com (Consultado el 2017-10-14).

McLachlan, G. J. & Basford, K. E. (1988). *Mixture models: Inference and applications to clustering*. (Vol. 84). Marcel Dekker.

Menzies, D. & McPherson, A. (2015). Highland piping ornament recognition using dynamic time warping.. *NIME*.

Messenger, R. & Mandell, L. (1972). A modal search technique for predictive nominal scale multivariate analysis. *Journal of the American statistical association*, *67*(340), 768–772.

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., & Leisch, F. (2017). *E1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*. Obtenido de https://CRAN.R-project.org/package=e1071 (R package version 1.6-8)

Miller, H. & Han, J. (2001). Spatial clustering methods in data mining: a survey. *Geographic data mining and knowledge discovery, Taylor and Francis*.

Molina, R. & Mairena, A. (1963). *Mundo y formas del cante flamenco* (J. Cenizo, Editor). (Reimpresión 2004 por José Cenizo en Ediciones Giralda ed.). Revista de Occidente.

Monge, A. E. & Elkan, C. (1996). The Field Matching Problem: Algorithms and Applications.. *KDD*.

Moore, J. (2015). Cante Libre is not free – Contrasting Approaches To Fandangos Personales. *Música oral del sur*, (12), 185 - 198.

Mora, J. (2013). La voz flamenca. Un estudio científico. *FLAMENCO EN RED 2013/2014. Ciclo de conferencias Flamenco y Ciencias*. Universidad de Cádiz. Obtenido de https://youtu.be/tLCcDINalc4

Mora, J., Gómez, F., Gómez, E., & Díaz-Báñez, J. M. (2016). Melodic Contour and Mid-Level Global Features Applied to the Analysis of FlamencoCantes. *Journal of*

New Music Research, *45*(2), 1–15. doi:10.1080/09298215.2016.1174717

Mora, J., Gómez, F., Gómez, E., Escobar-Borrego, F. J., & Díaz-Bañez, J. M. (2010). Melodic Characterization and Similarity in A Cappella Flamenco Cantes. *International Society for Music Information Retrieval Conference (ISMIR)*. International Society for Music Information Retrieval Conference, ISMIR, Utrecht, The Nederlands: Obtenido de http://mtg.upf.edu/files/publications/MelodyFlamenco-ISMIR2010.pdf

Mora, J. & López Gómez, C.. *TONAS: a dataset of flamenco a cappella sung melodies with corresponding manual transcriptions*. http://mtg.upf.edu/download/datasets/tonas (Consultado el 2017-04-10).

Morgan, J. N. & Sonquist, J. A. (1963). Problems in the analysis of survey data, and a proposal. *Journal of the American statistical association*, *58*(302), 415–434.

Müllensiefen, D. & Frieler, K. (2004). Measuring melodic similarity: Human vs. algorithmic judgments. *Proceedings of the Conference on Interdisciplinary Musicology (CIM04)*.

Murtagh, F. (1983). A survey of recent advances in hierarchical clustering algorithms. *The Computer Journal*, *26*(4), 354–359.

Music Technology Group. *Página principal del Music Technology Group*. https://www.upf.edu/web/mtg (Consultado el 2017-10-16).

Navarro García, J. -L. & Ropero Núñez, M. (1995). *Historia del flamenco* (J. -L. Navarro García & M. Ropero Núñez, Editores). Tartessos.

Needleman, S. B. & Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, *48*(3), 443–453.

Nevill-Manning, C. G. (1996). *Inferring sequential structure*. (Tesis doctoral). University of Waikato.

Nevill-Manning, C. G. & Witten, I. H. (1997). Identifying hierarchical structure in sequences: A linear-time algorithm. *J. Artif. Intell. Res.(JAIR)*, *7*, 67–82.

Ng, R. T. & Han, J. (2002). CLARANS: A method for clustering objects for spatial data mining. *IEEE transactions on knowledge and data engineering*, *14*(5), 1003–1016.

Núñez, F. (Editor). (1998). Todo el flamenco– de la A a la Z. Edilibro S.L..

Nystrom, A. & Hughes, J. (2016). Efficient Calculation of Polynomial Features on Sparse Matrices.

Ocón y Rivas, E. (1874). *Cantos españoles con notas explicativas y biográficas*. (Málaga)

Olson, C. F. (1995). Parallel algorithms for hierarchical clustering. *Parallel computing*, *21*(8), 1313–1325.

Orio, N., ... (2006). Music retrieval: A tutorial and review. *Foundations and Trends in Information Retrieval*, *1*(1), 1–90.

Osuna Lucena, M. -I. (1995). La música arábigo-andaluza. En J. L. Navarro García & M. Ropero Núñez (Editores) *Historia del flamenco*. (Vol. I, pp. 85-109). Tartessos.

Partee, B. H., ter Meulen, A. G. B., & Wall, R. E. (1990). *Mathematical methods in linguistics*. Kluwer Academic.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Pedrell, F. (1891). *Por nuestra música*. (Barcelona)

Peng, H., Long, F., & Ding, C. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on pattern analysis and machine intelligence*, *27*(8), 1226–1238.

Pennebaker, J. W., Boyd, R. L., Jordan, K., & Blackburn, K. (2015). The development and psychometric properties of LIWC2015.

Perez, A., Maestre, E., Kersten, S., & Ramirez, R. (2008). Expressive Irish Fiddle Performance Model Informed with Bowing.. *ICMC*.

Perkins, S., Lacker, K., & Theiler, J. (2003). Grafting: Fast, incremental feature selection by gradient descent in function space. *Journal of machine learning research*, *3*(Mar), 1333–1356.

Peterson, J., Hammond, K., Augustsson, L., Boutel, B., Burton, W., Fasel, J., ... (1997). *Report on the programming language Haskell*. (Informe técnico). version 1.4. Technical report, Department of Computer Science, Yale University, 1997. Available from http://www. haskell. org.

Pezeshk, A. & Tutwiler, R. L. (2011). Automatic feature extraction and text recognition from scanned topographic maps. *IEEE transactions on Geoscience and Remote sensing*, *49*(12), 5047–5063.

Pikrakis, A., Gómez, F., Oramas, S., Díaz-Báñez, J. M., Mora, J., Escobar-Borrego, F., ... Salamon, J. (2012). Tracking Melodic Patterns in Flamenco Singing by Analyzing Polyphonic Music Recordings.. *ISMIR*.

Pikrakis, A., Kroher, N., & Díaz-Báñez, J. -M. (2016). Detection of Melodic Patterns in Automatic Transcriptions of Flamenco Singing. *6th International Workshop on Folk Music Analysis*. Dublin Institute of Technology.

Pinto, D., Vilariño, D., Alemán, Y., Gómez, H., Loya, N., & Jiménez-Salazar, H. (2012). The Soundex phonetic algorithm revisited for SMS text representation. *Text, Speech and Dialogue*. Springer,

Porter, A., Sordo, M., & Serra, X. (2013). Dunya: A system for browsing audio music collections exploiting cultural context. *Proceedings of 14th International Society for Music Information Retrieval Conference (ISMIR 2013), Curitiba, Brazil*. PPGIa, PUCPR Curitiba, Brazil,

Powell, D. R., Allison, L., & Dix, T. I. (2000). Fast, Optimal Alignment of Three Sequences Using Linear Gap Costs. *Journal of Theoretical Biology*, *207*. doi:10.1006/jtbi.2000.2177

Preciado, D. (1969). *Folklore Español: música, danza y ballet*. (p. 336). Studium Ediciones.

Provost, F. & Fawcett, T. (2013). *Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking*. (1ª ed.). O'Reilly Media.

Puiggròs, M., Gómez, E., Ramírez, R., Serra, X., & Bresin, R. (2006). Automatic characterization of ornamentation from bassoon recordings for expressive synthesis. *Proceedings of International Conference on Music Perception and Cognition*.

Pujadas, M. P. (2016). *Historia de la música (4a edición revisada, aumentada)*. Ed. Universidad de Cantabria.

Purwins, H. (2005). *Profiles of Pitch Classes - Circularity of Relative Pitch and Key: Experiments, Models, Music Analysis, and Perspectives*. (Tesis doctoral).

Quinlan, J. R. (1993). *C4.5: programs for machine learning*. Elsevier.

Quinlan, J. R. (2014). *C4.5: programs for machine learning*. Elsevier.

Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, *1*(1), 81–106.

Quiñones Castilla, M. (2012). El fandango en la provincia de Huelva. *XXV Jornadas del Patrimonio de la Comarca de la Sierra.*. Diputación Provincial. Obtenido de http://www.federacionsierra.es/media/documentos/doc498.pdf

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. Obtenido de https://www.R-project.org

Ramos, J., ... (2003). Using tf-idf to determine word relevance in document queries. *Proceedings of the first instructional conference on machine learning*.

Rice, T. (2013). *Ethnomusicology: A very short introduction*. Oxford University Press.

Rico, F. (1998). Aparato Crítico. *Don Quijote de la Mancha*. (Instituto Cervantes Crítica ed.). Obtenido de http://cvc.cervantes.es/literatura/clasicos/quijote/default.htm

Rish, I. (2001). An empirical study of the naive Bayes classifier. *IJCAI 2001 workshop on empirical methods in artificial intelligence*. IBM,

Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, *14*(5), 465–471.

Ritchie, M. D., Hahn, L. W., Roodi, N., Bailey, L. R., Dupont, W. D., Parl, F. F., & Moore, J. H. (2001).

Multifactor-dimensionality reduction reveals high-order interactions among estrogen-metabolism genes in sporadic breast cancer. *The American Journal of Human Genetics*, *69*(1), 138–147.

Robnik-Šikonja, M. & Kononenko, I. (2003). Theoretical and empirical analysis of ReliefF and RReliefF. *Machine learning*, *53*(1-2), 23–69.

Romero Jiménez, J. (1996). *La otra historia del flamenco: la tradición semítico musical andaluza*. Junta de Andalucía, Consejería de Cultura.

Rossy, H. (1966). *Teoría del cante jondo*. (Vol. 35). Credsa.

Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, *20*, 53–65.

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. *Computer Vision (ICCV), 2011 IEEE international conference on*. IEEE,

Russell, S. & Norvig, P. (2005). *Artificial Inteligence: a modern approach*. (Vol. 2, Nº 3, p. 4).

Salamon, J. & Gómez, E. (2012). Melody Extraction from Polyphonic Music Signals using Pitch Contour haracteristics.. *IEEE Transactions on Audio, Speech and Language Processing*, *20*(6), 1759-1770. doi:10.1109/TASL.2012.2188515

Sandler, T., Blitzer, J., Talukdar, P. P., & Ungar, L. H. (2009). Regularized learning with networks of features. *Advances in neural information processing systems*.

Savaresi, S. M., Boley, D. L., Bittanti, S., & Gazzaniga, G. (2002). Cluster selection in divisive clustering algorithms. *Proceedings of the 2002 SIAM International Conference on Data Mining*. SIAM,

Schikuta, E. & Erhart, M. (1997). The BANG-clustering system: Grid-based data analysis. *Advances in Intelligent Data Analysis Reasoning about Data*, 513–524.

Scott, S. & Matwin, S. (1999). Feature engineering for text classification. *ICML*.

Seeger, C. (1958). Prescriptive and descriptive music-writing. *The Musical Quarterly*, *44*(2), 184–195.

Sellers, P. H. (1974). On the theory and computation of evolutionary distances. *SIAM Journal on Applied Mathematics*, *26*(4), 787–793.

Severyn, A. & Moschitti, A. (2013). Automatic Feature Engineering for Answer Selection and Extraction.. *EMNLP*.

Shafranovich, Y. (2005). *Rfc4180: Common Format and MIME Type for Comma-Separated Values (CSV) Files*. (RFC-4180)

Shannon, C. E. & Weaver, W. (1949). The mathematical theory of communication.. *Univ. Illinois Press*, *1*, 17.

Six, J. & Cornelis, O. (2011). Tarsos: a platform to explore pitch scales in non-western and western music. *12th International Society for Music Information Retrieval Conference (ISMIR-2011)*. University of Miami, Frost School of Music,

Sokal, R. R. (1958). A statistical method for evaluating systematic relationship. *University of Kansas science bulletin*, *28*, 1409–1438.

Srinivasamurthy, A., Holzapfel, A., & Serra, X. (2014). In search of automatic rhythm analysis methods for turkish and indian art music. *Journal of New Music Research*, *43*(1), 94–114.

Steinbach, M., Ertöz, L., & Kumar, V. (2004). The challenges of clustering high dimensional data. *New directions in statistical physics*. (pp. 273–309). Springer.

Steinbach, M., Karypis, G., Kumar, V., ... (2000). A comparison of document clustering techniques. *KDD workshop on text mining*. Boston,

Storer, J.. *Juce*. www.juce.com (Consultado el 2016-07-01).

Strehl, A. & Ghosh, J. (2002). Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, *3*(Dec), 583–617.

Subramaniam, L. V., Roy, S., Faruquie, T. A., & Negi, S. (2009). A Survey of Types of Text Noise and Techniques to Handle

Noisy Text. *Proceedings of The Third Workshop on Analytics for Noisy Unstructured Text Data*. New York, NY, USA: ACM. doi:10.1145/1568296.1568315

Sulzer, J. G., Baker, N. K., Koch, H. C., & Christensen, T. (1995). *Aesthetics and the art of musical composition in the German enlightenment: Selected writings of Johann Georg Sulzer and Heinrich Christoph Koch*. (Vol. 7). Cambridge University Press.

Swan, A. R. & Sandilands, M. (1995). Introduction to geological data analysis. *International Journal of Rock Mechanics and Mining Sciences and Geomechanics Abstracts*.

Syropoulos, A. (2000). Mathematics of multisets. *Workshop on Membrane Computing*. Springer,

Tan, P. -N., Steinbach, M., & Kumar, V. (2013). Data mining cluster analysis: Basic concepts and algorithms. *Introduction to Data Mining*. Addison-Wesley.

Tang, J., Alelyani, S., & Liu, H. (2014). Feature selection for classification: A review. *Data Classification: Algorithms and Applications*, 37.

Teleprensa (2014). El flamenco se apodera del Museo de la Guitarra. *Teleprensa*, (2014-03-07).

Tenenbaum, J. B., De Silva, V., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *science*, *290*(5500), 2319–2323.

Tenzer, M. (2006). *Analytical studies in world music*. Oxford University Press, USA.

Therneau, T. M., Atkinson, B., Ripley, B., ... (2010). Rpart: Recursive partitioning. *R package version*, *3*, 1–46.

Therneau, T., Atkinson, B., & Ripley, B. (2015). *Rpart: Recursive Partitioning and Regression Trees*. Obtenido de https://CRAN.R-project.org/package=rpart (R package version 4.1-10)

Tibshirani, R. (2011). Regression shrinkage and selection via the lasso: a retrospective. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *73*(3), 273–282.

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., & Knight, K. (2005). Sparsity and smoothness via the fused lasso.

Journal of the Royal Statistical Society: Series B (Statistical Methodology), *67*(1), 91–108.

Togootogtokh, E., Shih, T. K., Kumara, W., Wu, S. -J., Sun, S. -W., & Chang, H. -H. (2017). 3D finger tracking and recognition image processing for real-time music playing with depth sensors. *Multimedia Tools and Applications*, 1–16.

Tung-Shou, C., Tzu-Hsin, T., Yi-Tzu, C., Chin-Chiang, L., Rong-Chang, C., Shuan-Yow, L., & Hsin-Yi, C. (2005). A combined K-means and hierarchical clustering method for improving the clustering efficiency of microarray. *International Symposium on Intelligent Signal Processing and Communication Systems*. IEEE, doi:10.1109/ISPACS.2005.1595432

Tzanetakis, G. (2014). Computational ethnomusicology: a music information retrieval perspective.. *ICMC*.

Tzanetakis, G., Kapur, A., Schloss, W. A., & Wright, M. (2007). Computational ethnomusicology. *Journal of interdisciplinary music studies*, *1*(2), 1–24.

the Unicode Consortium (2017). *The Unicode Standard – Version 10.0*.

Van Rijsbergen, C. (1979). Information retrieval. Obtenido de http://citeseer.ist.psu.edu/vanrijsbergen79information. html

Vivien, B.. *Variation Techniques for Composers and Improvisors*. http://www.bestmusicteacher.com/index.php?id=articles_on_composing (Consultado el 2017-08-10).

Volk, A., Garbers, J., van Kranenburg, P., Wiering, F., Grijp, L., & Veltkamp, R. C. (2007). Comparing computational approaches to rhythmic and melodic similarity in folksong research. *International Conference on Mathematics and Computation in Music*. Springer,

Völkel, T., Abeßer, J., Dittmar, C., & Großmann, H. (2010). Automatic genre classification of latin american music using characteristic rhythmic patterns. *Proceedings of the 5th Audio Mostly Conference: A Conference on Interaction with Sound*. ACM,

Wagner, R. A. & Fischer, M. J. (1974). The string-to-string correction problem. *Journal of the ACM (JACM)*, *21*(1), 168–173.

Walker, J. S. (2008). *A primer on wavelets and their scientific applications*. CRC press.

Wang, A. (2006). The Shazam music recognition service. *Communications of the ACM*, *49*(8), 44–48.

Wang, J., Zhao, P., Hoi, S. C., & Jin, R. (2014). Online feature selection and its applications. *IEEE Transactions on Knowledge and Data Engineering*, *26*(3), 698–710.

Wang, J. (2012). Locally linear embedding. *Geometric Structure of High-Dimensional Data and Dimensionality Reduction*. (pp. 203–220). Springer.

Weinberger, K. Q. & Saul, L. K. (2006). An introduction to nonlinear dimensionality reduction by maximum variance unfolding. *AAAI*.

Welch, T. A. (1984). A technique for high-performance data compression. *Computer*, *6*(17), 8–19.

Winkler, W. E. (1990). String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage.. (ERIC)

Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

WolframAlpha. *Flamenco en la Wolfram Alpha Knowledgebase, 2017*. https://www.wolframalpha.com/input/?i=flamenco (Consultado el 2017-12-26).

Wolpert, D. H. (1996). The lack of a priori distinctions between learning algorithms. *Neural computation*, *8*(7), 1341–1390.

Wright, M., Schloss, W. A., & Tzanetakis, G. (2008). Analyzing Afro-Cuban Rhythms using Rotation-Aware Clave Template Matching with Dynamic Programming.. *ISMIR*.

Xia, Q. (2009). The geodesic problem in quasimetric spaces. *Journal of Geometric Analysis*, *19*(2), 452–479.

Xu, D. & Tian, Y. (2015). A Comprehensive Survey of Clustering Algorithms. *Annals of Data Science*, *2*(2), 165–193. doi:10.1007/s40745-015-0040-1

Xu, X., Ester, M., Kriegel, H. -P., & Sander, J. (1998). A distribution-based clustering algorithm for mining in large spatial databases. *Data Engineering, 1998. Proceedings., 14th International Conference on*. IEEE,

Yöre, S. (2012). Maqam in music as a concept, scale and phenomenon. *Zeitschrift für die Welt der Türken/Journal of World of Turks*, *4*(3), 267–286.

Young, F. W. (2013). *Multidimensional scaling: History, theory, and applications*. Psychology Press.

Yu, L. & Liu, H. (2003). Feature selection for high-dimensional data: A fast correlation-based filter solution. *Proceedings of the 20th international conference on machine learning (ICML-03)*.

Yuan, M. & Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *68*(1), 49–67.

Zadeh, L. A. (1965). Fuzzy sets. *Information and control*, *8*(3), 338–353.

Zhang, H. (2004). The optimality of naive Bayes. *AA*, *1*(2), 3.

Zhao, Z., Morstatter, F., Sharma, S., Alelyani, S., Anand, A., & Liu, H. (2010). Advancing feature selection research. *ASU feature selection repository*, 1–28.

Zinemanas, P., Arias, P., Haro, G., & Gómez, E. (In Press). Visual music transcription of clarinet video recordings trained with audio-based labelled data. *ICCV 2017 Workshop on Computer Vision for Audio-Visual Media (CVAVM)*. Venice, Italy: Obtenido de https://zenodo.org/record/848650

Ziv, J. & Lempel, A. (1978). Compression of individual sequences via variable-rate coding. *IEEE transactions on Information Theory*, *24*(5), 530–536.

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American statistical association*, *101*(476), 1418–1429.

Zou, H. & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *67*(2), 301–320.