Proceso de creación del software de extracción de características con jAudio

Brayan Mauricio Rodríguez Rivera

Escuela de Ingeniería de Sistemas y Computación,
Universidad del Valle
Cali, Colombia
brayan.rodriguez.rivera@correounivalle.edu.co

***Abstract*— Este documento hace una detallada explicación del proceso por el que se pasa al momento de implementar un software de extracción de características acústicas, con la ayuda de la librería *jAudio*, y de metadatos de cada canción, esto apoyándose en la librería Tika Apache. Además de la documentación de este proceso, se hacen las pruebas necesarias para su validación o desacreditación de los datos que brinda el software.**

***Palabras clave*— *jAudio*, software, características, librería, aplicativo, jMIR, canción, método, extracción.**

I. Introducción

Con este documento se pretende informar y documentar acerca del proceso por el que se pasa al momento de implementar un software de extracción de características, usando *jAudio*, como componente de *jMIR* que tiene como objetivo aportar a la investigación de la recuperación de información musical a través de *softwares* de extracción de características de grabaciones de audio, entre otros. Además de esta extracción extracción de características que nos ofrece *jAudio*, se pretende extraer metadatos, conocidos como *tags*, de cada canción.

A través de esta documentación, el lector que tenga como fin implementar un software semejante o idéntico al aquí expuesto, puedan omitir pasos infructuosos o saber de antemano los resultados de algunas modificaciones o soluciones planteadas en este proceso y, evidentemente, de esta manera se pueda ahorra tiempo y recursos.

Previo a lo aquí descrito, se había manejado el *jAudio* como aplicación con interfaz gráfica para tener la experiencia de extraer características acústicas de una canción y así saber el tipo de dato que se genera, el tipo de archivo que maneja la librería y demás características de ésta.

El escrito está delimitado a informar exclusivamente el proceso a partir de la descarga de la librería para su uso en el aplicativo a crear. En el texto se encuentra tanto errores que se cometieron en la búsqueda de el propósito general, como aciertos que ayudaron fomentaron de forma sustancial la misma.

II. Creación del software

Para llevar a cabo la creación de este software, se descarga la librería de jAudio de la página oficial. Adjuntando esta librería a un aplicativo que busca en el interior de cada subcarpeta archivos con la extensión .mp3 y pasa cada archivo a la librería dicha anteriormente a través del método extract de la clase jAudioFeatureExtractor.DataModel, se crea un software que en teoría funcionaría a la perfección, mas cuando se llevó a la práctica no fue así ya que se detectan algunos errores al interior de la librería.

Para seguir con el propósito de la creación del software se descarga el código fuente de la librería para verificar los errores que existen.

Se modifica la línea 98 y la línea 100 del archivo jAudioFeatureExtractor.DataModel para agregar el *path* al que yo quiero que sean escritos las *keys* (definiciones) y los *values* (valores) de las características a extraer. En este orden, se edita el constructor de DataModel para agregar como parámetros la dirección del *feature key* y los *feature values*, se eliminan algunas clases que generaban errores con esta modificación.

Posterior a esto, se presenta un problema con el método extract de la clase DataModel porque el parámetro que recibe es un arreglo de tipo RecordInfo y para definir un arreglo en la clase principal, donde se usa, se necesita saber el número de canciones a extraer características, por lo que se acomoda el código y el método *extract* para que funcione con un vector en lugar de un arreglo. Debido a esta modificación, aparece error de *Stream Closed* en la línea *writeBytes(feature_value_header)*;, se soluciona el error pasando la llamada desde la clase principal al método extract() de DataModel a otro método para que solamente se llame una sola vez, y no una vez por subcarpeta como estaba pasando.

En esta instancia, el programa funciona, extrae las características, sin embargo, los MFCC no, el software coloca un '?' en el .arff de salida.

Luego de investigar, se soluciona el error descargando otro archivo features.xml (https://github.com/DDMAL/jMIR/blob/master/jAudio/features.xml), de internet, que utiliza el DataModel para extraer las características. Sin embargo, éste produce que el software deje de extraer las siguientes características: *Spectral Flux0, Fraction Of Low Energy Windows0, Strongest Beat0, Beat Sum0, Strength Of Strongest Beat0.*

Se solicita la conversión de los archivos mp3 a wav para poder que a estos también se le haga la extracción de características. Para este fin, se prueba la conversión con varias librerías (LameOnj, mp3transform), se implementa la opción de cambiar solamente la extensión pero no funcionó ninguna de estas alternativas. Después de un tiempo de averiguaciones, se encuentra la librería *JLayer*, se prueba y funciona. Luego se agrega a la clase reconocedor del proyecto. En el propósito de la extracción de etiquetas de los archivos mp3, se investiga en Internet acerca de la librería de *jAudio* para evaluar si en su contenido tiene la funcionalidad de extraer los metadatos de los archivos mp3. Debido a que en esta investigación no se encuentra tal funcionalidad, se busca en general, en Java, las librerías que permiten disponer de esta funcionalidad.

En esta búsqueda se encuentra una librería llamada Tika Apache de The Apache Software Foundation (http://www.apache.org/dyn/closer.cgi/tika/tika-app-1.7.jar) donde me ofrecen los metadatos de la canción mp3. Esta librería me ofrece el título, los artistas, el compositor, el género y el álbum, cada uno de estos, claramente, si los tiene.

Al encontrar esta librería, se procedió a escribir el código para la extracción de metadatos de una canción. Verificada la funcionalidad de ésta, se une el código con la librería de *jAudio* que se venía manejando anteriormente en la clase FeatureProcessor se hacen unas modificaciones con el fin de agregar el método extractMP3Tags() se comenta las lineas agregadas con la aclaración "//added by Brayan".

III. Pruebas y análisis de resultados

En esta fase se hace una comparación entre los valores arrojados por el software implementado en el proceso ya explicado y un aplicativo extractor de características, *jAudio*, que se descarga desde la página oficial de jMIR. El primero imprime un solo valor de cada característica por canción, mientras que el extractor de jMIR extrae un promedio del valor que tiene cada característica por cada ventana (determinada por el usuario) por aparte de una misma canción con su respectiva desviación estándar. Para efectuar las pruebas se hace una comparación del software implementado con el descargado evaluándose si el número arrojado por el primero está dentro del rango del segundo, valorando la veracidad del valor de cada característica arrojado.

Previo a esto, se copia el archivo features.xml que se utiliza en la ejecución del software implementado, como se explicó arriba, y se usa también para la ejecución del aplicativo de jMIR. Cabe aclarar que se usa el mismo conjunto de 20 canciones al momento de ejecutar ambos aplicativos.

A continuación se muestran los resultados de las pruebas que se llevaron a cabo mostrando la cercanía del valor del software implementado con el rango que arroja el descargado por cada característica.

En las características *Spectral Centroid* y *Zero Crossings* el valor extraído por el software implementado está por arriba del rango de aceptación que brinda el aplicativo de jMIR para cada una; mientras que en el *Spectral Rolloff Point* y el *Compactness*, pasa totalmente lo contrario, están por debajo de los intervalos que brinda el aplicativo respectivamente. Por su parte *Spectral Variability* y *Root Mean Square*, estuvieron mucho mejor que las anteriores, ya que la mayoría de canciones quedaron en el rango, en estas características y unas poquitas quedaron fuera de éste. En cuanto a los MFCCs pasa algo particular y es que el software implementado extrae siempre el mismo número para cada MFCC con todas las grabaciones, es decir, extrae siempre -115 para el MFCC1 de todas las canciones, -8.527E-14 para el MFCC2 de todas las canciones y así sucesivamente un número determinado para cada uno de los 13 MFCC que hacen parte de las características acústicas que el software extrae. Los LPCs son en su mayoría cercanos a 0, por la derecha o por la izquierda. En particular, el LPC0 y el LPC1 quedan por encima y por debajo del rango arrojado por el aplicativo de jMIR respectivamente. De ahí en adelante, hasta el LPC8, en la mayoría de canciones de cada LPC el número arrojado por el software implementado pertenece al rango dado por el aplicativo; en el resto de canciones, este número queda muy cerquita del rango. Con el LPC9 hay una singularidad, ya que el software siempre brinda el 0 para estar característica e igualmente el aplicativo de jMIR su rango es [0,0]. Por último, en cuanto a las características Method of Moments 0 y 4, la mayoría de canciones tienen estas características pertenecientes al rango, mientras que Method of Moments 1 y 2 quedan por debajo y Method of Moments 3, por encima.

Así, la particularidad que se presenta con los MFCCs en lo referente al software implementado será tema de futuros trabajos para investigar el concepto de MFCC como característica acústica y el algoritmo que aplica la librería *jAudio* para extraerla, además del aplicativo de jMIR. Con LPC9, la duda es casi nula, ya que, a diferencia de los MFCCs, el aplicativo de jMIR y el software implementado coinciden en el dato. Respecto a las demás características es notable que en su mayoría pertenecen al rango o quedan próximos a éste que nos brinda el aplicativo dicho, lo que es un buen indicativo del software implementado tomando como punto de comparación los datos ofrecidos por el aplicativo completo, descargado de la página oficial de jMIR. En cuanto a las que quedan lejos, también habría un trabajo investigativo alrededor de esto, aunque deja un poco menos lugar a dudas que los MFCCs. En términos generales, considerando las pruebas anteriormente expuestas, el software extrae la mayoría de características acústicas propuestas de una forma cercana a lo que lo hace el aplicativo oficial de jMIR que es un buen punto de comparación.

IV. Conclusiones

Este trabajo, a pesar de los muchos problemas, algunos de ellos descritos aquí otros no, se pudo lograr en el tiempo dispuesto y con los requerimientos, al principio del proyecto, propuestos.

A manera de conclusiones, primeramente, en lo que refiere a la creación del software, se tuvieron muchos problemas debido a la inexperiencia y de falta de manejo de muchos conceptos que se dan en esta área de la computación. Además la impericia en la librería *jAudio* suma a esta gran cantidad de problemas que se tuvieron por este tema.

En este misma sección, los problemas que se presentaron totalmente imprevistos fueron en cuanto a la inestabilidad de la librería, ya que el método de una clase interna de la librería no funcionaba, a lo que, imprevisiblemente, tocó revisar el funcionamiento de las clases, procesos y funcionalidades por dentro de la librería. Por esto, se recomienda revisar la compatibilidad de cada versión de las librerías y el grado de compactación que tengan éstas en cuanto a la funcionalidad de sus clases y métodos.

En lo que respecta a las pruebas, lo complicado fue evaluar número por número en cada canción de muestra, así que se recomienda usar una hoja de cálculo para aplicar las fórmulas y así de forma más organizada poder hacer las comparaciones más rápido.

En general, fue un proceso muy bueno donde se conoció el área de la investigación musical a grandes rasgos, a través de las librerías y de la extracción de características acústicas. Proceso en el que se aprendió más que programación, la aplicación de ésta a aplicativos con fines de investigación musical.

REFERENCIAS

[1] (2015) Wordreference. [Online]. Available: http://www.wordreference.com/

[2] (2015) jMIR. [Online]. Available: http://jmir.sourceforge.net/

[3] (2015) jAudio. [Online]. Available: http://jaudio.sourceforge.net/

[4] (2015) jAudio download | SourceForge.net. [Online]. Available: http://sourceforge.net/projects/jaudio/

[5] (2015) Project6: Music Information Retrieval – Genre Classification. Avalible: http://www.cs.cmu.edu/~music/cmsip/projects/p6.pdf

[6] (2015) The IEEE website. [Online]. Available: http://www.ieee.org/