Generación de música estocástica usando Modelos ocultos de Markov

Camilo Andres Contreras Cristancho
Danier Elian Gonzalez Ordoñez
Jhon Sebastian Rojas Rodriguez
Juan Sebastian Nieto Giraldo
Yerson Andres Valderrama Ceron

Problemática:

Hoy en día los sistemas utilizados para identificar y administrar contenido en internet son un dolor de cabeza para los creadores de contenido de diferentes portales de vídeo y servicios de streaming. En el 2018 Google invertirá 100 millones de dólares en el sistema Content ID[1] y son numerosos los escándalos y quejas que usuarios presentan frente a las reclamaciones (en muchos casos injustas) que genera este sistema.
Aunque es posible encontrar música libre de derechos de autor en internet, una alternativa prometedora es utilizar música generada por computadora que sea económica y libre derechos de autor.

Justificación:

En un estudio hecho por Dave Carlton en 2013 llamado: "Popular Songs Statistics"*[2]*, estudió cómo las progresiones musicales ocurren en la música más popular y encontró que el acorde musical más popular es *C*(do), la probabilidad de que despues de *C* halla un *G*(sol) es del 31%, si se sigue mirando las tendencias se encuentra que la mayoría de las progresiones que se usan en gran parte de las canciones se encuentran encasilladas en 4 progresiones. Lo anterior indica que es muy factible encontrar distribuciones de probabilidad estables que modelen las transiciones armónicas que se utilizan en cierto estilo de música.

La generación de música por medio de cadenas de markov es un ejemplo frecuente de aplicación de este modelo*[3]*. Claro ejemplo de ello son los trabajos realizados por Tracy(2013)*[4]* y Rumbo(2017)*[5]* que son base para la concepción de este proyecto. Tracy modeló corales de Johann Sebastian Bach utilizando un Modelo Oculto de Markov capaz de componer corales inspirados en el trabajo de Bach. Para el entrenamiento Tracy construyó la matriz de transiciones a partir de un 7% de los 371 corales compuestos por Bach y obtuvo como resultado un modelo que fue probado experimentalmente con 40 expertos musicales que no pudieron identificar diferencias entre las armonías compuestas por Bach y las arrojadas por el algoritmo. De manera similar Rumbo utilizó cadenas de Markov con restricciones para modelar desde una melodía tan sencilla como Arroz con Leche hasta también corales de Bach.

Trabajos como este y muchos otros que se realizan sobre música generada por un computador (principalmente usando redes neuronales) evidencian que es un objeto importante de estudio que puede ayudar a tratar la problemática descrita anteriormente. Esto debido a que los sistemas de reconocimiento actuales como Content ID de Google comparan archivos de video y audio con

contenido registrado por dueños de contenido en busca de coincidencias para luego hacer retirar el contenido o monetizarlo.

Objetivos:

Objetivo General

Modelar patrones armónicos distintos géneros musicales usando modelos ocultos de Markov para generar música estocástica.

Objectivos Específicos

1. Crear un conjunto de datos de progresiones de acordes que representen una canción o pieza musical mediante el uso de la herramienta Melisma*[6]*.
2. Generar un modelo funcional haciendo uso de la librería hmmlearn*[7]* para ajustar un Modelo Oculto de Markov (HMM)*[8]* a partir del conjunto de datos.
3. Implementar un script de Python que haciendo uso de la librería Music21*[9]* produzca un archivo midi que facilite apreciar la salida del modelo.
4. Someter el resultado a una aplicación de reconocimiento como Shazam para evaluar si podría ser reconocido por un sistema de Copyright.

Alcance:

A partir de la revisión de trabajos similares limitamos el alcance del proyecto al de generar un modelo que explore más ampliamente la generación de progresiones armónicas en distintos géneros musicales, y evaluar si es posible aliviar la problemática del copyright utilizando música generada por este tipo de modelos.

Propuesta de modelo a utilizar:

Para el desarrollo del proyecto se propone un Modelo Oculto de Markov*[8]* entrenado a partir de secuencias de acordes obtenidos de archivos MIDI (Musical Instrument Digital Interface), que son la salida de un Sistema Analizador de archivos MIDI llamado Melisma (Modular Event-List-Input System for Music Analysis.)***[6]***.

El sistema toma como entrada una obra musical representada como “una lista de eventos” (generada a partir de una archivo MIDI) que consiste en una secuencia de notas con sus respectivas afinaciones, tiempo de entrada y salida, y realiza un análisis de la entrada para extraer información de la obra. Se compone de varios módulos que obtienen características diferentes del archivo. Como resultado de este procesamiento se obtiene un archivo de texto con el análisis por números romanos que nos indica el grado de la escala sobre el que está construido cada acorde.

Con el resultado de este análisis se entrenará el modelo que deberá a partir de la secuencia de acordes ajustar sus parámetros de matriz de transición, matriz de emisión y vector de probabilidades iniciales. Se utilizará la implementación de la librería HmmLearn y el algoritmo de Baum-Welch para el entrenamiento del modelo.

Se espera que a partir de estos modelos se puedan generar secuencias de acordes que simulen progresiones armónicas de los diferentes géneros musicales trabajados.

Música creada por computadoras

La música como industria representa 20.2 miles de millones de dólares en 2019, la cual se encuentra un constante crecimiento mas su generación de dinero se encuentra estancada esto dado a que el consumo de música ya no se da de la misma forma que en el pasado, el consumo ya no es específico y no se encuentra representado en un producto físico. esto genera varios problemas en sí el primero siendo el que al no ser representado por algo físico. la facilidad de ser duplicado aumenta y con ello la facilidad de que aquella industria en caída pierda dinero aumenta exponencialmente, con ello también se crearon imponer normas y regulaciones más estrictas sobre las que ya existían para controlar aquella pérdida de dinero esta fue los derechos de autor que aunque en el pasado existían ahora son regulados más minuciosamente lo cual controlo pérdidas de dinero pero al mismo tiempo limitó creativamente a los artistas al cualquier coincidencia musical genera problemas para los mismos al tener que pagar regalías a otros artistas que probablemente no tengan relación en el proceso creativo del artista pero al tener una coincidencia musical es marcado por derechos de autor.

Una solución para aquel problema fue la generación de música mediante computadoras usando modelos e inteligencia artificial, dando lugar a música completamente nueva. Un buen ejemplo es la música generada por AIVA technologies*[16]*.

Modelo Oculto de Markov

"Un modelo oculto de Markov es un modelo estadístico en el que se asume que el sistema a modelar es un proceso de Markov de parámetros desconocidos. El objetivo es determinar los parámetros desconocidos (u *ocultos*, de ahí el nombre) de dicha cadena a partir de los parámetros observables. Los parámetros extraídos se pueden emplear para llevar a cabo sucesivos análisis, por ejemplo en aplicaciones de reconocimiento de patrones."*[9]*
Los modelos ocultos de Markov surgen a partir de los 60s donde fueron descritos por el matemático Leonard Esaú Baum y algunos otros autores. Para la década de los 70 ya se utilizaban en aplicaciones para el reconocimiento del habla. En la actualidad estos modelos son muy utilizados en el reconocimiento de formas temporales surgiendo aplicaciones en reconocimiento del habla, escritura manual, textos, para hacer etiquetado gramatical y son innumerables sus aplicaciones en bioinformática.
Los tres principales problemas asociados con el modelo y sus principales operaciones (algoritmos) son:

- Dada una o más secuencias de observaciones, encontrar el conjunto de estados de transición y probabilidades de emisión que más probables (Entrenar los parámetros del modelo a partir de la secuencia dada). Esto se resuelve gracias al algoritmo de ***Baum-Welch***.
- Dados los parámetros del modelo, computar la probabilidad de generar una secuencia de salida particular. Este problema es resuelto por el algoritmo de avance-retroceso (***forward-backward***).
- Dados los parámetros del modelo, encontrar la secuencia más probable de estados ocultos que puedan haber generado una secuencia de salida dada. Este problema se soluciona con el **algoritmo de *Viterbi***.

Formulación matemática y algoritmo de entrenamiento

Formalmente un modelo oculto de Markov se representa como una tupla $(Q,\ V,\ \pi,\ A,\ B)$ donde:

$Q = \{q_1, q_2, ..., q_n\}$ es el conjunto de estados de la cadena de Markov oculta. n estados.
$V = \{v_1, v_2, ..., v_m\}$ es el conjunto de observaciones posibles en cada estado. m observaciones.
$\pi = \{\pi_1, \pi_2, ..., \pi_n\}$ es el vector de probabilidades iniciales, donde π_i es la probabilidad de empezar en el estado q_i.
$A = \{a_{ij}\}$ es la matriz de transición de la cadena de Markov oculta donde a_{ij} es la probabilidad de pasar del estado q_i al estado q_j en el siguiente instante de tiempo. $a_{ij} = P(q_{t+1} = q_j | q_t = q_i)$. Tiene tamaño $n\ x\ n$.
$B = \{b_j(v_k)\}$ es la matriz de emisión, el conjunto de probabilidades de emitir una observación v_k estando en un estado q_j . $b_j(v_k) = P(o_t = v_k | q_t = q_j)$. Es de dimensiones $n\ x\ m$.

Para el entrenamiento del modelo se utiliza el algoritmo de Baum-Welch*[11]*, el cual es un proceso iterativo que funciona de la siguiente manera:

1. Se parte de un modelo inicial que puede ser seleccionado aleatoriamente.
2. Se realiza el cómputo de las transiciones y emisiones más probables según el modelo actual.
3. Se ajustan las probabilidades dando lugar a un nuevo modelo incrementando las probabilidades de las transiciones y emisiones encontradas en el paso 2. Para la secuencia de observaciones dada el modelo tendrá ahora una mayor probabilidad.

Este proceso de los pasos 2 y 3 se repite hasta que no exista una mejora entre un modelo y el siguiente.

Metas

1. Aplicar el modelo a distintos géneros musicales. Ya que en la bibliografía revisada se encuentran aplicaciones solo a música barroca.
2. Lograr que el factor estocástico del modelo diferencie progresiones armónicas respectivas a cada género para que al ser aplicadas musicalmente no se encuentren en la base de datos de aplicaciones de reconocimiento de copyright.

Creación del Conjunto de Datos

Para la creación del conjunto de datos fue necesario hacer un procesamiento a los archivos MIDI en varias etapas:

Selección y descarga de los archivos midi

Los géneros musicales que se proponen trabajar son: Jazz, Rock, Salsa y Blues. Se escogieron alrededor de 30 canciones representativas por género y se buscaron versiones midi de las mismas en las diferentes páginas de internet que ofrecen estos archivos. Para el procesamiento de los archivos se decidió retirar de los archivos el canal 10 que corresponde en el estándar midi a los eventos de instrumentos de percusión, esto con el objetivo que no influyeran (al tratarse de eventos con tono) en

el análisis de la pieza. Para esto se utilizó la herramienta MuseScore2*[10]*, software musical libre que permite importar archivos midi y modificarlos.

Generación de lista de eventos y análisis por números romanos

El sistema Melisma tiene un programa llamado **mftext** que toma un archivo midi estándar y genera una lista de notas en un archivo de texto (notefile) con el formato Nota [tiempo de inicio] [tiempo de fin] [tono]. El siguiente archivo "beat list" es generado por el programa **meter** que toma el notefile y le añade una lista de filas de pulsos(beats). A partir del archivo beat list el programa **harmony** realiza un análisis de cada pulso y agrega al archivo una lista de acordes. FInalmente el programa **key** con una configuración de análisis por números romanos, analiza los acordes, determina la tonalidad de la pieza y nos ofrece un archivo de texto con el análisis por números romanos:

```
Kostka-Payne Roman Numeral Analysis:
|F: IV V7/VII |iv65 .  |.  V7/VII |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  iv7 |V7/VII iv7 |.  .  |.  .  |
.  .  |.  .  |V7/VII .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |iv7 V7/VII |.  .  |.  .  |.  .  |
iv42 .  |.  .  |.  .  |.  .  |.  V42/VII |.  .  |.  .  |.  .  |.  .  |.  iv42 |V42/VII .  |.  .  |.  .  |.  .  |.  .  |
.  .  |.  .  |iv7 V7/VII |.  .  |.  .  |iv7 .  |.  .  |.  .  |.  .  |.  .  |V7/VII .  |iv7 .  |V7/VII .  |.  .  |
iv7 V7/VII |.  .  |.  .  |.  .  |.  .  |.  iv43 |V43/VII .  |.  .  |.  .  |.  iv43 |.  .  |.  .  |.  .  |.  V43/VII |
.  .  |.  .  |.  .  |.  .  |iv43 .  |.  V43/VII |.  .  |.  .  |.  .  |iv43 V43/VII |.  iv43 |.  V43/VII |.  .  |
.  .  |.  iv43 |.  .  |.  V43/VII |.  .  |iv43 .  |V43/VII .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |
.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  iv43 |.  V43/VII |iv43 .  |.  .  |.  .  |.  V43/VII |.  .  |
.  .  |.  .  |.  iv43 |.  .  |V43/VII .  |.  .  |iv43 .  |.  .  |V7/VII .  |.  .  |.  .  |.  .  |.  .  |iv7 .  |
.  .  |.  .  |.  .  |V7/VII .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |
.  .  |.  .  |.  .  |.  iv7 |.  .  |.  .  |.  .  |.  .  |.  V7/VII |iv7 .  |.  V7/VII |.  .  |iv7 V7/VII |.  .  |
.  .  |iv7 .  |V7/VII .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |.  iv7 |.  .  |.  .  |.  i7 |
.  Dim |.  .  |i7 .  |V7/IV .  |C-: V7/IV .  |Dim V7/IV |.  .  |i7 .  |.  .  |.  .  |.  V7/IV |i7 .  |Dim .  |.  .  |
i7 Dim |i7 V7/IV |.  i7 |.  .  |V7/IV i7 |V7/IV .  |.  i7 |.  V7/IV |.  .  |.  .  |.  .  |.  .  |.  .  |.  .  |
Dim Dim |Dim Dim |i42 I64 |i42 I64 |.  .  |.  V42/IV ]|
```

Figura 2: Ejemplo de análisis por números romanos de la pieza There Will Never Be Another You.

Limpieza de los datos y generación de secuencias de acordes

Debido a que el análisis por números romanos no es una tarea sencilla y en muchos casos se pueden presentar ambigüedades, el resultado que nos ofrece Melisma no es perfecto y en ocasiones no le es posible determinar qué función tonal representa un acorde, por esto es necesario hacer una limpieza de estos datos para no tenerlos en cuenta. Por otra parte el espacio de acordes que se encuentran en la música puede llegar a ser muy grande para entrenar un modelo oculto de markov con este número de características así que las dominantes secundarias (una amplia categoría de acordes que se utilizan para reproducir relaciones de manera distinta a la habitual) las asociamos con funciones tonales básicas con el fin de reducir el número de parámetros del modelo. Para lo anterior se escribió un programa en C++ **clean.cpp** que toma los archivos generado por Melisma para un género, ignora los caracteres que no son relevantes y acordes no definidos, reemplaza las dominantes secundarias por acordes básicos que cumplen la misma función y escribe varios archivos:

- Un primer archivo que contiene la secuencia de todos los acordes de las canciones escritos en la notación de números romanos.
- Un segundo archivo que tiene la descripción de los datos: longitud de la secuencia para cada canción, cuantos acordes fueron encontrados y una asignación numérica a cada acorde encontrado.
- El último archivo generado por el programa contiene la lista de acordes encontrados para luego ser leído por el modelo como un arreglo de observaciones.

Etiquetado

Finalmente es necesario etiquetar los datos para que la implementación de la librería hmmlearn realice el entrenamiento, para esto también se escribió un programa en C++ **traducir.cpp** que etiqueta cada dato con el número asignado por el programa anterior.

Por medio del procedimiento anteriormente descrito se obtuvieron los siguientes datos para cada género musical:

Género	Número de canciones	Número de acordes diferentes encontrados	Número de acordes
Jazz	30	29	2461
Salsa	25	24	1072
Rock	31	24	1043
Blues	31	25	1405

Tabla 1: Descripción del conjunto de datos obtenido.

Planteamiento del modelo

Para aplicar este modelo a la generación de progresiones armónicas las observaciones V del modelo son los acordes encontrados en las secuencias del conjunto de datos de cada género musical. A partir de un entrenamiento con estas secuencias cada modelo ajusta sus matrices de transición y emisión con las cuales puede generar nuevas secuencias de acordes que simulan progresiones armónicas de cada género.

Objetivo de su aplicación

Se espera que una vez entrenado el modelo se puedan reproducir progresiones armónicas de cada género modelado con cierto factor de estocasticidad.

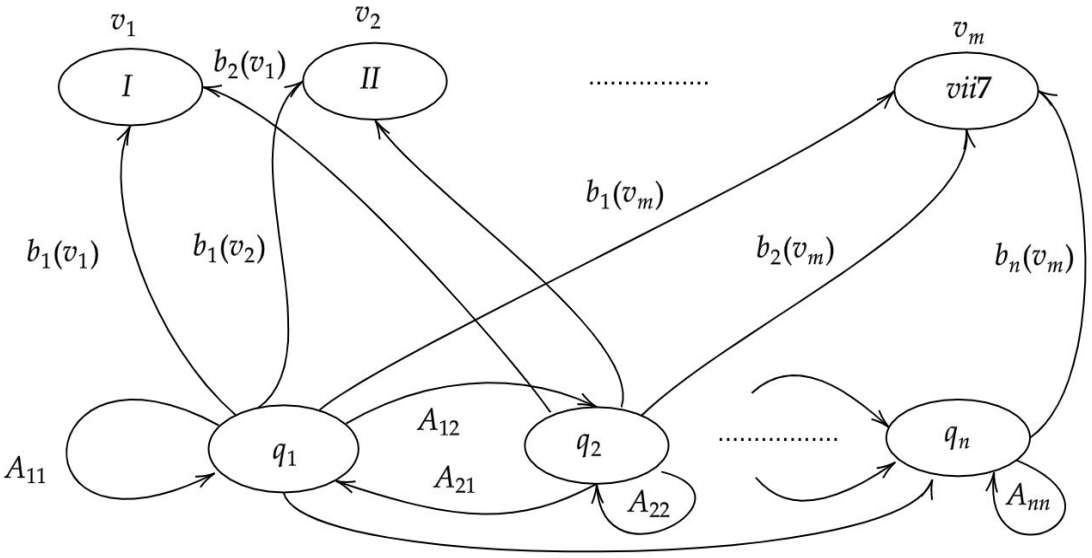
Definición de variables del modelo

Las variables de entrada y salida del modelo serán números enteros que representan los acordes definidos para cada género.

Parámetros y restricciones

Los parámetros del modelo son las matrices A y B y el vector de probabilidades π con las restricciones de que la suma de las probabilidades de cada fila en A y B debe ser igual a 1. Al igual que la suma de todas las probabilidades iniciales en π debe ser 1. Un hiperparámetro importante del modelo son el número de estados de la cadena oculta el cual se asemeja a el número de neuronas de una red neural, se debe experimentar con este número para encontrar el modelo que genere mejores resultados.

Desarrollo gráfico del modelo



$$A = \begin{bmatrix} A_{11} & A_{12} \ldots\ldots A_{1n} \\ A_{21} & A_{22} \ldots\ldots A_{2n} \\ \ldots & \ldots\ldots \\ A_{n1} & A_{n2} \ldots\ldots A_{nn} \end{bmatrix} \qquad \pi = [\pi_1 \ \pi_2 \ldots\ldots \pi_m] \qquad B = \begin{bmatrix} b_1(v_1) & b_1(v_2) & \ldots\ldots & b_1(v_m) \\ b_2(v_1) & b_2(v_2) & \ldots\ldots & b_2(v_m) \\ \ldots\ldots & \ldots\ldots & \ldots & \ldots\ldots \\ b_n(v_1) & \ldots\ldots & \ldots\ldots & b_n(v_m) \end{bmatrix}$$

Figura 1: Descripción gráfica del modelos

Función objetivo y calibración

La estimación de los parámetros de estos modelos se hace mediante el método de máxima verosimilitud*[12]* por lo que se tiene asociada una función de verosimilitud que mide qué tanto se ajusta un modelo estadístico a unos datos de ejemplo. Usualmente las implementaciones de modelos ocultos de Markov utilizan una transformación logarítmica de la función de verosimilitud*[13]* (logaritmo de verosimilitud) para el cálculo de las probabilidades, esto debido a que el cálculo implica productos de probabilidades que puede dar lugar a números muy pequeños que se salen del rango de representación del computador (underflow), por otro lado el logaritmo transforma productos en sumas lo cual es muy conveniente para los cálculos. Al tratarse de productos de probabilidades la función de verosimilitud siempre es menor que 1 y su logaritmo será negativo.

Para este caso la función de verosimilitud es la probabilidad del modelo de producir una secuencia de ejemplo lo cual se calcula con el algoritmo *forward*. Nuestra **función objetivo** es entonces, el logaritmo de verosimilitud y lo que se quiere es maximizar su valor. Para esto se deben ajustar los parámetros e hiperparámetros del modelo.

Un primer intento intuitivo para escoger un número de estados ocultos para el modelo es probar distintos números y escoger el que tenga mejor verosimilitud, sin embargo esta aproximación da lugar a modelos con sobreajuste ya que los modelos con mejor verosimilitud son los que tienen un gran número de estados ocultos, resultando en modelos complejos con muchos parámetros y que conllevan un gran coste computacional.

Criterios de validación AIC y BIC

Cabe recalcar que el problema de estimar la mejor arquitectura para un modelo oculto de markov es un problema abierto. Existen, sin embargo, algunos métodos y criterios de validación que se pueden usar para seleccionar el número de estados. Dos muy utilizados son el Criterio de información de Akaike (AIC)*[14]* y el Criterio de información Bayesiano (BIC)*[15]*. La idea fundamental de estos criterios es añadir un término de penalización asociado a la complejidad del modelo y se busca el modelo que minimice estos valores.

El valor AIC se define como:

$$AIC = 2k - 2ln(\widehat{L})$$

Donde k es el número de parámetros del modelo y $\widehat{L}$ es la máxima verosimilitud del modelo.

El valor del BIC está dado por:

$$BIC = kln(n) - 2ln(\widehat{L})$$

Donde k es el número de parámetros del modelo, n el número de observaciones y $\widehat{L}$ es la máxima verosimilitud del modelo.

Rock		
Número de estados	AIC	BIC
1	5676,8786	5805,5749
2	5016,3324	5283,6247
3	4693,9517	5109,7397
4	4795,1840	5369,3674
5	4635,5178	5377,9962
6	4598,2344	5518,9077
7	4562,8024	5671,5703
8	4565,0861	5871,8483
9	4370,9506	5885,6066
10	4637,9217	6370,3714
11	4455,5526	6415,6958

12	4687,8120	6885,5483
13	4487,6618	6932,8909
14	4741,4459	7444,0675
15	4939,5840	7909,4978
16	5047,7984	8294,9043
17	4842,8230	8377,0205
18	5103,3252	8934,5141
19	5212,9395	9351,0195
20	5301,1868	9756,0576
21	5425,0317	10206,5930
22	5743,7165	10861,8680
23	5799,3903	11264,0318
24	5819,6201	11640,6513
Mejor	9	3

Tabla 2: ejemplo de valores AIC y BIC obtenidos para el género Rock

Finalmente para escoger el número de estados ocultos se tomó de entre los que minimizan los valores AIC y BIC aquel que ofreciera mejor verosimilitud respecto al conjunto de datos.

Entrenamiento y evaluación del modelo

Debido a que el entrenamiento se realiza en base a varias secuencias independientes, la escogencia de un conjunto de datos de prueba no puede realizarse con porcentajes exactos del total de datos. Por lo tanto se planteó la siguiente separación de datos de entrenamiento y datos de prueba.

Género	Número de canciones para entrenamiento	Longitud secuencia de entrenamiento	Número de canciones de prueba	Longitud secuencia de prueba	Porcentaje del total (Prueba)
Jazz	24	1.849	8	612	24,87%
Salsa	19	846	6	226	21,08%
Rock	24	797	7	246	22,93%
Blues	27	1.050	4	355	25,27%

Tabla 3: Partición del conjunto de datos.

Al tratarse de un modelo que se debe inicializar aleatoriamente antes del entrenamiento, el algoritmo puede llegar a estancarse en un mínimo local por lo que el entrenamiento se debe hacer varias veces y escoger el modelo con mejor rendimiento. En este caso el entrenamiento de cada modelo se realizó 25 veces y se obtuvieron los siguientes resultados:

Género	Número de estados Ocultos	Log-likelihood	
		Datos de entrenamiento	Datos de prueba
Jazz	10	-2507,1173	-880,9250

Salsa	10	-1025,8036	-517,3502
Rock	9	-1177,9642	-491,1815
Blues	11	-1112,3412	-270,0336

Tabla 4: Resultados del entrenamiento.

Es importante mencionar que el logaritmo de verosimilitud es función de la longitud de las secuencias de observaciones por lo que por sí solo no refleja bien la bondad de ajuste del modelo y por esta razón también se hizo uso de los criterios AIC y BIC para determinar los modelos que mejor se ajustan a nuestras observaciones.

Generación de música estocástica

Una vez entrenado el modelo podemos hacer una simulación de lo que serían progresiones armónicas de una canción de Jazz, Salsa, Rock o Blues, los modelos permiten generar una secuencia de salida de una longitud deseada.

Para crear un archivo MIDI que permita escuchar las secuencias de salida del modelo se utilizo la Libreria *Music21*, la cual cuenta con un módulo de análisis romano que crea acordes a partir de esta notación lo cual facilita la inserción de acordes en un stream MIDI que posteriormente puede ser exportado. Para cada género se definió un conjunto de instrumentos y a cada uno se le asignó una nota de cada acorde de la secuencia de salida.

Para el ritmo, no fue posible entrenar modelos que generaran secuencias rítmicas debido a la complejidad de la extracción de datos para ello y las limitaciones que tiene la librería en cuanto a instrumentos de percusión, por lo que se optó por asignar aleatoriamente una figuración rítmica para cada nota de cada acorde. Para esto se implementaron funciones en Python que a partir de un número aleatorio y un tono en los acordes de salida, generan notas que se agregan a cada parte de la partitura que constituyen la pieza final. Para enriquecer más los ejemplos generados por el modelo se tomaron partes de Batería representativas de cada género y se agregaron manualmente a la partitura.

Procedimiento de resolución

Se dará resolución a las problemática mediante la generación de piezas musicales muy sencillas basadas en la salida del modelo, y posteriormente sometiendo aquella música generadas por el programa a aplicaciones de reconocimiento de copyright.

Resultados obtenidos

Como resultado se obtuvieron cuatro Modelos Ocultos de Markov que por medio de métodos de Aprendizaje Estadístico modelan las progresiones armónicas encontradas en un conjunto de datos elaborado. Se superaron problemas de sobreajuste en los modelos haciendo uso de los criterios AIC y BIC, esto siguiendo el principio de modelación de simplicidad. Y se facilitó la generación de música estocástica a partir de estos modelos.

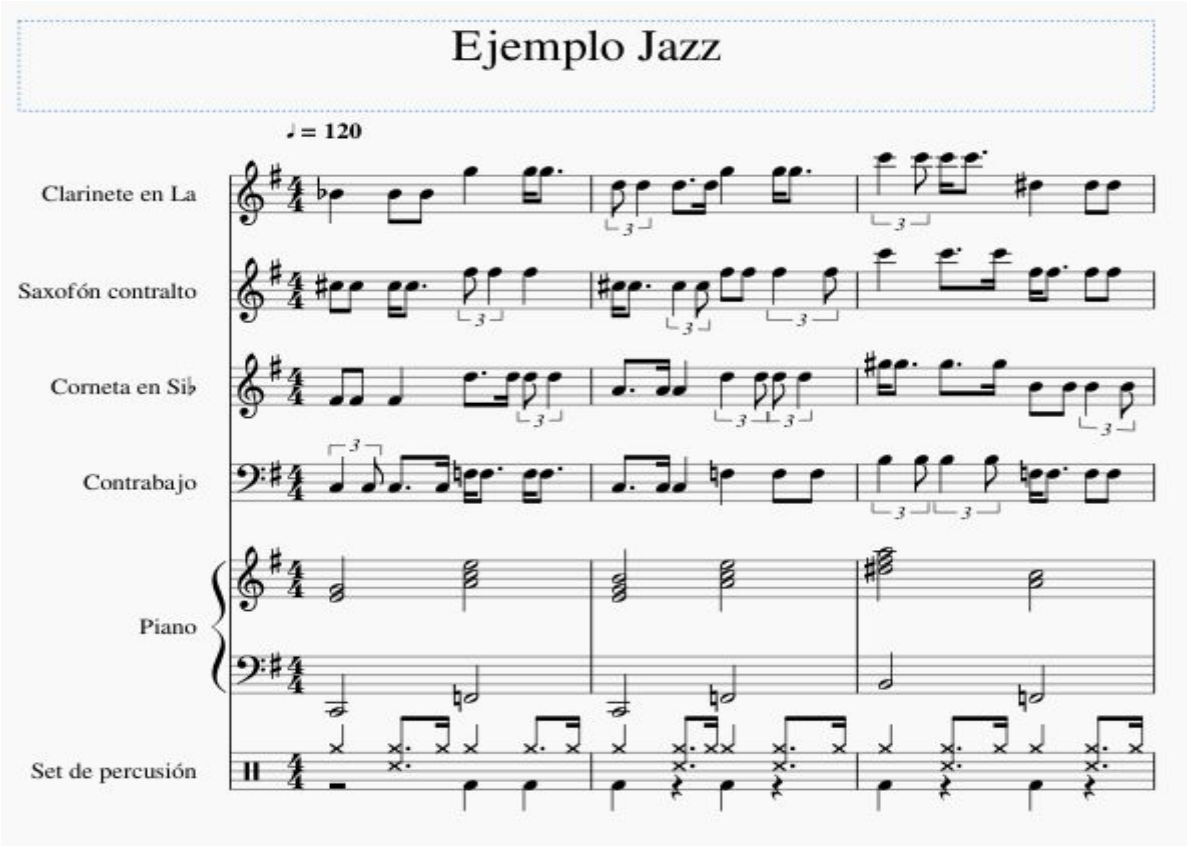


Figura 3: Ejemplo de partitura obtenida.

Otra aplicación interesante que se puede tener de estos modelos, es la de identificar la probabilidad de que una secuencia no clasificada sea generada por alguno de los modelos, dando una idea de si esta secuencia no identificada pueda pertenecer a uno de los géneros modelados.

Se sometieron audios generadas por el programa a la aplicación de reconocimiento Shazam obteniendo como resultado que ninguno de ellos fue relacionado con alguna canción existente en la base de datos, lo cual sugiere que a partir de las secuencias generadas por estos modelos se pueden crear metodologías de bajo costo, bien sea por artistas o algún grupo de desarrollo, para crear música original que no sea detectada por sistemas de reconocimiento y de esta manera mermar la problemática planteada de copyright.

Conclusiones

- Es posible modelar patrones armónicos de distintos géneros musicales distintos utilizando Modelos Ocultos de Markov y aplicarlos en la generación de música estocástica.
- Los modelos ocultos de Markov como alternativa para modelar progresiones armónicas pueden ser un primer paso para la generación de música estocástica.
- Las piezas generadas por el sistema no fueron reconocidas por una de las principales aplicaciones de reconocimiento.
- Es posible modelar otros aspectos secuenciales de la música utilizando estos modelos si se dispone de los datos necesarios.

Glosario

Música estocástica: Generación de elementos musicales estocásticos mediante procesos matemáticos. Utilizar procesos estocásticos para componer música.
Algoritmo forward-backward: Algoritmo de programación dinámica que calcula la probabilidad de una secuencia de observables.
Algoritmo de Viterbi: Algoritmo de programación dinámica que permite hallar la secuencia más probable de estados ocultos que produce una secuencia observada de sucesos.
Algoritmo de Baum-Welch: Algoritmo que permite estimar los parámetros de un Modelo Oculto de Markov que maximizan la probabilidad de una secuencia dada.
Cadena de Markov: Proceso estocástico discreto en el que la probabilidad de que ocurra un evento depende solamente de del evento inmediatamente anterior.
Dominantes secundarias: Construcción de un acorde de séptima dominante sobre el quinto grado de una escala mayor para crear tensión a resolver.
Función tonal: Función que desempeña un acorde dentro de la tonalidad.

Repositorio

El notebook, el dataset y los programas realizados para crear el dataset están disponibles en el siguiente repositorio de GitHub:
https://github.com/jhsrojasro/Stochastic-Music-using-HMM.git

Referencias

[1] Content ID (System) https://en.wikipedia.org/wiki/Content_ID_(system)
[2] 1300 songs chords analyzed:
https://www.hooktheory.com/blog/chord-progression-search-patterns-and-trends/
[3]Markov chain for music generation, Alexander Osipenko(2019).
https://towardsdatascience.com/markov-chain-for-music-generation-932ea8a88305
[4] Bach in beta: Modeling Bach chorals with Markov Chains. Michaela Tracy (2013). Harvard University.
http://www.people.fas.harvard.edu/~msantill/Mauricio_Santillana/Teaching_files/Michaela_Tracy_thesis_Final.pdf
[5] Cadenas de Markov con restricciones y aplicación a la composición automática de música tonal. Verónica Rumbo. Universidad de la República,Uruguay. 2017.
http://www.cmat.edu.uy/~vrumbo/monografia
[6] The Melisma Music Analyzer: https://www.link.cs.cmu.edu/melisma/.
[7] hmmlearn documentation page:
https://hmmlearn.readthedocs.io/en/latest
[8] Music21 documentation:
https://web.mit.edu/music21/doc/index.html
[9] Modelo Oculto de Markov: https://es.wikipedia.org/wiki/Modelo_oculto_de_Márkov
[10] MuseScore: https://musescore.org/es
[11] Algoritmo de Baum-Welch: https://es.wikipedia.org/wiki/Algoritmo_de_Baum-Welch
[12] Método de máxima verosimilitud. https://en.wikipedia.org/wiki/Maximum_likelihood_estimation

[13] Logaritmo de verosimilitud y cómo interpretarlo. https://www.seh-lelha.org/que-es-el-metodo-de-estimacion-de-maxima-verosimilitud-y-como-se-interpreta/
[14] Criterio de información Akaike. https://en.wikipedia.org/wiki/Bayesian_information_criterion
[15] Criterio de información Bayesiano. https://en.wikipedia.org/wiki/Akaike_information_criterion
[16] AIVA technologies página oficial. https://www.aiva.ai