Reconocimiento automático de notas del piano

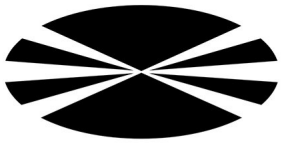
David Rodríguez Bacelar

Kevin Millán Canchapoma

Luca D'angelo Sabín

Jorge Hermo González

24 de abril de 2022



UNIVERSIDADE DA CORUÑA

Índice

1. Glosario **3**

2. Introducción **3**

3. Descripción del problema **4**
3.1. Restricciones 4
3.2. Características 5

4. Análisis bibliográfico **5**

5. Desarrollo **7**
5.1. Primera aproximación 7
5.1.1. Descripción 7
5.1.2. Resultados 10
5.1.3. Discusión 12
5.2. Segunda aproximación 15
5.2.1. Resultados 18
5.2.2. Discusión 20
5.3. Tercera aproximación 21
5.3.1. Resultados 27
5.3.2. Discusión 29

6. Conclusiones **30**

7. Trabajo futuro **30**

8. Bibliografía **30**

1. Glosario

- Sample: muestra de carácter musical
- Cover: Reinterpretación de una canción por parte de alguien diferente al que la compuso.
- VP: Verdadero positivo
- VN: Verdadero negativo
- FP: Falso positivo
- FN: Falso negativo
- RNA: Red de neuronas artificial.
- SVM: *Support Vector Machine* (Máquina de soporte vectorial).
- kNN: *K-nearest neighbors* (k vecinos más cercanos).
- poly: *Kernel* polinomial.
- rbf: *Kernel* de función de base radial (Gaussiano)
- sigmoid: *Kernel* sigmoidal

2. Introducción

A raíz de la pandemia, el aumento del interés por el aprendizaje en diferentes ámbitos llegó tambíen a la música, y con él, la aparición de herramientas para aprender a tocar diferentes instrumentos de forma autodidacta.

Así, para cualquiera que esté aprendiendo, el escuchar una canción que te gusta e intentar tocarla es algo que acaba siendo un proceso frustante y que requiere una gran cantidad de horas intentando sacar las notas que la componen.

Nuestro sistema se encargaría entonces de reconocer y diferenciar a partir de audios las notas de una pieza de piano pudiendo, en un futuro, ser capaz de detectar acordes y tonalidades, siendo útil en aplicaciones como Spotify, Tidal... Para ello, haremos uso de diferentes técnicas de aprendizaje automático como las redes de neuronas artificiales, árboles de decisión y máquinas de soporte vectorial, comparando su rendimiento y eligiendo la que mejor resultados nos ofrezca.

A lo largo de esta memoria analizaremos a fondo el problema a resolver en la Sección 3, desarrollaremos las diferentes soluciones en la Sección 5, hablaremos sobre las conclusiones del trabajo en la Sección 6 y finalizaremos comentando las aplicaciones al mundo real en la Sección 7. También se podrá consultar las bibliografía utilizada en la Sección 4 y 8.

3. Descripción del problema

Nuestro sistema se centrará en reconocer, a partir de un audio, la nota del piano que se está tocando. Escogimos este instrumento por la cantidad de recursos que podemos encontrar y por su naturaleza invariable al ser tocada por una u otra persona.

Dada la naturaleza de nuestro sistema, descartamos utilizar la especificidad o la sensibilidad ya que nos es indiferente las clases en las que se clasifiquen las notas (el coste de un falso positivo o un falso negativo es el mismo).

Así, como solo nos interesa una correcta clasificación global, pensamos en utilizar la precisión, la cual sigue la fórmula:

$$Precision = \frac{VN + VP}{(VN + FN + VP + FP)} \tag{1}$$

El incoveniente de esta métrica está en que si tenemos un conjunto de patrones desbalanceado (gran diferencia en el número de patrones positivos y negativos), la precisión podría alcanzar valores muy altos con sistemas que clasifiquen todos los patrones en la clase con mayor número de ellos en el entrenamiento.

La métrica que utilizaremos entonces para valorar los resultados obtenidos y que palia los problemas mencionados anteriormente será la **F1-score**. Esta se corresponde con la media armónica de la sensibilidad y el valor predictivo positivo y está caracterizada por la fórmula:

$$F1 - Score = \left(\frac{Sensibilidad^{-1} + VPP^{-1}}{2} \right)^{-1} \tag{2}$$

donde,

$$Sensibilidad = \frac{VP}{(FN + VP)} \tag{3}$$

$$VPP = \frac{VP}{(VP + FP)}, \tag{4}$$

Quizá el único inconveniente de esta métrica es su difícil interpretación, más allá de comparar los valores que ofrecen diferentes sistemas. El valor más alto que puede alcanzar es de 1 (todos los patrones se clasificaron correctamente) y el más bajo es de 0 (todos los patrones se clasificaron incorrectamente). Entonces, para facilitar la interpretación de la métrica, también se mostrará la precisión de cada modelo, pero se utilizará el F1-Score para determinar el mejor modelo.

3.1. Restricciones

Como única restricción, en dicho audio solo puede haber una nota sonando a la vez para que el sistema sea capaz de reconocerla correctamente.

3.2. Características

La base de datos con la que contamos tiene un total de 5.406 audios con una media de más de 50 *samples* por cada una de las 85 notas del piano (a partir de *C1*), tocadas desde posiciones e intensidades distintas y grabadas con micrófonos diferentes. Dichos *samples* están en formato *.wav* en estéreo, con un *bitrate* de *2304kbps*, *24 bits per sample* y un *sample rate* de *48kHz*. Todo ello ocupa un total de *34.5GB* en disco. La duración media de los *samples* es de aproximadamente 6 segundos.

El origen de la base de datos es una librería de piano de la compañía *FluffyAudio (https://www.fluffyaudio.com/shop/scoringpiano)* grabada en 2016 y pensada para jazz, música clásica y bandas sonoras.

4. Análisis bibliográfico

Para profundizar en el tema antes de abordarlo, en esta sección se analizan diferentes artículos científicos relacionados con la inteligencia artificial y el reconocimiento de audio, ya sea específicamente relacionado o no con el mundo del reconocimiento de piezas o notas musicales.

Trabajos como el de Osmalsky y col., 2012 nos aportan nuevos enfoques, en el que, en lugar de detectar las notas por separado, analizan todo el espectro de frecuencias para poder reconocer acordes completos de diferentes instrumentos; utilizan una técnica llamada Pitch Class Profile (PCP), que obtiene las relaciones energéticas de cada nota en la escala a partir de un audio.

Además, como resumen Benetos y col., 2018, a pesar del estado avanzado de la transcripción automática de canciones, aún están presentes retos tales como la independencia de los intrumentos, de los estilos musicales o la interpretación de la expresividad.

Otros trabajos mas antiguos como los de Foo y Wong, 1999, describen un algoritmo capaz de reconocer notas de un piano a partir de piezas sintetizadas o acústicas, que son digitalmente muestreadas y transformadas al dominio de frecuencia usando la transformada de Q constante a partir de la cual se la aplican diferentes técnicas para identificar las notas.

En un terreno más general, trabajos como el de Chang y col., 2017, exploran la identificación de *covers* utilizando estructuras más novedosas que no desarrollaremos en este trabajo como son las Redes de Neuronas Convolucionales. Las salidas de este sistema corresponderían con la probabilidad de ser una *cover* (comparándola con la canción original), y se ordenarían por dicha probabilidad elaborando un ranking.

También encontramos la tesis de Klapuri, 2004 que propone un sistema capaz de generar una representación simbólica a partir de un audio, centrándose en el desarrollo

de los algoritmos que pueden ser usados para detectar y observar sonidos harmónicos en señales polifónicas. Los cuales fueron evaluados aplicandolos en un programa de transcripción de música para piano implementado y simulado en un entorno Matlab.

En el trabajo de Marolt y col., 2002 destacamos el uso de sistemas conexionistas para la detección de inicio en señales musicales basandose en la combinacion de un banco de filtros auditivos , un red neuronal pulsante y un perceptron multicapa. Este uso de sistemas conexionistas también es empleado para sistemas de transcripción de musica de piano polifónico, mostrando ciertas ventajas sobre otros métodos debido a su estructura y presentándose como una alternativa viable a los algoritmos ya existentes

El uso de redes de neuronas artificiaes también están presentes en trabajos como Solanki y Pandey, 2019, que aborda la identificación de los instrumentos que forman part de piezas polifónicas. Utiliza una red de neuronas convolucional de 8 capas y se apoya en los espectrogramas MEL para mapear datos del audio.

Para finalizar, ya en un campo algo más alejado del musical, podemos destacar el trabajo elaborado por Baevski y col., 2021, el cual profundiza en el campo de reconocimiento del habla mediante inteligencia artificial. A diferencia de otros sistemas de reconocimiento, este trabajo no usa datos etiquetados que limiten el reconocimiento a un grupo reducido de idiomas. Esta técnica necesita menos requerimientos, aprovechando representaciones auto supervisadas del habla para segmentar el audio y aprender a mapear desde estas representaciones a fonemas via *Adversarial Training*.

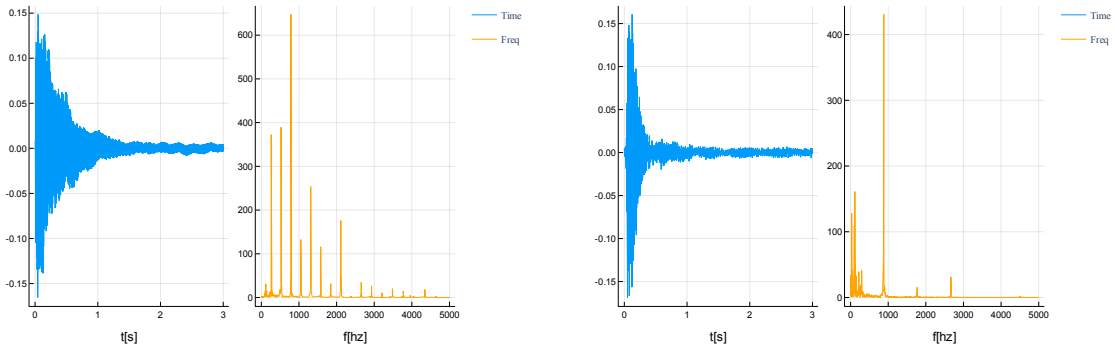
5. Desarrollo

Para el desarrollo de este sistema utilizaremos un método basado en aproximaciones, es decir, comenzaremos acotando el problema e iremos aumentando la complejidad a medida que obtenemos resultados satisfatorios.

5.1. Primera aproximación

5.1.1. Descripción

En esta primera aproximación nos limitaremos a diferenciar únicamente entre dos notas. Escogimos las notas *C4* y *A5* de los cuales usaremos 92 y 56 audios de cada una, respectivamente, en total se utilizarán 148 patrones. En las figuras 1a y 1b podemos ver a la izquierda la señal en tiempo y, a la derecha, la señal en el dominio de la frecuencia.



(a) C4 (b) A5

Figura 1: Notas en tiempo y frecuencia

Dado que cada muestra tiene una duración diferente, decidimos quedarnos solo con los 3 primeros segundos de cada audio (que es donde está la información más relevante de la nota) y, para la frecuencia, acotaremos cada señal entre 0 y 5000 Hz. Ya que la frecuencia máxima que se alcanza en el piano clásico es de 4186 Hz (*C8*) y la más baja, en nuestra base de datos, es de 32.7Hz (*C1*), escogimos un rango de frecuncias ampliado debido a que es posible que exista información que nos ayude a identificar la nota.

Para los experimentos, se va a realizar ***cross-validation*** con cada modelo, para así poder evaluar el rendimiento de cada uno, con independencia de la aleatoriedad que se pueda producir al seleccionar un subconjunto de patrones para test. En concreto, se realizará un ***cross-validation*** con **10** ***folds***. El valor del F1-Score y de la precisión del modelo será el promedio de dichos valores obtenidos en los *folds*.

Además, como las notas de un mismo instrumento se diferencian principalmente por la frecuencia (una nota más aguda tiene una mayor frecuencia y una más grave, una

menor), calcularemos de las señales en el dominio del tiempo su relación en el dominio de la frecuencia utilizando la Transformada de Fourier y, posteriormente, extraeremos las siguientes características:

- **Energia media de la señal**: ya que las señales con más frecuencia (más agudas) tienen más energía, esta característica nos podría ayudar a diferenciar entre notas tocadas con la misma intensidad.
- **Media, desviación tipica, y valor máximo de la intensidad en el dominio de la frecuencia, en intervalos no uniformes**: la frecuencia de cada nota aumenta de forma exponencial siguiendo la fórmula:

$$f_{i+1} = f_i \cdot (\sqrt[12]{2}), f_0 = 27{,}5Hz \tag{5}$$

  Por ello decidimos dividir el espectro en **10 intervalos** de longitud variable siguiendo dicha distribución. Podríamos entonces obtener un intervalo donde la media y la desviación típica de la frecuencia fueran más elevados que el resto, ayudándonos a identificar la nota.

  Los intervalos que usaremos son los siguientes:

  (0.0, 380.3), (380.3, 783.21), (783.21, 1210.08), (1210.08, 1662.33), (1662.33, 2141.47), (2141.47, 2649.11), (2649.11, 3186.93), (3186.93, 3756.73), (3756.73, 4360.42) y (4360.42, 5000.0)

- **Zero-crossing/s**: esta característica determina las veces que la señal, en el dominio del tiempo, toma el valor 0 por segundo. Como dicha cracterística nos proporcionaría valores similares a la frecuencia media de la señal, podría contribuír a su correcta clasificación.

En lo referente al preprocesado de los datos, hemos decidido optar por una **normalización de media cero**, debido a que los datos no están acotados en un intervalo y pueden tomar cualquier valor, ya sea en tiempo o en frecuencia. Aunque las frecuencias de las notas sí que están acotadas en un intervalo, en la señal puede haber ruido que sobrepase los valores típicos, por lo que una normalización entre mínimo y máximo no sería lo adecuado.

Así, obtuvimos para cada característica, la media y la desviación típica. Se muestran dichos datos en la Tabla 1.

Tabla 1: Parámetros de normalización para cada característica

Característica	Media	Desviación Típica
E	0,0045678928	0,0035726614
zero-crossing	1797,2139	1352,7441
m_1	16,821161	11,616611
m_2	9,4079	5,8349323
m_3	14,607474	11,515968
m_4	3,8720663	3,5469663
m_5	2,3520973	2,1539617
m_6	0,48698702	0,6487291
m_7	0,7153551	1,0973071
m_8	0,40077275	0,500489
m_9	0,22367951	0,3412378
m_{10}	0,23234393	0,5112222
std_1	54,282375	46,679535
std_2	31,582827	27,185955
std_3	57,896194	38,439426
std_4	11,108885	12,696077
std_5	7,0041313	6,732649
std_6	0,52131796	0,72755563
std_7	2,7318008	4,6750984
std_8	0,9234557	1,2807367
std_9	0,47755784	0,89846444
std_{10}	0,5695597	1,2890217
max_1	832,8222	779,30615
max_2	567,7255	505,1785
max_3	901,9232	548,2265
max_4	192,80989	237,47154
max_5	98,938354	103,55783
max_6	4,392064	6,0777674
max_7	40,596066	65,99385
max_8	11,672165	15,524432
max_9	5,7904434	10,440257
max_{10}	5,6131926	11,565159

Donde:

- E indica la **energía de la señal**.
- m_i indica la **media de la intensidad** en el intervalo i en frecuencia.
- std_i indica la **desviación típica de la intensidad** en el intervalo i en frecuencia.
- max_i indica el **máximo de la intensidad** en el intervalo i en frecuencia.

Se van a realizar experimentos sobre los siguientes modelos: RNA, SVM, Árboles de decisión y kNN.

Además, como el entrenamiento de una RNA es un proceso no determinista, en cada *fold* tendremos que entrenar la RNA varias veces, y promediar los resultados. Para estos experimentos, se ha repetido 30 veces el entrenamiento de la RNA en cada *fold*, devolviendo el resultado promedio de esas 30 ejecuciones.

Para poder evaluar el modelo con mejor rendimiento, como ya se ha dicho antes, utilizaremos como medida el promedio del **F1-Score** en los *folds*. Si hay empate en dicha media, se elegirá el modelo con menor desviación típica de la métrica.

5.1.2. Resultados

Para permitir la reproducibilidad de los experimentos, se utilizó *100* como semilla de generación de números aleatorios.

5.1.2.1 RNA

Para el entrenamiento de las RNA, se han utilizado los siguientes parámetros en común:

- Ratio de aprendizaje: 0,01
- Ratio de patrones para el conjunto de validación: 0,2.
- Máximo número de ciclos (*epochs*) de entrenamiento: 1500.
- Máximo número de ciclos (*epochs*) de entrenamiento sin mejorar el error de validación: 5.
- Algoritmo de optimización: ADAM.
- Función de *loss*: *Binary Cross Entropy*.

El hiperparámetro que hemos variado ha sido la arquitectura de neuronas a utilizar. Utilizamos 8 arquitecturas distintas, donde la arquitectura $[i]$ denota una RNA con una única capa oculta con i neuronas y la arquitectura $[i, j]$ denota una neurona con dos capas ocultas; con i neuronas en la primera capa oculta y j neuronas en la segunda capa oculta.

Se muestran los resultados de la experimentación en la Tabla 2

5.1.2.2 SVM

Para las máquinas de soporte vectorial hemos utilizado 8 configuraciones distintas de los hiperparámetros de *kernel*, grado del *kernel* (sólo se utiliza en *kernel* polinomial), *gamma* y *C*.

Se muestran los resultados de la experimentación en la Tabla 3

Tabla 2: Resultados RRNNA

Arquitectura	F1-Score	Precisión
[2]	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
[4]	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
[8]	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
[2, 4]	$0{,}9119 \pm 0{,}12531$	$0{,}96487 \pm 0{,}05054$
[2, 8]	$0{,}90667 \pm 0{,}14724$	$0{,}96524 \pm 0{,}05412$
[4, 2]	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
[4, 4]	$0{,}99 \pm 0{,}0225$	$0{,}99643 \pm 0{,}00804$
[4, 8]	$0{,}9621 \pm 0{,}0974$	$0{,}97934 \pm 0{,}05699$

Tabla 3: Resultados SVM

(kernel, grado, gamma, C)	F1-Score	Precisión
(poly, 3, 1, 1)	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
(rbf, 3, 1, 1)	$0{,}93642 \pm 0{,}07368$	$0{,}94119 \pm 0{,}07166$
(sigmoid, 3, 1, 1)	$0{,}71875 \pm 0{,}15823$	$0{,}77143 \pm 0{,}11289$
(poly, 5, 10, 0,1)	$0{,}99 \pm 0{,}03162$	$0{,}98571 \pm 0{,}04518$
(rbf, 3, 10, 0,1)	$0{,}76689 \pm 0{,}01542$	$0{,}62214 \pm 0{,}02026$
(sigmoid, 3, 10, 0,1)	$0{,}70059 \pm 0{,}1406$	$0{,}77768 \pm 0{,}06262$
(rbf, 3, 0,01, 100)	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
(rbf, 3, 100, 0,01)	$0{,}76689 \pm 0{,}01542$	$0{,}62214 \pm 0{,}02026$

5.1.2.3 Árboles de decisión

Para los árboles de decisión hemos utilizado 6 configuraciones distintas del hiperparámetro de profundidad máxima.

Se muestran los resultados de la experimentación en la Tabla 4

Tabla 4: Resultados Árbol de decisión

Altura máxima	F1-Score	Precisión
2	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
4	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
8	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
16	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
32	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
64	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$

5.1.2.4 kNN

Para el modelo kNN hemos utilizado 6 configuraciones distintas del hiperparámetro de k (número de vecinos más cercanos).

Se muestran los resultados de la experimentación en la tabla 5

Tabla 5: Resultados kNN

K	F1-Score	Precisión
2	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
3	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
5	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
8	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
13	$1{,}0 \pm 0{,}0$	$1{,}0 \pm 0{,}0$
21	$0{,}99091 \pm 0{,}02875$	$0{,}99286 \pm 0{,}02259$

5.1.3. Discusión

Los resultados han sido muy buenos, ya que en la mayoría de casos de alcanza una **precisión** y un **F1-Score** del 100 % o muy alto.

En general, los sistemas se equivocan en muy pocos patrones. Esto se puede deber a que el problema elegido en esta aproximación era excesivamente sencillo para el elevado número de características escogidas, permitiéndonos separar perfectamentes los patrones en las dos clases.

En lo referente a las características, por un parte estas han sido correctamente elegidas, ya que alcanzamos una precisión muy alta. Pero por otra parte, el hecho de que algunos sistemas concretos no funcionen tan bien (por ejemplo, algunos SVM), se podría deber al elevado número de características, ya que algunas de ellas pueden ser innecesarias o hasta contraproducentes en este sencillo problema. Una buena estrategia sería el intentar reducir el número de características en futuras aproximaciones, pero hay que tener cuidado, ya que si aumentamos el número de clases, podría darse el caso de haber eliminado una característica necesaria para una correcta clasificación.

En concreto, las características que creemos que son las que producen mejores resultados son las que se refieren a los intervalos en frecuencia de *(0.0, 380.3)* y *(783.21, 1210.08)*, debido a que es en ellos donde se sitúan las frecuencia de las notas *C4* y *A5*, respectivamente. Las características que se corresponden a dichos intervalos son m_1, std_1, max_1 para el primer intervalo y m_3, std_3, max_3 para el segundo intervalo.

Otra característica que podría funcionar sería la frecuencia en la que se alcanza el pico absoluto en intensidad de la señal, o la media de las K frecuencias con mayor intensidad, sin tener en cuenta la división en intervalos. También se podría aumentar el número de intervalos en los que obtener las características, para poder así separar mejor las notas con frecuencias más cercanas.

Todos los algoritmos tienen al menos una configuración con un **F1-Score** de 1,0±0,0, por lo que es complicado encontrar uno que sea el mejor.

Podemos destacar la RNA con una arquitectura de una única capa oculta con 2 neuronas, ya que es un modelo muy sencillo y podemos esperar que, cuanto más sencillo sea un modelo, mayor será también su capacidad de generalización.

En la Figura 2 podemos ver un gráfico del entrenamiento de dicha RNA, donde observamos el error cometido en los conjuntos de test, validación y entrenamiento para cada ciclo de entrenamiento. Este se ha realizado utilizando todos los patrones, separando un 20 % para el conjunto de test. Además, se puede ver que se ha producido una parada temprana, ya que el sistema no ha realizado el máximo número de ciclos de entrenamiento (1500 ciclos) y se puede distinguir también como la curva de la gráfica desciende muy rápido. Esto se puede deber a que tenemos un ratio de aprendizaje alto para este problema concreto.

En la Tabla 6 podemos ver una matriz de confusión para la RNA entrenada anteriormente. Las predicciones se hicieron sobre un conjunto de test que supone un 20 % de todos los patrones utilizados. El 80 % de los patrones restantes se utilizó para el entrenamiento de la RNA, ya que el conjunto de entrenamiento no se debería utilizar para realizar predicciones.

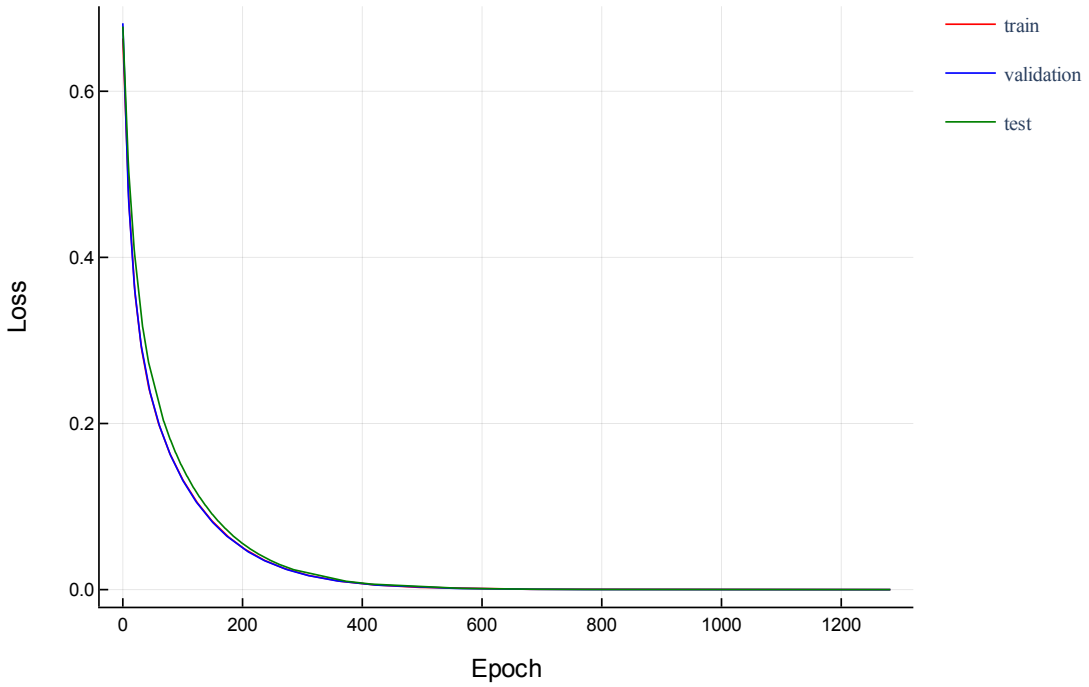


Figura 2: Entrenamiento RNA

		Predicc. A5	Predicc. C4
Real	A5	15	0
	C4	0	15

Tabla 6: Matriz de confusión RNA

Pero aunque se alcance ya un buen resultado con un modelo sencillo como puede ser la RNA anterior, caben destacar los árboles de decisión, que son modelos que pueden ofrecer explicabilidad sobre sus predicciones, además de presentar un entrenamiento determinista, por lo que si tenemos que elegir el mejor modelo, nos decantaríamos por este (en concreto, el de altura máxima de 2, ya que es el más sencillo), ya que presenta también un 100 % de **F1-Score**.

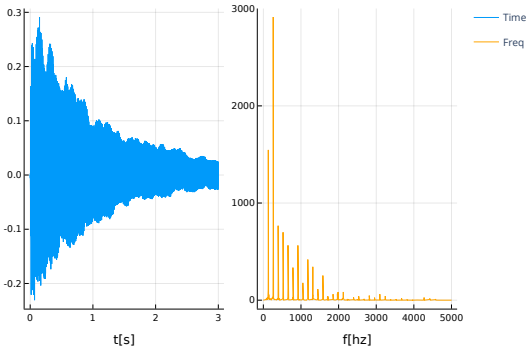
Aunque si deseamos un modelo con una base matemática más fuerte, en vez de un modelo con una base estadística como puede ser un árbol de regresión, utilizar una máquina de soporte vectorial con un *kernel* polinomial de grado 3, y ambos valores de *gamma* y C de 1, también sería una buena opción.

Los algoritmos que han funcionado peor han sido ciertas configuraciones de los SVM, en concreto las que utilizan un *kernel* sigmoidal. Entonces, utilizar un *kernel* sigmoidal no parece ofrecer buenos resultados para estos patrones.

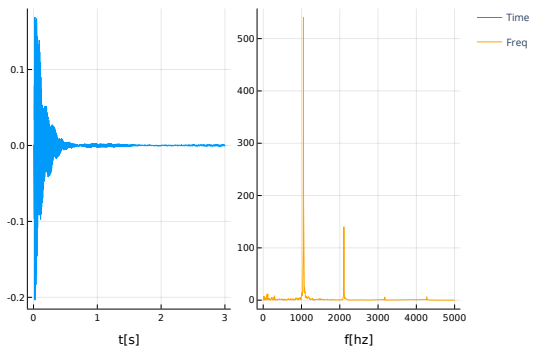
5.2. Segunda aproximación

Como hemos obtenido resultados muy buenos en la aproximación anterior, se va a aumentar la complejidad del problema, es decir, añadiremos más notas a ser clasificadas. Se clasificarán entonces 15 notas, y se dispondrá de 1090 patrones para ser utilizados por el sistema.

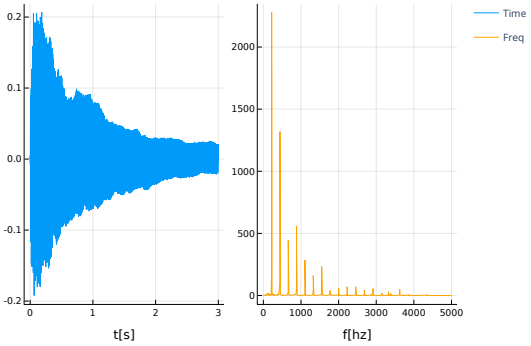
Las notas que vamos a clasificar son: *C1*, *C2*, *C3*, *C4*, *C5*, *C6*, *C7*, *C8*, *A1*, *A2*, *A3*, *A4*, *A5*, *A6* y *A7*.



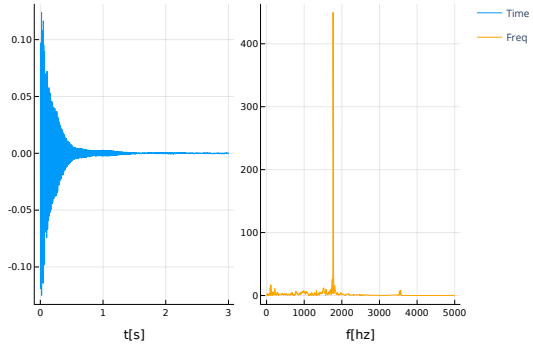
(a) C3



(b) A6



(a) A3



(b) A6

Como hemos aumentado mucho el número de clases a clasificar, aumentaremos también el número de características para poder diferenciar bien las clases que sean más similares.

En concreto, añadiremos las siguientes características:

- **Frecuencia a la que se alcanza el máximo de intensidad de la señal en frecuencia** en cada uno de los 10 intervalos que ya se especificaron en la aproximación anterior.

No se producen cambios sobre las características que fueron incorporadas en la aproximación anterior.

Para el preprocesado de los datos, utilizaremos, como se mencionó en la aproximación anterior, una normalización de media cero. Así, obtuvimos para cada característica, la media y la desviación típica. Se muestran dichos datos en las tablas 7 y 8.

Tabla 7: Parámetros de normalización (I)

Característica	Media	Desviación Típica
E	0,0050883736	0,0088912
zero-crossing	3743,606	5035,344
m_1	15,578591	18,397396
m_2	11,147778	9,179351
m_3	9,6564455	9,295277
m_4	4,0943613	4,225765
m_5	3,6645029	4,699565
m_6	1,0346977	1,3888823
m_7	0,71540374	1,0062366
m_8	0,68384236	1,307346
m_9	0,38978294	0,8350492
m_{10}	0,44757652	1,4078437
std_1	49,61187	74,658005
std_2	39,12518	41,206196
std_3	29,023785	34,23634
std_4	8,560635	11,001557
std_5	9,463276	12,920476
std_6	1,5626754	2,6529887
std_7	1,6301519	2,8170295
std_8	1,2500806	2,295387
std_9	0,5342404	0,98041487
std_{10}	0,7347808	2,2108128
max_1	944,37256	1596,2578
max_2	748,07245	844,44037
max_3	442,8393	524,42633
max_4	130,52594	192,10622
max_5	129,60143	179,40135
max_6	19,159048	36,38486
max_7	22,589241	40,29418
max_8	13,267355	21,899231
max_9	5,3473926	10,327133
max_{10}	5,520794	13,594501

Donde:

Tabla 8: Parámetros de normalización (II)

Característica	Media	Desviación Típica
max-freq$_1$	174,89925	87,058235
max-freq$_2$	521,6259	125,11409
max-freq$_3$	961,683	100,904564
max-freq$_4$	1394,9509	126,07898
max-freq$_5$	1897,9567	174,84656
max-freq$_6$	2353,7712	191,61922
max-freq$_7$	2791,1377	166,82562
max-freq$_8$	3426,2026	169,13255
max-freq$_9$	4093,2415	226,97453
max-freq$_{10}$	4583,3877	193,1689

- E indica la **energía de la señal**.
- m_i indica la **media de la intensidad** en el intervalo i en frecuencia.
- std_i indica la **desviación típica de la intensidad** en el intervalo i en frecuencia.
- max_i indica el **máximo de la intensidad** en el intervalo i en frecuencia.
- *max-freq*$_i$ indica la **frecuencia donde se alcanza el máximo de intensidad** en el intervalo i en frecuencia.

Para estos experimentos se ha repetido 20 veces el entrenamiento de la RNA en cada *fold*, devolviendo el resultado promedio de esas 20 ejecuciones.

Para poder evaluar el modelo con mejor rendimiento, como ya se ha dicho antes, utilizaremos como medida el promedio del **F1-Score** en los *folds*. Si hay empate en dicha media, se elegirá el modelo con menor desviación típica de la métrica.

5.2.1. Resultados

Para permitir la reproducibilidad de los experimentos, se utilizó *100* como semilla de generación de números aleatorios.

5.2.1.1 RNA

Para el entrenamiento de las RNA, se han utilizado los siguientes parámetros en común:

- Ratio de aprendizaje: 0,01
- Ratio de patrones para el conjunto de validación: 0,2.
- Máximo número de ciclos (*epochs*) de entrenamiento: 1500.
- Máximo número de ciclos (*epochs*) de entrenamiento sin mejorar el error de validación: 5.
- Algoritmo de optimización: ADAM.
- Función de *loss*: *Cross Entropy*.

El hiperparámetro que hemos variado ha sido la arquitectura de neuronas a utilizar. Utilizamos 8 arquitecturas distintas, donde la arquitectura $[i]$ denota una RNA con una única capa oculta con i neuronas y la arquitectura $[i, j]$ denota una neurona con dos capas ocultas; con i neuronas en la primera capa oculta y j neuronas en la segunda capa oculta.

Se muestran los resultados de la experimentación en la Tabla 9

Tabla 9: Resultados RNA

Arquitectura	F1-Score	Precisión
[8]	$0{,}92977 \pm 0{,}01923$	$0{,}95523 \pm 0{,}0131$
[16]	$0{,}96092 \pm 0{,}0163$	$0{,}97591 \pm 0{,}0112$
[32]	$0{,}96305 \pm 0{,}01805$	$0{,}97745 \pm 0{,}01137$
[8, 8]	$0{,}87808 \pm 0{,}01593$	$0{,}92368 \pm 0{,}00628$
[16, 4]	$0{,}76012 \pm 0{,}03119$	$0{,}8518 \pm 0{,}0233$
[16, 8]	$0{,}92264 \pm 0{,}02515$	$0{,}95168 \pm 0{,}01708$
[32, 16]	$0{,}9506 \pm 0{,}01989$	$0{,}97044 \pm 0{,}01033$
[32, 32]	$0{,}95122 \pm 0{,}02256$	$0{,}97004 \pm 0{,}013$

5.2.1.2 SVM

Para las máquinas de soporte vectorial hemos utilizado 8 configuraciones distintas de los hiperparámetros de *kernel*, grado del *kernel* (sólo se utiliza en *kernel* polinomial), *gamma* y *C*.

Se muestran los resultados de la experimentación en la Tabla 10

Tabla 10: Resultados SVM

(kernel, grado, gamma, C)	F1-Score	Precisión
(poly, 3, 1, 1)	0,97261 ± 0,02358	0,98158 ± 0,01438
(rbf, 3, 1, 1)	0,67559 ± 0,04297	0,69947 ± 0,05196
(sigmoid, 3, 1, 1)	0,28516 ± 0,03747	0,3575 ± 0,03929
(poly, 3, 10, 0,1)	0,97619 ± 0,02456	0,98415 ± 0,01345
(rbf, 3, 10, 0,1)	0,01607 ± 0,00059	0,13703 ± 0,00568
(sigmoid, 3, 10, 0,1)	0,23652 ± 0,03974	0,33481 ± 0,04109
(rbf, 3, 0,01, 100)	0,97415 ± 0,02285	0,98444 ± 0,0122
(poly, 3, 100, 0,001)	0,9678 ± 0,03578	0,97987 ± 0,01547

5.2.1.3 Árboles de decisión

Para los árboles de decisión hemos utilizado 6 configuraciones distintas del hiperparámetro de profundidad máxima.

Se muestran los resultados de la experimentación en la Tabla 11

Tabla 11: Resultados Árbol de decisión

Altura máxima	F1-Score	Precisión
4	0,44767 ± 0,00668	0,66703 ± 0,01287
8	0,86809 ± 0,02258	0,92653 ± 0,01909
16	0,96505 ± 0,02104	0,9725 ± 0,0143
32	0,96786 ± 0,01849	0,97302 ± 0,01609
64	0,97438 ± 0,01244	0,97779 ± 0,0102
128	0,97415 ± 0,0161	0,9779 ± 0,01186
256	0,9672 ± 0,02388	0,97452 ± 0,0172

5.2.1.4 kNN

Para el modelo kNN hemos utilizado 6 configuraciones distintas del hiperparámetro de k (número de vecinos más cercanos).

Se muestran los resultados de la experimentación en la tabla 12

Tabla 12: Resultados kNN

K	F1-Score	Precisión
2	$0{,}95086 \pm 0{,}01526$	$0{,}97073 \pm 0{,}0088$
3	$0{,}9323 \pm 0{,}03125$	$0{,}95942 \pm 0{,}0156$
5	$0{,}90323 \pm 0{,}03494$	$0{,}94503 \pm 0{,}01712$
8	$0{,}86641 \pm 0{,}03906$	$0{,}93135 \pm 0{,}0199$
13	$0{,}84065 \pm 0{,}02462$	$0{,}90985 \pm 0{,}01911$
21	$0{,}80344 \pm 0{,}03804$	$0{,}88737 \pm 0{,}02667$

5.2.2. Discusión

En general los resultados son muy buenos, ya que hay varios modelos con un alto F1-Score. El modelo que mejor funciona es un SVM con función de *kernel* polinomial de grado 3, con un *gamma* y C de 10 y 0,1, respectivamente.

Este modelo tiene un F1-Score de $0{,}97619 \pm 0{,}02456$, con una precisión de $0{,}98415 \pm 0{,}01345$ en los folds. De este resultado podemos concluír que nuestro sistema funciona muy bien, ya que clasificar 15 notas es un problema relativamente complejo.

Cabe destacar modelos como una RNA con una única capa oculta de 32 neuronas; un svm con *kernel* de base radial, con los parámetros de *gamma* y C de 0,01 y 100, respectivamente; un árbol de decisión con altura máxima de 64; y un modelo kNN con un k de 2. Estos modelos también presentan muy buenos resultados, aunque no los mejores.

El modelo con peor rendimiento es un SVM con kernel de base radial con *gamma* y C de 10 y 0,1, respectivamente. Esta configuración del modelo tiene resultado increiblemente bajo (*F1-Score* de $0{,}01607 \pm 0{,}00059$), pero desconocemos a qué se puede deber.

En cuanto a las características, creemos que fue un acierto añadir la frecuencia a la que se encuentra el pico de intensidad en cada intervalo, ya que es una característica muy representativa de la señal y que sirve para diferenciar mejor las notas. Sin estas nuevas características, los resultados no serían tan buenos.

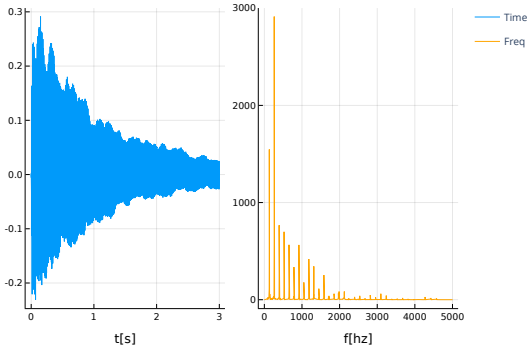
El problema que podemos observar en la matriz de confusión para el mejor sistema es que el sistema de equivoca en las notas más cercanas, como pueden ser A1, A2, C1 y C2, notas que tienen una frecuencia muy similar y se encuentran en el primer intervalo, *(0.0, 380.3)*.

Para poder diferenciar mejor entre estas notas, y entre futuras notas que puedan ser similares entre ellas, sería una buena opción que se aumentase el número de intervalos a dividir las frecuencias, para así poder diferenciar de mejor manera notas similares.

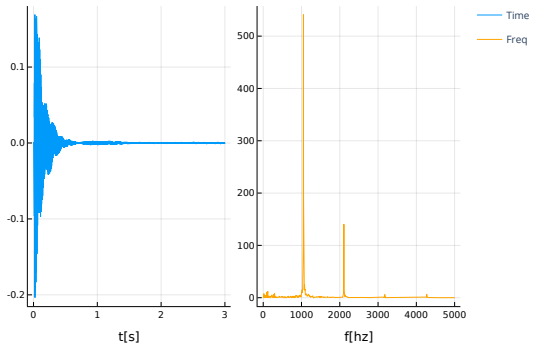
5.3. Tercera aproximación

Como hemos obtenido resultados muy buenos en la aproximación anterior, se va a aumentar la complejidad del problema, es decir, añadiremos más notas a ser clasificadas. Serán un total de 50 notas a clasificar, y se dispondrá de 3194 patrones para ser utilizados por los modelos.

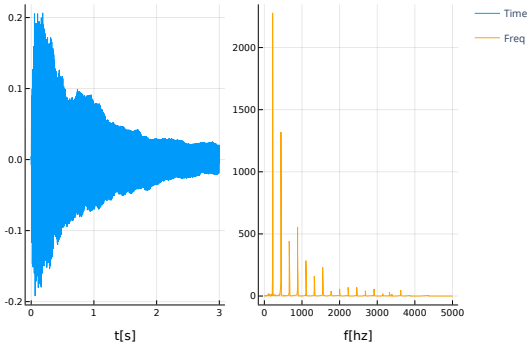
Las notas que vamos a clasificar son: *C1*, *C2*, *C3*, *C4*, *C5*, *C6*, *C7*, *C8*, *A1*, *A2*, *A3*, *A4*, *A5*, *A6*, *A7*, *B1*, *B2*, *B3*, *B4*, *B5*, *B6*, *B7*, *D1*, *D2*, *D3*, *D4*, *D5*, *D6*, *D7*, *E1*, *E2*, *E3*, *E4*, *E5*, *E6*, *E7*, *F1*, *F2*, *F3*, *F4*, *F5*, *F6*, *F7*, *G1*, *G2*, *G3*, *G4*, *G5*, *G6* y *G7*.



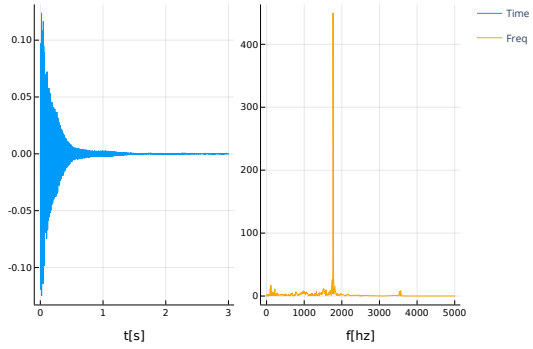
(a) C3



(b) A6



(a) A3



(b) A6

Como en esta aproximación aumentamos mucho el número de clases a clasificar (50 clases), nos parece buena opción aumentar las características a utilizar, pero de una forma distinta a la anterior aproximación. Lo que haremos será aumentar el número de intervalos a dividir el dominio de la frecuencia; de 10 intervalos que teníamos anteriormente, a 30 intervalos. De esta manera, esperamos que se puede clasificar mejor notas que tengan frecuencias muy cercanas, además de repartir mejor el número de notas que coinciden en cada intervalos.

En concreto, los intervalos de frecuencias que utilizaremos serán los siguientes:
(0.0, 63.845),*(63.845, 131.486)*, *(131.486, 203.149)*,*(203.149, 279.074)*, *(279.074, 359.513)*,*(359.513, 444.735)*, *(444.735, 535.025)*,*(535.025, 630.684)*, *(630.684, 732.032)*, *(732.032, 839.405)*, *(839.405, 953.163)*, *(953.163, 1073.686)*, *(1073.686, 1201.376)*, *(1201.376, 1336.658)*, *(1336.658, 1479.984)*, *(1479.984, 1631.833)*, *(1631.833, 1792.712)*, *(1792.712, 1963.157)*, *(1963.157, 2143.737)*, *(2143.737, 2335.055)*, *(2335.055, 2537.749)*, *(2537.749, 2752.496)*, *(2752.496, 2980.013)*, *(2980.013, 3221.059)*, *(3221.059, 3476.437)*, *(3476.437, 3747.002)*, *(3747.002, 4033.655)*, *(4033.655, 4337.353)*, *(4337.353, 4659.11)* y *(4659.11, 5000.0)*

Para el preprocesado de los datos, utilizaremos, como se mencionó en aproximaciones anteriores, una normalización de media cero. Así, obtuvimos para cada característica, la media y la desviación típica. Se muestran dichos datos en las tablas 13, 14 15 y 16,

Donde:

- E indica la **energía de la señal**.
- m_i indica la **media de la intensidad** en el intervalo i en frecuencia.
- std_i indica la **desviación típica de la intensidad** en el intervalo i en frecuencia.
- max_i indica el **máximo de la intensidad** en el intervalo i en frecuencia.
- $max\text{-}freq_i$ indica la **frecuencia donde se alcanza el máximo de intensidad** en el intervalo i en frecuencia.

Para estos experimentos se ha repetido 20 veces el entrenamiento de la RNA en cada *fold*, devolviendo el resultado promedio de esas 20 ejecuciones.

Para poder evaluar el modelo con mejor rendimiento, como ya se ha dicho anteriormente, utilizaremos como medida el promedio del **F1-Score** en los *folds*. Si hay empate en dicha media, se elegirá el modelo con menor desviación típica de la métrica.

Tabla 13: Parámetros de normalización (I)

Característica	Media	Desviación Típica
E	0,0054126005	0,009019938
zero-crossing	3397,5913	4540,8813
m_1	2,5406487	5,2477403
m_2	20,31295	34,20369
m_3	18,736494	33,777767
m_4	19,506176	27,648252
m_5	17,283388	26,778984
m_6	14,071337	21,961533
m_7	15,653144	21,659325
m_8	7,2894015	11,942419
m_9	10,673911	22,342806
m_{10}	8,886631	17,860262
m_{11}	8,768464	14,431124
m_{12}	9,399654	13,1116905
m_{13}	7,141891	16,054686
m_{14}	5,671832	9,146762
m_{15}	4,4691596	7,688136
m_{16}	4,622483	7,50842
m_{17}	2,8246067	5,453456
m_{18}	2,1192815	3,5254076
m_{19}	3,2075806	5,558346
m_{20}	1,3101652	2,2760432
m_{21}	1,5997506	4,51931
m_{22}	1,4330753	3,7737772
m_{23}	0,9662523	2,9858894
m_{24}	0,79353756	1,8264828
m_{25}	0,48343834	0,8716932
m_{26}	0,6009706	1,4477166
m_{27}	0,3706837	0,93128383
m_{28}	0,41856125	1,3392309
m_{29}	0,38214317	1,6580197
m_{30}	0,13477807	0,25888646

Tabla 14: Parámetros de normalización (II)

Característica	Media	Desviación Típica
std_1	3,777772	9,5573845
std_2	43,746407	101,179634
std_3	44,765743	106,52451
std_4	44,928486	86,98192
std_5	36,79204	74,466835
std_6	34,05622	69,12505
std_7	35,642204	58,005997
std_8	14,239829	31,195595
std_9	20,148722	46,739002
std_{10}	18,305641	41,97697
std_{11}	15,096799	33,64003
std_{12}	17,944155	31,481377
std_{13}	14,973824	42,54836
std_{14}	10,101966	21,576153
std_{15}	7,4685116	17,61445
std_{16}	7,9542217	17,202675
std_{17}	5,18349	11,849964
std_{18}	2,358577	5,2823844
std_{19}	6,350091	12,173027
std_{20}	1,757622	4,281193
std_{21}	2,7778556	8,633124
std_{22}	2,4016538	6,636576
std_{23}	1,6487495	5,640244
std_{24}	1,3393545	3,0001454
std_{25}	0,606615	1,4506007
std_{26}	0,9203455	2,2596686
std_{27}	0,4674074	1,0911392
std_{28}	0,64302534	2,1712365
std_{29}	0,45122066	1,628174
std_{30}	0,19327588	0,4538704

Tabla 15: Parámetros de normalización (III)

Característica	Media	Desviación Típica
$\max_1$	33,78178	108,59057
$\max_2$	448,6926	1190,1057
$\max_3$	473,44827	1207,2843
$\max_4$	464,8955	994,3883
$\max_5$	362,47415	770,89215
$\max_6$	343,7377	698,5618
$\max_7$	339,793	571,1343
$\max_8$	146,53598	331,86856
$\max_9$	187,07274	410,4047
$\max_{10}$	172,76352	362,9538
$\max_{11}$	142,42677	321,43628
$\max_{12}$	170,99661	295,21555
$\max_{13}$	140,03787	364,36176
$\max_{14}$	98,230865	208,93227
$\max_{15}$	68,81298	154,13226
$\max_{16}$	74,12923	160,24773
$\max_{17}$	51,78036	117,74015
$\max_{18}$	20,73961	55,032047
$\max_{19}$	55,946613	99,87239
$\max_{20}$	16,036867	44,92402
$\max_{21}$	23,404562	60,15183
$\max_{22}$	19,944548	46,585274
$\max_{23}$	14,242097	44,026596
$\max_{24}$	11,504292	23,90699
$\max_{25}$	5,2844114	13,773306
$\max_{26}$	7,49625	17,74915
$\max_{27}$	3,6148624	8,03344
$\max_{28}$	4,8835807	14,053778
$\max_{29}$	3,2500324	9,766679
$\max_{30}$	1,6198401	3,8292391

Tabla 16: Parámetros de normalización (IV)

Característica	Media	Desviación Típica
max-freq$_1$	35,377235	14,779791
max-freq$_2$	109,41023	14,310052
max-freq$_3$	157,08893	22,204115
max-freq$_4$	237,34592	25,370153
max-freq$_5$	310,14645	24,312443
max-freq$_6$	403,7519	27,410894
max-freq$_7$	489,3408	23,016937
max-freq$_8$	584,6093	31,939516
max-freq$_9$	672,22675	27,801172
max-freq$_{10}$	781,4551	30,98548
max-freq$_{11}$	904,9106	33,565907
max-freq$_{12}$	1004,27	35,98437
max-freq$_{13}$	1136,5785	49,034184
max-freq$_{14}$	1269,4824	45,24625
max-freq$_{15}$	1409,3927	50,92556
max-freq$_{16}$	1535,1046	41,005337
max-freq$_{17}$	1704,3861	49,30979
max-freq$_{18}$	1873,1862	58,287815
max-freq$_{19}$	2052,2415	60,40939
max-freq$_{20}$	2217,6611	66,80756
max-freq$_{21}$	2417,3076	57,010002
max-freq$_{22}$	2638,4033	58,779163
max-freq$_{23}$	2857,4375	74,33377
max-freq$_{24}$	3086,9392	80,97323
max-freq$_{25}$	3342,3274	80,32275
max-freq$_{26}$	3589,7864	78,69773
max-freq$_{27}$	3879,9844	98,408936
max-freq$_{28}$	4168,601	103,71468
max-freq$_{29}$	4459,1206	91,55003
max-freq$_{30}$	4837,6816	100,40303

5.3.1. Resultados

Para permitir la reproducibilidad de los experimentos, se utilizó *100* como semilla de generación de números aleatorios.

5.3.1.1 RNA

Para el entrenamiento de las RNA, se han utilizado los siguientes parámetros en común:

- Ratio de aprendizaje: 0,01
- Ratio de patrones para el conjunto de validación: 0,2.
- Máximo número de ciclos (*epochs*) de entrenamiento: 1500.
- Máximo número de ciclos (*epochs*) de entrenamiento sin mejorar el error de validación: 5.
- Algoritmo de optimización: ADAM.
- Función de *loss*: *Cross Entropy*.

El hiperparámetro que hemos variado ha sido la arquitectura de neuronas a utilizar. Utilizamos 8 arquitecturas distintas, donde la arquitectura [i] denota una RNA con una única capa oculta con i neuronas y la arquitectura [i, j] denota una neurona con dos capas ocultas; con i neuronas en la primera capa oculta y j neuronas en la segunda capa oculta.

Se muestran los resultados de la experimentación en la Tabla 17

Tabla 17: Resultados RNA

Arquitectura	F1-Score	Precisión
[8]	$0{,}91045 \pm 0{,}01106$	$0{,}93454 \pm 0{,}00795$
[16]	$0{,}98154 \pm 0{,}00805$	$0{,}98529 \pm 0{,}00607$
[32]	$0{,}99329 \pm 0{,}00318$	$0{,}99473 \pm 0{,}00229$
[8, 8]	$0{,}81746 \pm 0{,}00835$	$0{,}86732 \pm 0{,}0055$
[16, 4]	$0{,}51003 \pm 0{,}02221$	$0{,}62847 \pm 0{,}0195$
[16, 8]	$0{,}92262 \pm 0{,}00672$	$0{,}94141 \pm 0{,}00598$
[32, 16]	$0{,}9773 \pm 0{,}00769$	$0{,}9813 \pm 0{,}00694$
[32, 32]	$0{,}98522 \pm 0{,}00663$	$0{,}98798 \pm 0{,}00494$

5.3.1.2 SVM

Para las máquinas de soporte vectorial hemos utilizado 8 configuraciones distintas de los hiperparámetros de *kernel*, grado del *kernel* (sólo se utiliza en *kernel* polinomial), *gamma* y *C*.

Se muestran los resultados de la experimentación en la Tabla 18

Tabla 18: Resultados SVM

(kernel, grado, gamma, C)	F1-Score	Precisión
(poly, 3, 1, 1)	$0{,}99264 \pm 0{,}0053$	$0{,}99382 \pm 0{,}00422$
(rbf, 3, 1, 1)	$0{,}10019 \pm 0{,}01854$	$0{,}11513 \pm 0{,}0161$
(sigmoid, 3, 1, 1)	$0{,}13051 \pm 0{,}02229$	$0{,}20283 \pm 0{,}02546$
(poly, 3, 10, 0,1)	$0{,}99113 \pm 0{,}00724$	$0{,}99314 \pm 0{,}00464$
(rbf, 3, 10, 0,1)	$0{,}00179 \pm 7{,}0e-5$	$0{,}04673 \pm 0{,}00189$
(sigmoid, 3, 10, 0,1)	$0{,}10752 \pm 0{,}01502$	$0{,}18363 \pm 0{,}01528$
(rbf, 3, 0,01, 100)	$0{,}99504 \pm 0{,}00411$	$0{,}99563 \pm 0{,}00257$
(poly, 3, 100, 0,001)	$0{,}99257 \pm 0{,}00408$	$0{,}99379 \pm 0{,}00278$

5.3.1.3 Árboles de decisión

Para los árboles de decisión hemos utilizado 6 configuraciones distintas del hiperparámetro de profundidad máxima.

Se muestran los resultados de la experimentación en la Tabla 19

Tabla 19: Resultados Árbol de decisión

Altura máxima	F1-Score	Precisión
4	$0{,}09189 \pm 0{,}01043$	$0{,}17819 \pm 0{,}0157$
8	$0{,}3111 \pm 0{,}01697$	$0{,}42453 \pm 0{,}01404$
16	$0{,}71777 \pm 0{,}04105$	$0{,}79892 \pm 0{,}03264$
32	$0{,}94074 \pm 0{,}01447$	$0{,}94584 \pm 0{,}01159$
64	$0{,}94887 \pm 0{,}01332$	$0{,}95193 \pm 0{,}01202$
128	$0{,}95088 \pm 0{,}00891$	$0{,}9555 \pm 0{,}00573$
256	$0{,}95003 \pm 0{,}00775$	$0{,}95269 \pm 0{,}0076$

5.3.1.4 kNN

Para el modelo kNN hemos utilizado 6 configuraciones distintas del hiperparámetro de k (número de vecinos más cercanos).

Se muestran los resultados de la experimentación en la tabla 20

Tabla 20: Resultados kNN

K	F1-Score	Precisión
2	$0{,}98683 \pm 0{,}00434$	$0{,}98845 \pm 0{,}00303$
3	$0{,}98153 \pm 0{,}00991$	$0{,}98636 \pm 0{,}00479$
5	$0{,}98086 \pm 0{,}00835$	$0{,}98617 \pm 0{,}00599$
8	$0{,}97161 \pm 0{,}01603$	$0{,}98024 \pm 0{,}01049$
13	$0{,}96203 \pm 0{,}01287$	$0{,}97342 \pm 0{,}00864$
21	$0{,}95836 \pm 0{,}00902$	$0{,}96928 \pm 0{,}00517$

5.3.2. Discusión

Los resultados de esta iteración son muy buenos, ya que hemos aumentado en gran medida el número de notas a clasificar (50) y hemos conseguido varios modelos con más de un 99 % de *F1-Score* y precisión.

En concreto,el modelo que mejor funciona es un SVM con un *kernel* de base radial y con valores de *gamma* y C de 0,01 y 100, respectivamente. Este modelo alcanzó un *F1-Score* de $0{,}99594 \pm 0{,}00411$ y una precisión de $0{,}99563 \pm 0{,}00257$, unos valores muy altos y que indican que el sistema funciona muy bien.

Caben destacar también modelos que alcanzan un rendimiento parecido, como puede ser un SVM con un kernel polinomial de grado 3, con ambos valores de *gamma* y C de 1, además de una RNA con una única capa oculta con 32 neuronas.

En contra, el modelo que peor rendimiento presenta es un SVM con un *kernel* de base radial, con valores de *gamma* y C de 10 y 0,1, respectivamente. Cabe destacar que este modelo es similar al modelo que presenta el mejor rendimiento, siendo lo único que cambia los parámetros de *gamma* y C. Es notable ver que un ligero cambio en estos dos parámetros, puede producir que un sistema pase de ser muy malo, a ser tremendamente bueno.

Como hemos podido ver, los resultados son muy buenos en general, pero estos sistemas tienen un problema (principalmente las RRNNAA), y es que el tiempo de entrenamiento es mucho mayor que en aproximaciones anteriores. Creemos que esto se puede deber tanto por aumentar el número de instancias (anteriormente 1090, y en esta aproximación, 3194) a utilizar por el sistema, como por el número de características que se utilizan (122). Como hemos obtenido resultados muy buenos, estaría bien que en una próxima aproximación se intente reducir el número de características, intentando no perder el rendimiento de clasificación del sistema, y así poder producir modelos que sean más rápidos de entrenar, a la vez que más rápidos para realizar predicciones ante futuros patrones.

6. Conclusiones

7. Trabajo futuro

8. Bibliografía

Referencias

Baevski, A., Hsu, W.-N., Conneau, A., & Auli, M. (2021). Unsupervised speech recognition. *Advances in Neural Information Processing Systems*, *34*.

Benetos, E., Dixon, S., Duan, Z., & Ewert, S. (2018). Automatic music transcription: An overview. *IEEE Signal Processing Magazine*, *36*(1), 20-30.

Chang, S., Lee, J., Choe, S. K., & Lee, K. (2017). Audio cover song identification using convolutional neural network. *arXiv preprint arXiv:1712.00166*.

Foo, S. W., & Wong, P. L. (1999). Recognition of piano notes. *IEEE International Conference on Information, Communications and Signal Processing (1999: Singapore)*.

Klapuri, A. (2004). *Signal processing methods for the automatic transcription of music*. Tampere University of Technology Finland.

Marolt, M., Kavcic, A., & Privosnik, M. (2002). Neural networks for note onset detection in piano music. *Proceedings of the 2002 International Computer Music Conference*.

Osmalsky, J., Embrechts, J.-J., Van Droogenbroeck, M., & Pierard, S. (2012). Neural networks for musical chords recognition. *Journées d'informatique musicale*, 39-46.

Solanki, A., & Pandey, S. (2019). Music instrument recognition using deep convolutional neural networks. *International Journal of Information Technology*, 1-10.