ISSN 1870-4069

Comparación de clasificadores para el reconocimiento de notas musicales

Omar Velázquez López, José Luis Oropeza Rodríguez, Sergio Suárez Guerra

Instituto Politécnico Nacional, Centro de Investigación en Computación, Ciudad de México, México

{joropeza, ssuarez}@cic.ipn.mx

Abstract. Este artículo describe el conjunto de experimentos realizados para obtener el reconocimiento de 60 notas musicales de un piano digital por medio de técnicas de procesamiento digital de señales y clasificadores. Para la etapa de técnicas de procesamiento digital de señales se utilizaron: Frecuencia fundamental, coeficientes Cepstrales en la Frecuencia de Mel y Cepstrales de Mecánica Coclear. Para los clasificadores se consideraron Modelos Ocultos de Markov usando HTK (Hidden Markov Model Toolkit), Modelos de Mixturas Gaussianas, Redes Neuronales Artificiales y Cuantificación Vectorial. El corpus creado para este trabajo consistió de 720 notas comprendidas en un rango de cinco octavas. Se usaron 12 archivos para cada una de las 60 notas a reconocer, las cuales se grabaron a una velocidad 70 bpm (beats per minute). Los mejores resultados se obtuvieron al aplicar coeficientes CMCC como técnica y Modelos Ocultos de Markov como clasificador (100% de reconocimiento en cada octava). Igualmente se obtuvo el 100% de reconocimiento por octava al usarse coeficientes MFCC como técnica y Cuantificación Vectorial como clasificador.

Palabras clave: Transcripción automática de música, procesamiento digital de señales, reconocimiento de patrones.

Comparison of Classifiers for Recognition of Musical Notes

Abstract. This paper describes the set of experiments performed for recognition of 60 musical notes of a digital piano using digital signal processing techniques and classifiers. In the stage of digital signal processing techniques we used: Fundamental frequency (Pitch), Mel-Frequency Cepstral Coefficients (MFCC) and Cochlear Mechanics Cepstral Coefficients (CMCC). The implemented classifiers were: Hidden Markov Models (HMM) using HTK (Hidden Markov Model Toolkit), Gaussian Mixture Models (GMM), Neuronal Networks and Vector Quantization (VQ). The created corpus for this work consists of 720 notes included in a range of five octaves. We used 12 files for each one of the 60 notes for recognition, which were recorded in a speed of 70 beats per minute. The best results were obtained applying CMCC as technic and HMM as classifier (100%

of recognition in each octave). Also we obtained the 100% of recognition for each octave using MFCC as technique and VQ as classifier.

Keywords: Automatic transcription of music, signal digital processing, pattern recognition.

1. Introducción

El reconocimiento de notas musicales forma parte del problema de realizar transcripción automática de música, el cual es el proceso de convertir una señal de audio a la notación musical convencional mediante métodos computacionales y electrónicos. Para dicha tarea, es necesario realizar la extracción de características de la señal de audio utilizando técnicas de procesamiento digital de señales. La información contenida en una señal de música es de múltiples dimensiones, comúnmente está descrita por su tono, duración y timbre.

El tono de una señal de música en términos de análisis de señales, se conoce como frecuencia fundamental. Existe la posibilidad de que dos instrumentos estén ejecutando la misma nota y por lo tanto el mismo tono, pero lo que la diferencia entre uno y el otro es su timbre.

2. Características y generalidades

Existen tres maneras posibles de resolver el problema de transcripción automática de música desde el punto de vista como plataforma: software (no en tiempo real), hardware (en tiempo real) o híbridamente. En cuanto a software, en [3] se describen las diferentes técnicas de procesamiento digital de señales clásicas y avanzadas; y los tipos de clasificadores utilizados para la solución de este problema. Por otro lado, las soluciones propuestas en el estado del arte para resolver el problema de transcripción en tiempo real mediante hardware han demostrado resultados desde diferentes enfoques, tal como se menciona a continuación.

[4] consiste en un algoritmo de búsqueda del tono fundamental de una señal de audio mediante un microcontrolador. Por otra parte [1] describe un sistema de transcripción de música en tiempo real para guitarras acústicas, éste consiste en tres componentes principales de hardware que se colocan en la guitarra y en una mano del intérprete: estampas sensores adheridas al diapasón, un recolector con pastillas en la boca de la guitarra y un anillo colocado en el dedo de la mano derecha del intérprete.

Tabla 1. Porcentaje de notas transcritas correctamente en [4].

Duración de Notas Ejecutadas	**Tamaño de Ventana Estática**		**Tamaño de Ventana Dinámica**
	20 ms	***100 ms***	
Corta duración	67%	50%	80%
Larga duración	70%	75%	92%

En el caso de [4] se hace una clasificación de las notas entre corta y larga duración. No se específica cuales corresponden a una clase, pero se puede suponer que se trata de notas corcheas para el primero, y notas negras para el segundo. Tampoco se hace referencia al alcance de notas reconocidas en cuanto a frecuencia o tono. Por otro lado en [1] al tratarse de un sistema que consta de sensores en cada traste de una guitarra acústica puede suponerse que el alcance de notas comprende todas aquellas que están contenidas en el diapasón de la guitarra. Sin embargo, no se menciona un porcentaje de notas reconocidas por el sistema.

3. Un sistema de transcripción automática de música

Es posible resolver el problema de transcripción automática de música utilizando un transcriptor, el cual consta de dos partes: *front end* y *back end*. La primera contiene los algoritmos encargados de la extracción de características de la señal de música y el segundo contiene los clasificadores que reconocen las notas interpretadas. Como primer paso es necesario entrenar el *back end* por medio de los datos procesados en el *front end*, para ello se lleva a cabo el proceso descrito por el diagrama de bloques de la Fig. 1. Una vez entrenado el clasificador, se procede a probar el transcriptor y reconocer cada nota, siguiendo el procedimiento descrito por la Fig. 2.

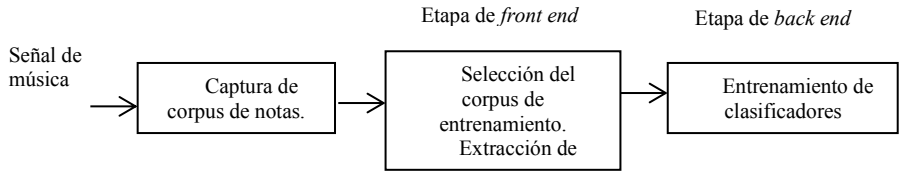

Fig. 1. Diagrama de bloques de la etapa de entrenamiento.

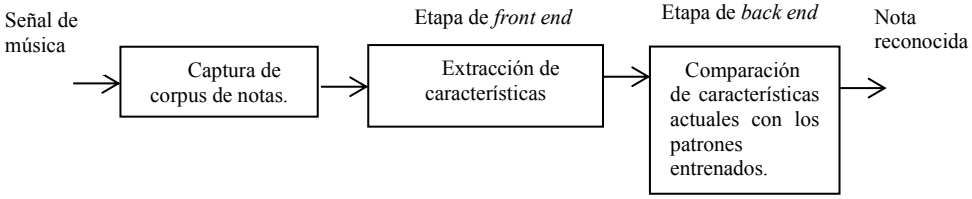

Fig. 2. Diagrama de bloques de la etapa de reconocimiento.

4. Características de la señal de música

Para la implementación del *front end* se utilizaron las siguientes características: Tono fundamental (Pitch), los coeficientes MFCC (cepstrales en la escala de mel) y CMCC (cepstrales de mecánica coclear). A continuación cada una de ellas se describe de forma breve:

4.1. Tono fundamental (*Pitch*)

El tono fundamental (*Pitch* en inglés) es una característica de la señal de música que depende de la frecuencia fundamental (F0) de la misma. Hay diferentes maneras de calcular el F0, para este trabajo se hizo uso de la técnica de autocorrelación parcial.

4.2. Coeficientes cepstrales en la frecuencia de Mel (MFCC)

Los Coeficientes Cepstrales en las Frecuencias de Mel (MFCC en inglés) son otro tipo de características de la señal de música. Trabajan de acuerdo al modelo del sistema auditivo de las personas, es decir, la forma que un ser humano escucha.

El oído humano no percibe las frecuencias de los sonidos linealmente. La manera en que este trabaja demuestra que su escala es aproximadamente lineal hasta 1 kHz y después de ello se comporta de manera logarítmica, y es la Escala de Mel la que caracteriza esa percepción de frecuencia logarítmica por medio de bancos de filtros.

Si se desea obtener más información de la señal de audio además de los MFCC, se pueden calcular más coeficientes tales como los MFCC-Delta (o ΔMFCC) y los MFCC-Delta-Delta (o ΔΔMFCC) [7].

4.3. Coeficientes cepstrales de mecánica coclear (CMCC)

Los coeficientes cepstrales de mecánica coclear (CMCC en inglés) se pueden obtener a partir del funcionamiento de la cóclea [2]. Se utiliza un modelo basado en mecánica de fluidos y su solución se obtiene bajo un análisis de resonancia en el dominio de la frecuencia.

En este tipo de coeficientes la distribución de banco de filtros a los MFCC, es una distribución que depende de la respuesta interna del oído al estímulo que recibe.

5. Clasificadores

Para la parte *back end* se implementaron los siguientes clasificadores: Modelos Ocultos de Markov usando HTK (Hidden Markov Model Toolkit), Modelos de Mixturas Gaussianas, Redes Neuronales y Cuantificación Vectorial. Cada uno de ellos se describe a continuación:

5.1. Modelos ocultos de Markov (HMM)

Un modelo oculto de Markov (HMM en inglés) es un modelo probabilístico aplicado sobre un conjunto de variables aleatorias $(O, Q) = \{O_1, \dots, O_T, Q_1, \dots, Q_T\}$. Las observaciones pueden ser continuas o discretas.

Un HMM resuelve tres problemas básicos:

1. Calcular eficientemente $P(O|\lambda)$, la probabilidad de la secuencia de observaciones $O = \{O_1, O_2, \dots, O_T\}$, dado el modelo $\lambda(A, B, \pi)$.
2. Hallar una secuencia óptima de estados $Q = \{q_1, q_2, \dots, q_T\}$, dada una secuencia de observaciones $O = \{O_1, O_2, \dots, O_T\}$ y un modelo λ.

3. Ajustar los parámetros del modelo $\lambda(A, B, \pi)$ para maximizar $P(O|\lambda)$.

Los tres problemas mencionados anteriormente se aplican para el reconocimiento automático de voz [6]. De manera similar se pueden aplicar en la tarea de reconocer y modelar el problema la transcripción automática de música.

5.2. Modelos de mixturas gaussianas (GMM)

Un modelo de mixtura gaussiana (GMM en inglés) es un modelo probabilístico en el que se hace la suposición de que todos los datos son generados de una mixtura de un número finito de distribuciones gaussianas con parámetros desconocidos, la forma de una mixtura Gaussiana se describe en [5] por:

$$g(\mu, \Sigma)(x) = \frac{1}{\sqrt{2\pi^d}\sqrt{\det(\Sigma)}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}, \quad (1)$$

en dónde las letras μ y Σ representan la media y la covarianza de cada componente respectivamente. Un conjunto de mixturas gaussianas se describe como:

$$gm(x) = \sum_{k=1}^{K} w_k * g\left(\mu_k, \Sigma_k\right)(x). \quad (2)$$

5.3. Redes neuronales (ANN en inglés)

Una red neuronal artificial es un tipo de clasificador perteneciente al área de Inteligencia Artificial. Su modelo puede constar de diferentes elementos dependiendo de su tipo, entre los esenciales se tienen: entradas y salidas, neuronas en sus capas ocultas, pesos y funciones de transferencia.

Las redes tipo feedforward cuentan con capas de neuronas con funciones transferencia no lineales que permiten que la red pueda aprender relaciones lineales y no lineales entre los vectores de entrada y de salida. Si se desea trabajar con valores de salida entre 0 y 1, se debe utilizar una función de transferencia tipo log-sigmoidal. En el presente trabajo se utilizaron las funciones tan-sigmoidal y log-sigmoidal.

5.4. Cuantificación vectorial (VQ)

La cuantificación vectorial (VQ en inglés) es una técnica en la cual un conjunto de vectores de entrada es dividido en un número n de regiones, siendo un vector aquel que define dicha región. El algoritmo que realiza esta tarea se conoce como LBG algoritmo propuesto por [8] este agrupa un conjunto de vectores de entrenamiento L en un conjunto de vectores del libro de códigos M.

6. Experimentos

Para llevar a cabo la evaluación de cada técnica y clasificador, como primer paso fue necesario crear el corpus de las notas a reconocer. Para ello se hizo uso de un piano

digital con un rango de notas comprendido entre C2 y B6, es decir, un total de 60 notas como se muestra en la Fig. 3; cada una de ellas se ejecutó un total de 12 veces a un tempo de 70 bpm para grabarse en un archivo. Posteriormente se recortó cada archivo para separar cada muestra y así tener un total de 720 archivos de muestras.

Cada clasificador se implementó por octavas, y en cada una de estas, hay un contenido total de 12 notas. Debido a que se tienen 12 archivos muestras por cada nota, se tiene un total de 144 archivos muestras por octava.

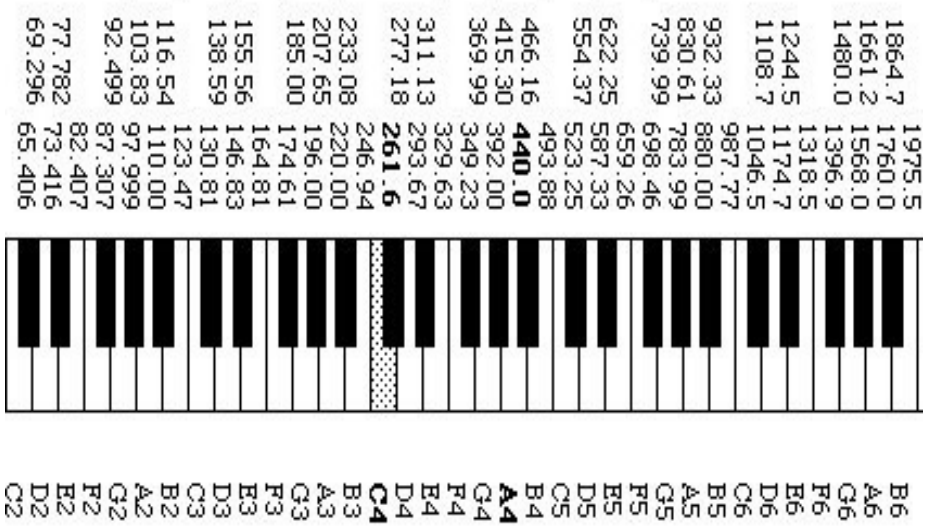

Fig. 3. Rango de notas a reconocer.

6.1. Implementación de HMM con HTK

HTK (Hidden Markov Model Toolkit) es una herramienta utilizada para el reconocimiento de voz. En esta ocasión se ha aplicado para resolver el problema de reconocimiento de notas musicales.

Se crearon modelos para dos tipos de características: MFCC y CMCC. Se emplearon vectores característicos de 39 coeficientes para cada caso. Los primeros 13 coeficientes del vector corresponden a los 12 coeficientes MFCC más el valor de la energía; los segundos 13 son los coeficientes de velocidad (ΔMFCC); y los últimos 13 son los coeficientes de aceleración (ΔΔMFCC). Por lo que cada vector característico v se conformó de la siguiente manera:

$$v = (c_1, c_2, \cdots, c_{12}, E, \ \Delta c_1, \Delta c_2, \cdots, \Delta c_{13}, \ \Delta\Delta c_1, \Delta\Delta c_2, \cdots, \Delta\Delta c_{13}). \quad (3)$$

Se hizo un Modelo Oculto de Markov por cada nota y el experimento se realizó para cada una de las cinco octavas. Cada modelo está compuesto por 5 estados y 1 Mixtura Gaussiana por estado. Se dividieron todos los archivos muestras para hacer el entrenamiento con una mitad de ellos y el reconocimiento con la otra. El procedimiento se describe a continuación.

Con la función HParse se crea una red a partir de la gramática propuesta. Esta función recibe un archivo (gramatica.txt) con una descripción sintáctica que sigue un conjunto de reglas, el cual genera la red. El resultado se guarda en un archivo .slf.

Posteriormente se hace el cálculo de coeficientes MFCC y CMCC mediante la función HCopy, la cual copia los archivos de datos a un archivo de salida. Convirtiendo los datos en una forma parametrizada. Esta función originalmente funciona con coeficientes MFCC, por lo cual fue necesario modificarla para calcular coeficientes CMCC.

Con la función HInit es posible realizar las estimaciones iniciales para los parámetros del modelo. Posteriormente la función HRest permite llevar a cabo la reestimación básica Baum-Welch de cada modelo.

Con la función HVite se ejecuta un reconocedor Viterbi de propósito general. Este relaciona un archivo de voz contra una red de modelos HMM y da como salida una transcripción para cada uno. Los resultados se guardan en un archivo .mlf.

Finalmente se aplica la función HResults. Ésta lee un conjunto de archivos de etiqueta y compara con los archivos de transcripción correspondientes. Para el análisis de salida de reconocimiento de voz, la comparación está basada en un procedimiento de alineamiento de cadenas basado en programación dinámica.

6.2. Implementación de GMM

En este clasificador también se utilizó el mismo corpus de datos, una mitad para entrenamiento y la otra para reconocimiento. Para llevar a cabo la implementación de este clasificador se hizo uso de la herramienta VOICEBOX creada por Mike Brookes en MATLAB. En ella están contenidas funciones que fueron empleadas para crear cada modelo de mixtura Gaussiana. El experimento se hizo para 2, 3 y 4 mixturas gaussianas por modelo. Dando como mejores resultados los modelos de 3 mixturas, los cuales se muestran en la Tabla 2.

Se emplearon vectores característicos de 13 coeficientes; compuestos por 12 coeficientes MFCC más el valor del F0, el cual fue calculado por un algoritmo de detección del tono fundamental en cada segmento. Por lo que cada vector característico v se conformó de la siguiente manera:

$$v = (c_1,\ c_2,\ c_3, \cdots, c_{12},\ F0). \tag{4}$$

Este procedimiento se realizó para la mitad destinada al entrenamiento por cada octava de análisis. Una vez entrenados los modelos, se probaron con la mitad restante de los datos. Se hizo un programa capaz de construir una tabla de confusión con la finalidad de una comparación final de resultados.

6.3. Implementación de ANN

Para implementar este clasificador se hizo uso de las funciones mencionadas anteriormente en la implementación de GMM. Se empleó la herramienta Neuronal Network Toolbox de MATLAB. En ella están contenidas funciones que fueron empleadas para crear cada modelo de redes neuronales. El experimento se hizo para 20, 30 y 40 neuronas en la capa oculta. Dando como mejores resultados los modelos de 20 neuronas, los cuales se muestran en la Tabla 2.

6.4. Implementación de VQ

Para implementar este clasificador se hizo uso de las funciones mencionadas anteriormente para GMM. Se creó un libro código por cada nota para cada octava. El experimento se realizó con 16 y 32 vectores de libro código. Dando como mejores resultados el libro código de 32 vectores, los cuales se muestran en las Tabla 2.

Tabla 2. Resultados por modelo utilizado.

Octava 2							
Clasificador		%Corr	H	D	S	I	N
HMM (MFCC)	[39]	77.78	56	0	16	0	72
HMM (CMCC)	[39]	**100**	72	0	0	0	72
GMM	[13]	**100**	72	0	0	0	72
ANN	[13]	98.61	71	0	1	0	72
VQ	[13]	**100**	72	0	0	0	72
Octava 3							
Clasificador		%Corr	H	D	S	I	N
HMM (MFCC)	[39]	90.28	65	0	7	0	72
HMM (CMCC)	[39]	**100**	72	0	0	0	72
GMM	[13]	**100**	72	0	0	0	72
ANN	[13]	91.66	66	0	6	0	72
VQ	[13]	**100**	72	0	0	0	72
Octava 4							
Clasificador		%Corr	H	D	S	I	N
HMM (MFCC)	[39]	98.61	71	0	1	0	72
HMM (CMCC)	[39]	**100**	72	0	0	0	72
GMM	[13]	97.22	70	0	2	0	72
ANN	[13]	98.61	71	0	1	0	72
VQ	[13]	**100**	72	0	0	0	72
Octava 5							
Clasificador		%Corr	H	D	S	I	N
HMM (MFCC)	[39]	98.61	71	0	1	0	72
HMM (CMCC)	[39]	**100**	72	0	0	0	72
GMM	[13]	**100**	72	0	0	0	72
ANN	[13]	98.61	71	0	1	0	72
VQ	[13]	**100**	72	0	0	0	72
Octava 6							
Clasificador		%Corr	H	D	S	I	N
HMM (MFCC)	[39]	97.22	70	0	2	0	72
HMM (CMCC)	[39]	**100**	72	0	0	0	72
GMM	[13]	75	54	0	18	0	72
ANN	[13]	83.33	60	0	12	0	72
VQ	[13]	**100**	72	0	0	0	72

7. Resultados

En la Tabla 2 se muestran los resultados obtenidos por cada clasificador. Como se mencionó anteriormente, se tiene un total de 144 archivos muestras por octava. Dado que la mitad de ellos fue utilizada para el entrenamiento y la otra para el reconocimiento, éste último se aplica sobre 72 archivos, N. El indicador *%Corr* es el porcentaje de notas reconocidas correctamente. *H* es el número de notas correctas en la octava de análisis. *D* es el número de eliminaciones. *S* es el número de sustituciones, cuando una nota se reconoce como otra distinta existe una sustitución. Y finalmente *I* es el número de inserciones. Los valores entre corchetes a lado del nombre de cada clasificador, indican el número de coeficientes utilizado en cada vector de entrenamiento.

8. Conclusiones y trabajos futuros

Después de haber realizado los experimentos anteriores, se ha demostrado que los clasificadores con mejores resultados para realizar una transcripción automática de música han sido los HMM con CMCC y la Cuantificación Vectorial con MFCC como características. Sin embargo, es importante mencionar que cuando se han aplicado los HMM estos no contemplan como necesario el cálculo del F0, lo cual representa una gran ventaja ya que se ahorra la implementación y ejecución del algoritmo para su detección.

Cabe decir que a pesar de no ser un clasificador tan complejo como el resto, el cuantificador vectorial ha alcanzado excelentes resultados en este trabajo. Esto probablemente debe a que la tarea de reconocimiento se llevó a cabo únicamente para notas monofónicas y aisladas con una misma duración.

Dados los resultados mostrados, se puede considerar llevar a cabo la implementación de los HMM con CMCC y la Cuantificación Vectorial con MFCC como características sobre una plataforma, para realizar la transcripción automática de música en tiempo real.

Referencias

1. Han-Jong, K., Tek-Jin, N.: Muzlog: Instant Music Transcribing System for Acoustic Guitarists. CIDR Lab, Department of Industrial Design (2014)
2. Oropeza, J., L.: New Parameters for Automatic Speech Recognition Based on the Mammalian Cochlea Model Using Resonance Analysis (2013)
3. Benetos, E.: Automatic Transcription of Polyphonic Music Exploiting Temporal Evolution. PhD thesis, School of Electronic Engineering and Computer Science Queen Mary University of London (2012)
4. Farshad, A., Doraisamy, S.: A Real-Time Note Transcription Technique Using Static and Dynamic Window Sizes. In: ICCSE, Malaysia (2009)
5. Bilmes, J. A.: A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models. International Computer Science Institute, Berkeley, CA. (1998)

6. Lawrence, Rabiner, L., Biing-Hwang J.: Fundamentals of Speech Recognition. Prentice Hall (1993)
7. Furui, S.: Comparison of speaker recognition methods using statistical features and dynamic feature (1981)
8. Linde, Y., Buzo, A., Gray, R., M.: An algorithm for Vector Quantizer Design (1980)