Universidad Internacional de La Rioja (UNIR)

ESIT

Máster Universitario en Inteligencia Artificial

Composición de Música Flamenca para Guitarra mediante Técnicas de Inteligencia Artificial.

Trabajo Fin de Máster

Presentado por: Tapia Velázquez, José Luis

Director/a: Bello Méndez, Yan

Ciudad: Aranjuez (Madrid)
Fecha: 22/07/2021

Agradecimientos

Este trabajo de fin de Máster, realizado en la Universidad Internacional de la Rioja, es el resultado de un gran esfuerzo en el que han participado numerosas personas de una forma directa o indirecta, dando su más sincera opinión acerca del desarrollo de este trabajo, corrigiendo cada una de las entregas y alentándome en los momentos más difíciles.

En primer lugar, a mi director del trabajo de fin de máster, Yan Bello Méndez, mi más amplio agradecimiento por haberse prestado a dirigir este trabajo y por todos sus consejos.

Como no, agradecer al resto de docentes del Máster en Inteligencia de Artificial por todo lo aprendido en las diferentes asignaturas, por sus grandes conocimientos, por el espíritu de grupo, por el buen clima de enseñanza que han logrado crear y por su paciencia.

Por supuesto, a mis compañeros de Máster que se merecen muchas y buenas palabras, ya que con ellos he compartido clases, prácticas, horas de trabajo y conversaciones enriquecedoras. Por todo ese tiempo que hemos compartido, por el buen humor, el buen ambiente y la amistad, gracias.

También, quiero agradecer a mi profesor de guitarra flamenca todos los conocimientos adquiridos en sus clases a lo largo de estos años.

Finalmente, todo esto no hubiera sido posible sin el amparo y el cariño de mi familia, en especial mi mujer y mi hijo, que, de forma incondicional, entendieron mis ausencias y el tiempo de trabajo que exige este Máster. A pesar de mi poca dedicación a la familia durante estos 2 años, siempre estuvieron apoyándome y animándome para conseguir terminar estos estudios. Sin ellos no lo hubiera podido conseguir y no tengo palabras suficientes para mostrarles el agradecimiento que realmente se merecen.

A todos vosotros, gracias de todo corazón.

"Un extracto de fuego y de veneno, eso es el Flamenco".

(Antonio Gades)

Resumen

La generación de contenidos musicales, a partir de información contenida en datasets perteneciente a uno o varios estilos mediante el uso de técnicas de Deep Learning, es un campo de estudio dentro de la Inteligencia Artificial que está en pleno auge.

Los algoritmos y frameworks existentes en la actualidad son capaces de generar música de diferentes estilos entre los que se podrían incluir los diferentes palos flamencos, pero no tienen en cuenta las particularidades rítmicas y melódicas propias de estos, principalmente porque solo utilizan como entrada las melodías y, en algunos casos, los acordes que la acompañan.

Con el fin de poder componer música flamenca del estilo de la Soleá, se ha incluido el compás como un elemento más de entrada para que la música que se genere siga el ritmo propio de la Soleá. Adicionalmente, se utilizarán mecanismos de temperatura variable para conseguir aleatoriedad musical dentro del estilo flamenco sin perder las características melódicas propias de la Soleá.

Palabras Clave: Inteligencia Artificial, IA, composición musical, flamenco.

Abstract

The generation of musical content from information contained in datasets belonging to one or several styles using Deep Learning techniques is an area within Artificial Intelligence that is booming.

The algorithms and frameworks currently available are capable of generating music of different styles, including the different flamenco styles, but they do not take into account the rhythmic and melodic particularities of these styles, mainly because they only use as input the melodies and, in some cases, the chords that accompany them.

To be able to compose flamenco music in the Soleá style, the compás has been included as an additional input element so that the music generated follows the rhythm of the Soleá. Additionally, variable temperature mechanisms will be used to achieve musical randomness within the flamenco style without losing the melodic characteristics of the Soleá.

Keywords: Artificial Intelligence, AI, music composition, flamenco, Soleá.

Índice de contenidos

Índice de tablas

Índice de figuras

1. Introducción

1.1 Motivación

La motivación, para la realización de este trabajo de fin de Máster, viene inspirada por la naturaleza innovadora de la generación de música flamenca utilizando un sistema de Inteligencia Artificial.

Los conocimientos, que se necesitan para la realización de este estudio, son: la Inteligencia Artificial y la música flamenca. Ambos mundos pertenecen a áreas del conocimiento muy distantes y que aparentemente parecen estar desconectadas.

Por esta razón, realizar un trabajo de estas características, que intenta aunar la tecnología más actual con el arte flamenco de ancestrales raíces y conseguir generar este tipo de contenidos musicales, es un reto y la principal motivación.

La Inteligencia Artificial, dentro del ámbito de la creatividad computacional y la generación de contenidos musicales, no ha tratado todavía la generación de música flamenca de forma específica. No hay trabajos concretos para la generación automática de este tipo de música. Únicamente hay sistemas de reconocimiento del cante para los diferentes palos flamencos (Mora Merchán, 2018).

Las preguntas, que surgen y que son las líneas directrices de este trabajo, son: ¿la música generada por algoritmos de IA puede realmente sonar a "*flamenco*" ?, ¿es posible crear música para diferentes palos flamencos respetando el ritmo y las tonalidades?, ¿puede un sistema basado en Inteligencia Artificial aprender las particularidades de este tipo de música?, ¿es posible medir la calidad musical de las composiciones realizadas?

Finalmente, este estudio plantea ser una aportación al auge y desarrollo del área de la creación computacional y acercar el flamenco al mundo de la tecnología y viceversa.

1.2 Planteamiento del Trabajo

El flamenco tiene particularidades melódicas y rítmicas que lo hacen muy especial y diferente de otros estilos musicales.

Una de estas particularidades es el "*compás*" que tienen los estilos flamencos, también denominados "*palos*", que hace que la melodía siempre siga un patrón rítmico del cual no es posible salirse. Esto implica que la melodía comience en un determinado compás, fluya y varíe durante otros compases y se cierre o finalice exactamente en el mismo compás.

Esta rítmica, tan especial, no es tratada por los algoritmos de generación musical que solo utilizan como entrada las melodías de un estilo o de diferentes estilos y, en algunos casos, los acordes que la acompañan.

Con el fin de aportar la rítmica, que necesitan los algoritmos para generar música de estilo flamenco, se plantea el uso de información adicional. Básicamente, consiste en añadir a la información, existente en los dataset de entrada, los datos de la rítmica del palo flamenco. Otra forma, puede ser la utilización de mecanismos dentro del ámbito de la Inteligencia Artificial para que el sistema de generación de composiciones musicales aprenda las particularidades de forma no supervisada.

Dado que el flamenco está formado por multitud de estilos, resultaría sumamente ambicioso abordar todos y cada uno de ellos, por lo que es necesario acotar cuales van a formar parte de este estudio. Sin duda, uno de los estilos más representativos de la música flamenca es la Soleá y este será el palo flamenco sobre el que se basará la generación de música en este proyecto de fin de Máster.

Plantear un sistema de composición de música flamenca para el estilo de la Soleá, además de ser un reto, puede significar un avance en la creación de contenidos flamencos y una herramienta de ayuda para los compositores y para los que no conocen esta música tan singular.

1.3 Estructura de la Memoria

Seguidamente, se describe la estructura y contenido de este trabajo de fin de Máster:

Capítulo 1: Introducción, donde se indica la motivación del trabajo y el planteamiento de éste.

Capítulo 2: Presentación del contexto y de los resultados del estudio del estado del arte, donde se realiza una revisión histórica de los sistemas de generación de contenidos artísticos musicales, tanto de la composición musical como de la generación de música, así como una revisión de las técnicas actuales de Deep Learning.

Capítulo 3: Descripción del objetivo general, de los objetivos específicos y la metodología que se ha utilizado para el desarrollo del trabajo.

Capítulo 4: Descripción de cada uno de los experimentos. Se detallan los realizados para la generación de música flamenca del estilo de la Soleá mediante el uso del framework de Google Magenta y los realizados utilizando redes neuronales especificas con arquitecturas LSTM.

Capítulo 5: Descripción de los resultados de cada una de las fases del estudio realizado.

Capítulo 6: Discusión de los diferentes resultados obtenidos durante el estudio.

Capítulo 7: Conclusiones y trabajos futuros.

Capítulo 8: Referencias y bibliografía adicional.

Anexos: Contiene varios apartados en los que se incluyen las diferentes representaciones de la música, las particularidades de la música flamenca, código utilizado en los experimentos, partituras MIDI generadas y, por último, un artículo de investigación sobre el trabajo que se ha realizado.

2. Contexto y Estado del Arte

La música del mundo antiguo es de la que menos conocimiento se tiene porque apenas existen registros, pero sí se tiene conocimiento de que en el antiguo Egipto existían arpas, flautas e instrumentos de percusión.

En Grecia, cuna de la civilización occidental, sabemos que la palabra "*música*" viene de "*μουσική* [*τέχνη*] - *mousikē* [*téchnē*]- *el arte de las musas*", de las que se decía que inspiraban las actividades creativas e intelectuales de los seres humanos. La música, además de ser protagonista en la sociedad, también aparece en varios de sus mitos; como el de Orfeo que con su música podía dominar a las bestias más feroces.

El "*Epitafio de Sícilo*", que data del siglo I D.C, es una columna de mármol griega que contiene inscrita la pieza musical más antigua del mundo que se conserva de forma completa. Fue mandada construir por Sícilo para recordar a su mujer Euterpe, cerca de la actual Éfeso en Turquía (Palisca & Burkholder, 2006).

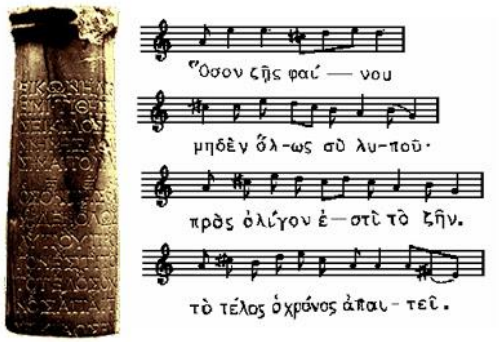

Figura 1. Epitafio de Sícilo. Fuente: (Misterica, 2017)

Desde estos orígenes hasta la actualidad, la música ha ido evolucionando lentamente y se ha visto influenciada por las diferentes épocas, la religión, las corrientes artísticas y la tecnología. En la sociedad de hoy en día, la música sigue evolucionando y desarrollándose, creándose incesantemente nuevos estilos musicales, instrumentos y sonidos que permiten realizar nuevas composiciones.

La creación de música de forma automática nace con el inicio de la informática y de los primeros ordenadores, siendo actualmente un campo de investigación en auge dentro la Inteligencia Artificial. Incluso grandes empresas han invertido en proyectos para llevar estas

innovaciones musicales a los usuarios, como es el caso del proyecto Google Magenta (Google, 2016) o el proyecto Flow Machines (Sony CSL, 2016).

Derivadas de los avances de la Inteligencia Artificial, surge la recuperación de información musical o MIR (Music Information Retrieval) que permite analizar las composiciones musicales y extraer características como el género, el tempo, compás, tonalidad, progresión de acordes, estructura de la composición, etc. (Downie, 1993) (Tjoa, 2021).

La composición mediante Inteligencia Artificial continúa avanzando inexorablemente y cada vez más compositores pueden generar composiciones de forma automática, pero surge la pregunta de si alguna vez una pieza compuesta íntegramente por un ordenador podrá superar a una creada por un ser humano y que sea recordada durante años por su calidad artística. De momento, la composición mediante técnicas de Inteligencia Artificial permite la creación automática de bandas sonoras o música sincronizada, que se utiliza en spot publicitarios, documentales, videos corporativos, es decir, en aquellos ámbitos donde la música forma parte de fondo (background) (Sympathy for the Lawyer, 2017).

La Inteligencia Artificial es mejor que cualquier humano jugando al ajedrez (IBM, 2011) o al tradicional juego chino del Go (BBC Online, 2016) cuando tenemos una función de evaluación (evaluación heurística) perfectamente definida que cumple la misión de asignar un valor numérico a las posiciones analizadas, lo que permite decidir qué camino tomar en el árbol de variantes (Internet Archive, 2003). Sin embargo, como indica (Pachet, 2018), en el mundo de la creatividad no existe una única función de evaluación que permita valorar numéricamente el grado de artisticidad de una composición musical o una obra de arte. Además, en el caso de la música, la evaluación depende de las personas que la escuchan, la cultura y de la época, pudiendo ocurrir que, composiciones que pasaron desapercibidas en su estreno, ahora son consideradas obras maestras.

En los últimos años, el aprendizaje profundo (Deep Learning) ha alcanzado un éxito sorprendente y actualmente se utiliza en multitud de ámbitos para tareas de clasificación, predicción, reconocimiento de imágenes, reconocimiento de voz, traducción a diferentes lenguas y generación de contenidos artísticos.

Dentro de la Inteligencia Artificial, y más concretamente en el área de Deep Learning, es necesario destacar los grandes avances que se realizan en la "*generación de contenidos*" (Generative Deep Learning) y concretamente en la "*generación de contenidos musicales*", consistente en el uso de datasets o corpus de música para aprender de forma automática los estilos musicales y generar nuevos contenidos basados en ellos.

Dentro del ámbito de la generación de contenidos musicales, debemos hacer la distinción entre la "*composición musical por ordenador*" y la "*generación de música por ordenador*". Este último también denominado "*composición musical algorítmica*", se basa en el uso de un proceso formal, que incluye pasos (algoritmo) y componentes, para componer música (Cope, 2000). Según John A. Maurer (Maurer, 1999), existen diferentes subdivisiones, enfoques e hitos relevantes dentro de la composición algorítmica que merecen ser destacados y que se indican en los siguientes apartados.

2.1 Composición Algorítmica

Composición Algorítmica – sin ordenador

La composición algorítmica, también denominada "*composición automatizada*", hace referencia, según Adam Alpern (Alpern, 1995), a la "*utilización de algún proceso formal para hacer música con una mínima intervención humana*". Estos "*procesos formales*" se han utilizado para la generación de música desde la antigüedad, no obstante, el término "*algoritmo*" se adoptó en informática y computación hacia la segunda mitad del siglo pasado. Los algoritmos y su implementación en computadores han permitido a los compositores automatizar el proceso de composición. Además, en los últimos años se han desarrollado métodos más eficaces para hacerlo.

La idea de utilizar instrucciones y procesos formales para crear música se remonta en la historia musical a los griegos en la antigüedad. Pitágoras, creía en una relación directa entre las leyes de la naturaleza y la armonía de los sonidos expresada en la música. Ptolomeo, creía que las leyes matemáticas "*subyacen en los sistemas de intervalos musicales y de los cuerpos celestes*", y que ciertos modos musicales e incluso ciertas notas "*se corresponden con planetas concretos, sus distancias entre sí y sus movimientos*" (Grout, Donald Jay. Palisca, 2001)**.**

Mozart, también utilizó técnicas de composición automatizada en su "*Musikalisches Wurfelspiel*" (música de dados), un juego musical que según Adam Alpern (Alpern, 1995) consistía en *"ensamblar pequeños fragmentos musicales, y combinarlos de forma aleatoria, formando de este modo, una nueva pieza musical con partes elegidas al azar*".

Con posterioridad a la Segunda Guerra Mundial, surge "*el método de los doce tonos*" y "*el serialismo*" (Griffiths, 1980), que trataron de controlar todos los parámetros de la música y de objetivar y abstraer el proceso de composición musical. Las decisiones sobre notas, ritmos, etc. estarían sujetas a "*series*" y "*matrices*" de valores preconfigurados que "*automatizan*" el

orden en que deben aparecer en una composición musical. Estas series y matrices eran los "*algoritmos*" que permitían la composición de piezas, como el "*estudio para piano*" (1949) y el "*Mode de valeurs et d'inensites*" (1949), ambas de Olivier Messiaen.

Podemos encontrar ejemplos más actuales de composición algorítmica en John Cage, que al igual que Mozart, utilizó el azar en sus composiciones creando piezas como "*Reunión*"[1] (1968) y "*Atlas Eclipticalis*"[2] (1961), que se compuso colocando papel de partitura sobre cartas astronómicas y colocando notas simplemente donde se encontraban las estrellas, delegando de nuevo el proceso de composición en la indeterminación (Schwartz, 1993).

Composición Algorítmica – con ordenador

La primera pieza musical creada por un ordenador apareció en 1957. Era una breve melodía de 20 segundos de duración llamada "*The Silver Scale*" por su autor Newman Guttman (Guttman, 1957) y fue generada por un software de síntesis de sonido llamado "*Music I*", desarrollado por Mathews en los Laboratorios Bell.

Ese mismo año, apareció "*The Illiac Suite*" que se considera la primera partitura compuesta por un ordenador para un instrumento tradicional (Swada, 2014). Se llamó así por el ordenador ILLIAC I de la Universidad de Illinois (UIUC) (Internet Archive, n.d.). Los meta-compositores humanos fueron Lejaren A. Hiller y Leonard M. Isaacson, ambos músicos y científicos. La partitura de la pieza fue compuesta por el ordenador y luego transpuesta a la notación musical tradicional para ser interpretada por un cuarteto de cuerda. Lo que Hiller e Isaacson habían hecho era:

1. Generar materiales en bruto con el ordenador.
2. Transformar estos materiales aplicándoles varias funciones.
3. Y, por último, seleccionar los mejores resultados según diferentes reglas.

Este paradigma de "*generador/modificador/selector*" también se aplicó posteriormente a MUSICOMP (acrónimo de MUsic SImulator-Interpreter for COMpositional Procedures). Considerado uno de los primeros sistemas informáticos para la composición de forma automática, fue desarrollado por Hiller y Robert Baker y realizó en 1963 "*Computer Cantata*". MUSICOMP fue desarrollado como una biblioteca de subrutinas pequeñas y bien definidas que facilitaba el proceso de escribir programas de composición, lo que ha hecho que este

[1] En un tablero equipado con fotorreceptores los diferentes movimientos de las piezas realizados por los jugadores producen sonidos, de este modo la pieza musical que se crea durante es diferente en cada partida.
[2] Se creo poniendo papel pautado encima de cartas astronómicas y asignando notas musicales donde se encontraban las estrellas.

enfoque fuera muy popular llegando hasta muchos sistemas de composición algorítmica actuales (Di Nunzio, 2013).

Otro uso pionero del ordenador en la composición algorítmica es el de Iannis Xenakis, que desarrollo un programa informático que generaba datos probabilísticos para sus composiciones "*estocásticas*". Como ejemplo de composiciones tenemos "*Atrées*" (1962) y "*Morsima-Amorsima*" (1962). En este caso, y al igual que en los casos anteriores, hay que señalar que "*el ordenador no ha producido realmente el sonido resultante; sólo ha ayudado al compositor en virtud de sus cálculos de alta velocidad*" (Cope, 1984). En esencia, lo que el ordenador producía, no era la composición en sí, sino el material con el que Xenakis podía componer. En cambio, el trabajo de Hiller e Isaacson intentaba simular por completo el proceso de composición, delegando por completo las decisiones creativas en el ordenador. Ambos casos (Xenakis e Hiller) fueron los primeros ejemplos de composición algorítmica que utilizaban modelos estocásticos (cadenas de Markov) para la generación, así como reglas para filtrar el material generado según las propiedades deseadas.

Un segundo enfoque de la composición algorítmica mediante el ordenador es el de los sistemas "*basados en reglas*" y las "*gramáticas formales*". Un proceso basado en reglas se centraría en una serie de pasos (reglas) que conducen al siguiente paso nuevo. Ejemplos de este enfoque son el "*canon*" del siglo XV del periodo renacentista, así como el "*método de los doce tonos*" y el "*serialismo integral*", citados anteriormente. En la actualidad, este tipo de enfoque se implementa en forma de procesos informáticos que acceden a bases de datos de reglas y que recogen las diferentes técnicas de composición almacenadas. Un ejemplo de uso de un método de composición algorítmica "*basado en reglas*" es el del programa de contrapunto automático de especies de William Shottstaedt, que genera música basándose en las reglas del "*Gradus ad Parnassum*" de Johann Joseph Fux (libro de instrucciones de contrapunto de principios del siglo XVIII). Otro ejemplo basado en reglas es el del sistema automatizado de Kemal Ebcioglu llamado "*CHORAL*", que genera corales a cuatro voces en el estilo de J. S. Bach a partir de un sistema basado en más de 350 reglas (Ebcioglu, 1988).

Un último enfoque, es el de los sistemas informáticos basados en Inteligencia Artificial. Estos sistemas se asemejan a los sistemas basados en reglas, en el sentido de que son programas que se basan en alguna gramática predefinida; sin embargo, los sistemas de Inteligencia Artificial tienen la capacidad adicional de "*aprender*". Un ejemplo de este enfoque es el sistema de David Cope llamado "*Experimentos de Inteligencia Musical*" (EMI) (Cope, 2000). Al igual que el ejemplo anterior de Shottstaedt y de "*CHORAL*" de Ebcioglu, EMI utiliza una gran base de datos de reglas para almacenar, tanto las descripciones de los diferentes estilos como las estrategias de composición. Sin embargo, EMI también tiene la capacidad de crear su propia

base de datos de reglas, que el propio sistema deduce a partir de las partituras de la obra del compositor que le hayan introducido. Este sistema se ha utilizado para componer automáticamente música que evoca los estilos de Bach, Mozart, Bartók, Brahms, Joplin, etc.

Otra rama interesante de las técnicas de Inteligencia Artificial es la de los "*algoritmos genéticos*" que, en lugar de basar sus reglas en las partituras introducidas en el ordenador, generan sus propios materiales musicales y conforman sus propias reglas. Por lo tanto, el compositor debe programar una función "*crítica*" que escuche los numerosos resultados producidos automáticamente en las distintas etapas del procesamiento para decidir cuáles son aptos para la salida final (Alpern, 1995).

2.2 Composición Musical por Ordenador

Según los estudios de Jean-Pierre Briot (J. P. Briot, 2020), en la composición musical por ordenador (computer-based music composition) podemos considerar dos enfoques diferentes en cuanto a la participación humana en un proceso de creación musical asistido por ordenador:

1. **Generación autónoma**: Algunos ejemplos recientes son productos de empresas como Amper, AIVA o Jukedeck, basados en diversas técnicas como, por ejemplo, reglas, Reinforcement Learning, Deep Learning, etc. y destinados a la creación de música original para anuncios y documentales. En estos sistemas autónomos la generación musical está automatizada y el papel que desempeña el usuario es el de parametrizador de las características de estilo, emoción, etc.

2. **Asistencia a la composición**: Estos sistemas ayudan a los músicos (compositores, arreglistas, productores musicales) en la creación de piezas musicales. Como ejemplo podemos destacar los entornos FlowComposer desarrollado por Sony CSL-Paris (Sony CSL, 2016) y OpenMusic desarrollado por IRCAM basado en el leguaje Lisp (Agon Amado, 1998). En estos sistemas, altamente interactivos, el usuario va componiendo de forma incremental con la ayuda del entorno.

Como señala Jordi Pons (Pons, 2018), una primera oleada de aplicaciones, basadas en redes neuronales artificiales para la composición musical, apareció a finales de la década de 1980. Esto se corresponde con la segunda ola del movimiento de redes neuronales, con la innovación de las capas ocultas y el método de cálculo de "*backpropagation*" para el entrenamiento de redes neuronales.

Los experimentos realizados por Peter Todd (Todd, 1989) fueron las primeras exploraciones acerca de cómo utilizar redes neuronales artificiales para realizar composiciones musicales.

Aunque las arquitecturas que inicialmente propuso no se utilizan hoy en día, sus experimentos fueron pioneros y siguen siendo un referente en la actualidad.

La arquitectura denominada "*Time-Windowed*" fue la primera que creo Peter Todd con el objetivo de generar una melodía monofónica de forma iterativa. Este diseño considera una ventana deslizante de periodos de tiempo sucesivos de tamaño fijo (en la práctica, un compás). La generación de música se realiza de forma iterativa, segmento melódico por segmento melódico y de forma recursiva, ya que el segmento de salida generado se incluye como el siguiente segmento de entrada y así sucesivamente. Esta arquitectura aprende las correlaciones por pares entre dos segmentos melódicos sucesivos, pero no es capaz de aprender las correlaciones a largo plazo ya que no dispone de una memoria dedicada a tal fin.

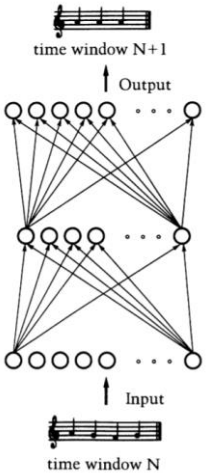

Figura 2. Arquitectura Time Windowed. Fuente: (Todd, 1989)

Un diseño más avanzado que el anteriormente expuesto, es el denominado "*Secuencial*" (Todd, 1989), donde la capa de entrada de la red neuronal se divide en dos partes, denominadas respectivamente "*contexto*" y "*plan*". El "*contexto*" es la memoria real de la melodía generada hasta el momento y consiste en neuronas correspondientes a cada nota musical (D4 a C6), más una neurona sobre la nota con información de la nota inicial de la melodía (anotada como "*nb*" en la figura de abajo). Por lo tanto, el contexto recibe información de la capa de salida (output layer) con la siguiente nota que se genera mediante una conexión de reentrada correspondiente a cada neurona. Por otro lado, el "*plan*" es una forma de nombrar a la melodía que la red ha aprendido. Esta arquitectura "*Secuencial*" de Todd, se considera uno de los primeros ejemplos de utilización de una arquitectura recurrente y una estrategia iterativa para la generación de música. Además, la introducción de una entrada

adicional, denominada "*plan*", que representa una melodía que la red ha aprendido, podría considerarse como un precursor de las arquitecturas condicionadas, en las que se utiliza una entrada adicional específica para condicionar (parametrizar) el entrenamiento de la arquitectura.

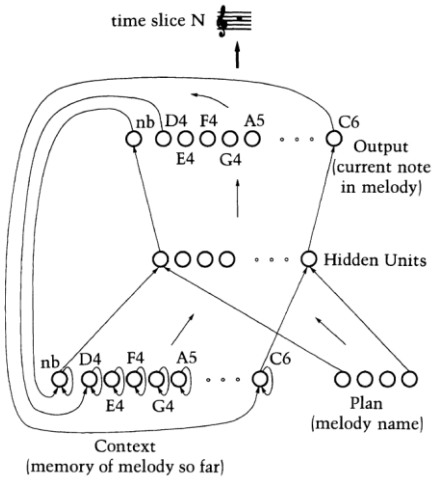

Figura 3. Arquitectura Secuencial. Fuente: (Todd, 1989)

Los estudios de J.P. Lewis (Lewis, 1988) introdujeron una forma novedosa de crear melodías, que se denominó creación por refinamiento (CBR) al "revertir" la forma estándar de utilizar el descenso de gradiente para ajustar los pesos y minimizar el error de clasificación. Esta forma consistía en ajustar la entrada para que la salida de la clasificación resultase positiva. En su experimento inicial, Lewis utilizó una arquitectura feedforward tradicional utilizada para la clasificación binaria para catalogar melodías "*bien formadas*". La entrada consistía en una melodía de 5 notas, cada una de las cuales se encuentra entre las 7 notas musicales (de Do a Si, sin alteraciones). Para la fase de entrenamiento, Lewis construyó manualmente 30 ejemplos de lo que entendía por melodías "*bien formadas*" y también construyó ejemplos de melodías "*mal formadas*". El entrenamiento de la red es, por tanto, convencional, entrenándola con los ejemplos positivos (bien formados) y negativos que se han construido. Para la fase de creación por refinamiento, se genera un vector con valores aleatorios que son introducidos a los nodos de entrada de la red. Seguidamente, se aplica una optimización por descenso de gradiente de forma iterativa para refinar estos valores con el fin de maximizar una clasificación positiva. De este modo, se crea una nueva melodía que se clasifica como "*bien formada*". El proceso puede repetirse, generando un nuevo conjunto de valores aleatorios y controlando su ajuste para crear una nueva melodía "*bien formada*".

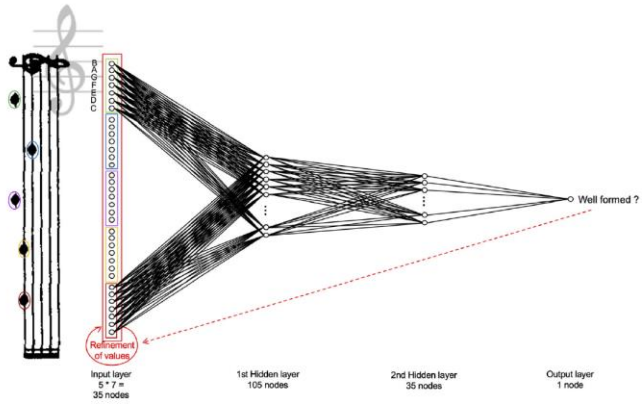

Figura 4. Creation by Refinement. Fuente: (Lewis, 1988)

2.3 Arquitecturas Deep Learning

Desde el punto de vista de Jean-Pierre Briot (J. P. Briot et al., 2020), con la tercera ola de redes neuronales artificiales denominada Deep Learning, se han propuesto nuevos tipos de arquitecturas y los trabajos realizados sobre la generación de música se han beneficiado de la era del Big Data, caracterizada por la gran capacidad de procesamiento de las máquinas y la gran disponibilidad de datos, lo que ha permitido realizar experimentos a gran escala.

Este mismo autor (J. P. Briot et al., 2020) opina que, la motivación para el uso de las técnicas de Deep Learning en la generación de contenidos musicales es su "*generalidad*". A diferencia de los modelos basados en reglas, los sistemas de generación musical basados en Deep Learning pueden considerarse como "*agnósticos*". En este sentido, el mismo sistema puede utilizarse para entrenar diferentes estilos musicales.

A continuación, se indican las principales arquitecturas utilizadas en Deep Learning para generar contenidos musicales.

2.3.1 Arquitectura Feedforward

Las redes neuronales con topología feedforward se caracterizan por ejecutar el procesamiento de datos en una sola dirección. Este tipo de redes constan de tres capas (layers) principales de unidades independientes de cómputo llamadas neuronas:

1. Capa de entrada (input layer): Donde se reciben los datos que van a procesarse.
2. Capa intermedia (hidden layer): Donde se realiza el procesamiento propiamente dicho.
3. Capa de salida (output layer): Donde se obtienen los valores generados.

Las redes neuronales artificiales tipo feedforward (RNAf) son fáciles de programar, por lo que son muy populares entre la comunidad científica.

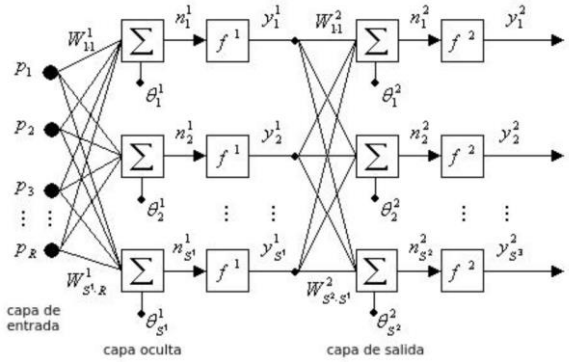

Figura 5. Esquema de bloques de una red feedforward. Fuente: (Arroyo-Hernández et al., 2013)

Si bien este tipo de redes feedforward no se utiliza de forma aislada para la generación de contenidos musicales, sí se utilizan dentro de arquitecturas más complejas como es el caso de DeepBatch (Hadjeres et al., 2017) que combina 2 redes LSTM y 2 redes feedforward para generar música coral del estilo de Bach.

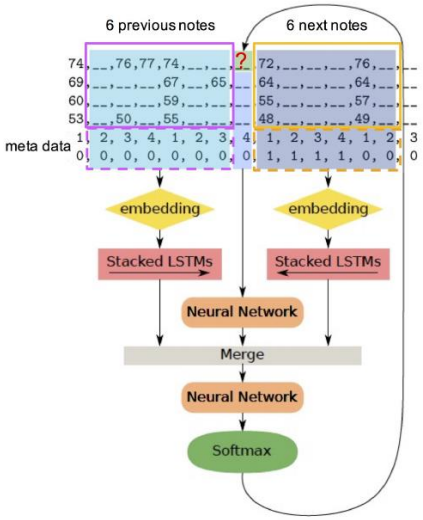

Figura 6. Arquitectura DeepBach. Fuente: (J.-P. Briot & Pachet, 2017)

En esta arquitectura se utilizan las redes LSTM para procesar las notas anteriores y las notas futuras, mientras que la primera red feedforward se utiliza para procesar las notas que se producen al mismo tiempo. Estas 3 salidas se combinan (merge) y se pasan como entrada a una segunda red feedforward para su procesamiento final.

2.3.2 Arquitectura Convolucional (CNN)

Las redes convolucionales (CNN) son un tipo de red neuronal propuestas por Yann André Lecun (Lecun et al., 1998) que aplicó este tipo de arquitectura al reconocimiento de dígitos escritos manualmente.

Posteriormente, este tipo de redes se aplicaron en el reconocimiento de imágenes y se hicieron muy populares tras ganar AlexNet en 2012 la competición de clasificación de imágenes ImageNet (ImageNet Large Scale Visual Recognition Challenge). Hoy en día, son ampliamente utilizadas en diferentes campos de la Inteligencia Artificial entre el que destaca la visión por computador (detección de objetos, segmentación, etc.).

Este tipo de redes utiliza una técnica especial llamada "*convolución*". En matemáticas, la convolución es una operación sobre dos funciones que genera una nueva función que expresa cómo la forma de una es modificada por la otra.

Las CNN están compuestas por múltiples capas, cada una de ellas especializada en la detección de unas determinadas características o patrones. De este modo, la primera capa suele extraer características básicas como los bordes horizontales o diagonales. Esta salida se pasa a la siguiente capa, que detecta otras características más complejas basadas en las anteriores, como bordes o esquinas. A medida que se avanza en profundidad en la red, se pueden identificar características más complejas, como pueden ser objetos, formas etc.

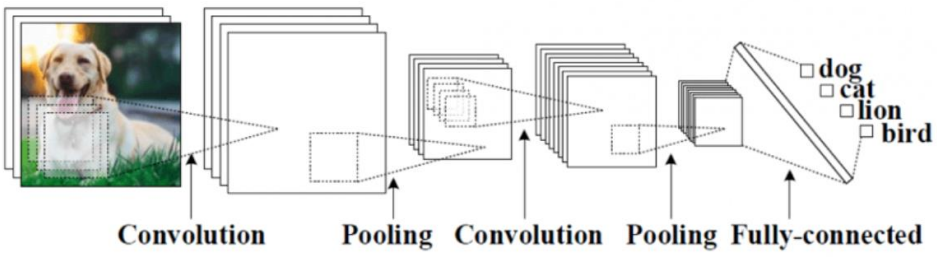

Figura 7. Estructura de una red Convolucional. Fuente: (KDnuggets, 2021)

Aunque este tipo de redes son muy populares en la visión por computador, también han sido utilizadas para la generación de música, como es el caso de WaveNet (Oord et al., 2016). Este modelo tenía como entrada y salida audio en crudo (raw audio) y, aunque generaba música, tenía el inconveniente de que ésta variaba cada poco tiempo en su estilo musical, el volumen de la música, la calidad, los instrumentos, etc.

2.3.3 Arquitectura Recurrente (RNN)

En los apartados anteriores se han tratado las arquitecturas de redes neuronales que han sido la base de diferentes trabajos y proyectos para generar música. Sin embargo, si tuviéramos que destacar unas arquitecturas, que sobresalen entre todas las demás por su amplia utilización y su capacidad para generar contenidos musicales, esas serían las redes neuronales recurrentes (RNN).

Una red neuronal recurrente (RNN) es un tipo de red neuronal que utiliza datos secuenciales o de series temporales. Su particularidad y elemento característico es su memoria interna o "*hidden state*" que le permite persistir lo ocurrido anteriormente y utilizarlo para influir en la salida actual.

Si comparamos las RNN con las redes neuronales anteriores (feedforward, CNN, etc.) se puede ver que, en las primeras, la salida depende de los elementos anteriores dentro de la secuencia, mientras que, en las segundas, no existe relación entre las entradas y salidas (son independientes entre sí).

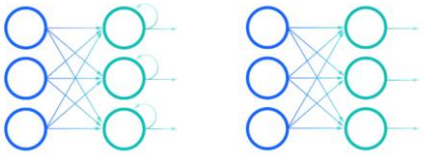

Figura 8. Comparación de RNN (izquierda) y Feedforward (derecha). Fuente: (IBM, 2020)

Las RNN son muy potentes en el tratamiento de análisis de textos y se utilizan habitualmente para la traducción de idiomas, el Procesamiento del Lenguaje Natural (PLN) y el subtitulado de imágenes, incluso se incluyen en aplicaciones tan populares como Siri o en Google Translate. Además, gracias a la capacidad que tienen de predecir (con cierta probabilidad) el siguiente valor de un conjunto de datos en base a los valores previos, las hacen idóneas para su uso en la generación de música.

Uno de los primeros de proyectos de composición musical que utilizaban las RNN fue el sistema CONCERT (Mozer, 1994). Uno de los hitos que se consiguieron con la creación de este sistema fue demostrar que conseguía aprender la estructura subyacente de los datos de entrenamiento, lo que le permitía componer nuevas melodías. Aunque estas composiciones eran en ocasiones agradables, adolecían de falta de coherencia global.

Otros trabajos, como los realizados por Gaetan Hadjeres and Frank Nielsen (Hadjeres & Nielsen, 2017), presentan una arquitectura novedosa que denominaron "*Anticipation-RNN*" que tiene las ventajas de las arquitecturas RNN, al tiempo que permite aplicar las restricciones posicionales definidas por el usuario. Utilizaron esta arquitectura para generar piezas musicales al estilo de Bach que cumplieran las restricciones de posición en las partes de soprano.

Existe gran variedad de RNN dependiendo del modo en el que se realiza la retropropagación ("*backpropagation*") y en función del número de capas ocultas (hidden layers). Los tipos más interesantes para la generación de música se describen a continuación:

LSTM: (Long Short Term Memory) son un tipo especial de RNN que fueron introducidas por Hochreiter & Schmidhuber (Hochreiter & Schmidhuber, 1997) y refinadas por muchos investigadores en trabajos posteriores. Esta topología de red es capaz de aprender y memorizar las relaciones entre los datos a largo plazo.

Las LSTM están pensadas para evitar el problema de la dependencia a largo plazo. Se puede afirmar que este tipo de redes fueron diseñadas para que su comportamiento por defecto fuera almacenar y recordar información durante largos periodos de tiempo. Esto permite resolver el problema del "*exploding/vanishing gradient*" que sufren las RNN y que dificulta el aprendizaje de secuencias largas de datos.

Todas las RNN tienen el aspecto de un conjunto de módulos de red neuronal que se repiten. En las RNN estándar, el módulo que se repite de forma sucesiva tiene una estructura muy sencilla, basada en una única capa tanh (función de activación no lineal que regula los valores que pasan por la red, manteniendo los valores entre -1 y 1).

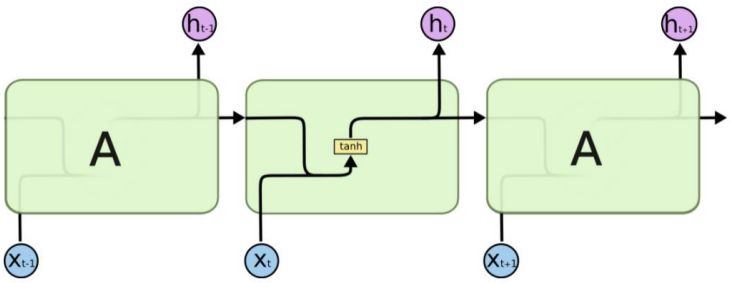

Figura 9. Módulo de repetición en una RNN estándar de una sola capa. Fuente: (Olah, 2015)

Las LSTM heredan la estructura en forma de cadena de las RNN, pero el módulo que se repite de forma sucesiva cambia ligeramente. Básicamente, la diferencia consiste en que hay cuatro capas de redes neuronales que interactúan entre sí, en lugar de haber una única capa de red neuronal como ocurre con las RNN.

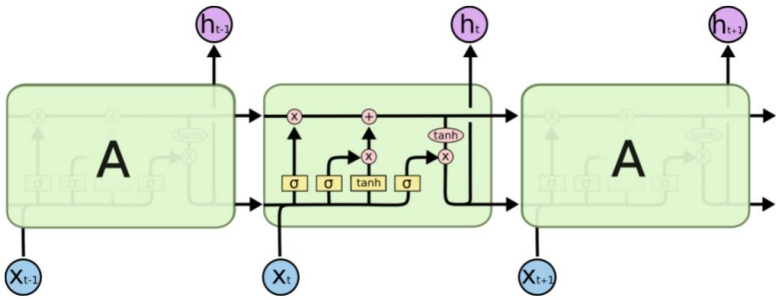

Figura 10. Módulo de repetición de una LSTM con cuatro capas. Fuente: (Olah, 2015)

La clave de las LSTM es el "*state cell*" que como se puede ver es la línea horizontal que atraviesa la parte superior de la figura.

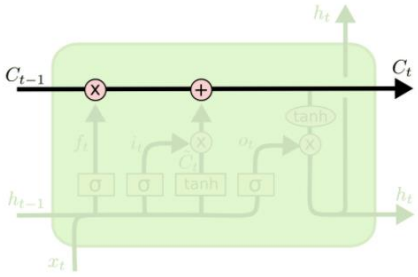

Figura 11. Detalle de "state cell". Fuente: (Olah, 2015)

Este "*state cell*" es una especie autopista de información que fluye en línea recta por toda la cadena. De hecho, lo más probable es que la información fluya a través de ella sin experimentar cambio alguno. La LSTM tiene la capacidad de borrar o añadir información al "*state cell*" mediante unas estructuras denominadas "*gates*". Estas puertas están diseñadas para permitir el paso de información de forma opcional y están formadas por una capa de red neuronal sigmoide y una operación de multiplicación. La capa sigmoide genera valores entre 0 y 1, que indican la cantidad de información que se deja pasar. Un valor de 0 significa "*no dejar pasar nada*", mientras que, un valor de 1 significa "*dejar pasar todo*".

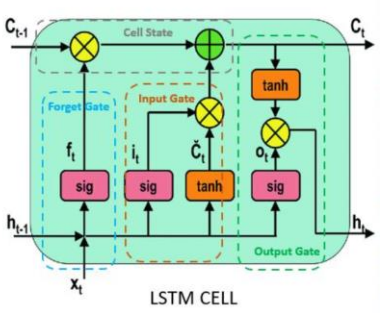

Figura 12. LSTM Cell. Fuente: (Singhal, 2020)

Las puertas que componen las redes LSTM son:

- Puerta de olvido (forget gate): Decide qué información necesita atención y cuál puede ignorarse, o lo que es lo mismo, que información se mantiene y cual se olvida.
- Puerta de entrada (input gate): Decide qué información es relevante para actualizar en el estado actual de la célula de la unidad LSTM.
- Puerta de salida (output gate): Determina el estado oculto actual que pasará a la siguiente unidad LSTM.

Las redes LSTM se utilizan en diversas áreas de la Inteligencia Artificial, entre los que destacan la generación de contenidos musicales. De los numerosos estudios y proyectos que se realizaron, hay que destacar las investigaciones de Douglas Eck y Jurgen Schmidhuber sobre la composición musical mediante redes LSTM (Eck & Schmidhuber, 2002). En estos se destacaba, que la música compuesta por las RNN adolece de una falta de estructura global ya que no pueden hacer un seguimiento de los eventos temporalmente distantes que indican la estructura musical general. Sin embargo, demostraron que las LSTM aprenden con éxito y son capaces de componer melodías nuevas en el estilo en el que se han entrenado. En los experimentos que se realizaron, se utilizaron datasets de música de blues de 12 compases y sorprendentemente, una vez que la red encontraba la estructura relevante, no se desviaba de ella. Es más, la LSTM era capaz de generar música blues con buen ritmo y una estructura adecuada.

Otro ejemplo de aplicación de LSTM a la generación musical es el proyecto ”*Music transcription modelling and composition*” (Sturm et al., 2016) donde se aplicaron métodos de Deep Learning, especialmente redes LSTM, al modelado y la composición de transcripciones musicales. Se utilizaron unas 23.000 transcripciones de música celta expresadas en notación ABC para entrenar las redes LSTM de 3 capas con 512 neuronas por capa, con el fin de generar nuevas composiciones musicales.

Redes GRU: (Gated Recurrent Unit) Son un tipo especial de RNN que utilizan un sistema similar de puertas (gates) parecido al descrito anteriormente para las LSTM. Las mayores diferencias con las LSTM son que se combina el "*cell state*" y el "*hidden state*" en un solo elemento, así como la "*forget gate*" y la "*input gate*" en una única puerta.

A diferencia de la red LTSM descrita anteriormente, y que consta de 3 puertas, la red GRU contiene únicamente dos puertas que controlan y deciden qué información de entrada debe pasarse a la salida:

- Puerta de actualización (update gate): Ayuda a determinar la cantidad de información pasada que debe transmitirse al futuro.
- Puerta de reinicio (reset gate): Es responsable de la memoria a corto plazo de la red. Es decir, permite al modelo decidir la cantidad de información pasada que se debe olvidar y cual debe persistir.

Existe diferentes estudios acerca de este tipo de redes y sus aplicaciones a la generación de música entre los que cabe destacar el "*Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*" (Chung et al., 2014) en el que se comparan diferentes tipos de RNN en las tareas de modelado de música polifónica y de señales de voz. Estos trabajos revelaron que las GRU eran mejores que las RNN más tradicionales, como la tanh y prácticamente iguales a las LSTM.

Otros trabajos de interés en composición musical, fueron los realizados por Aran Nayebi y Matt Vitelli y recogidos en "*Algorithmic Music Generation using Recurrent Neural Networks*" (Nayebi & Vitelli, 2015). Estos estudios se centraban en la comparación del rendimiento entre GRU y LSTM para la tarea de generación algorítmica de música que utilizan como entrada ondas de audio. Los resultados indicaron que las salidas generadas por la red LSTM eran significativamente más plausibles desde el punto de vista musical que las de la GRU.

Redes LSTM Bidireccionales (BLSTM):

Mike Schuster and Kuldip K. Paliwal presentaron las redes LSTM bidireccionales en su trabajo "*Bidirectional Recurrent Neural Networks"* (Schuster & Paliwal, 1997) donde detallaban el funcionamiento de este tipo de redes derivadas de las LSTM. En este estudio indicaban que este modelo de procesamiento de secuencias constaba de dos LSTM. La primera tomaba la entrada en dirección hacia delante y la segunda en dirección hacia atrás.

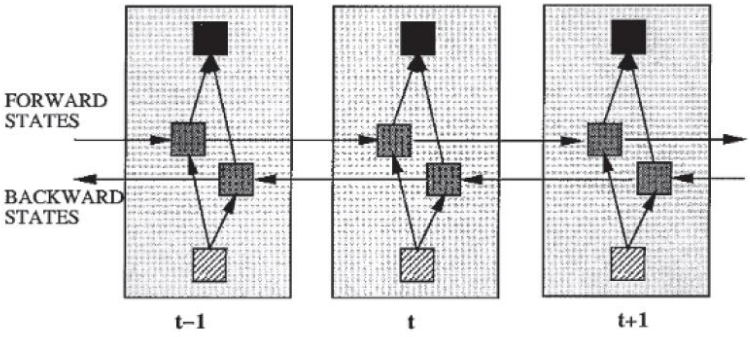

Figura 13. Estructura general de la red neuronal recurrente bidireccional (BRNN) desplegada para 3 pasos temporales. Fuente: (Schuster & Paliwal, 1997)

Estas redes aumentan efectivamente la cantidad de información disponible para la red, mejorando el contexto disponible para el algoritmo. Si utilizamos estas redes en la generación musical se podría saber que notas siguen y preceden inmediatamente a una nota en un compás dentro de una partitura.

Existen diferentes trabajos donde se utilizan este tipo de redes aplicadas a la composición musical, entre ellos se pueden destacar "*Chord generation from symbolic melody using BLSTM networks*" (Hyungui LIm, Seungyeon Rhyu, 2017), en el que se presenta un método novedoso para generar secuencias de acordes a partir de una melodía simbólica utilizando una arquitectura de red BLSTM con 2 capas y 128 neuronas cada una. La red se entrenó en base a un dataset de más de 2.000 de partituras de música moderna (jazz, rock, etc.).

Las secuencias musicales generadas por el modelo basado en BLSTM fueron comparadas con las generadas por otras arquitecturas, confirmándose que las secuencias de acordes generadas por el modelo BLSTM eran preferidas por los oyentes frente al resto.

2.3.4 Arquitectura de Redes Generativas Adversarias (GAN)

Las Redes Generativas Adversarias (GAN) son un tipo de redes neuronales dentro de la categoría de Deep Learning.

Una Red Generativa Adversaria (GAN) tiene dos partes:

- El generador: Aprende a generar datos plausibles. De este modo, los datos generados se convierten en ejemplos negativos de entrenamiento para el discriminador.
- El discriminador: Aprende a distinguir los datos falsos del generador de los datos reales. El discriminador penaliza al generador por producir resultados inverosímiles.

Cuando comienza el entrenamiento, el generador produce datos, obviamente falsos, y el discriminador aprende rápidamente a distinguir que son falsos. A medida que el entrenamiento avanza, el generador se acerca cada vez más a producir unos datos de salida que puedan engañar al discriminador.

Finalmente, si el entrenamiento del generador va bien, el discriminador empeora a la hora de diferenciar si lo generado es real o si es falso. Por lo tanto, el discriminador empieza a clasificar los datos falsos como reales y su precisión disminuye.

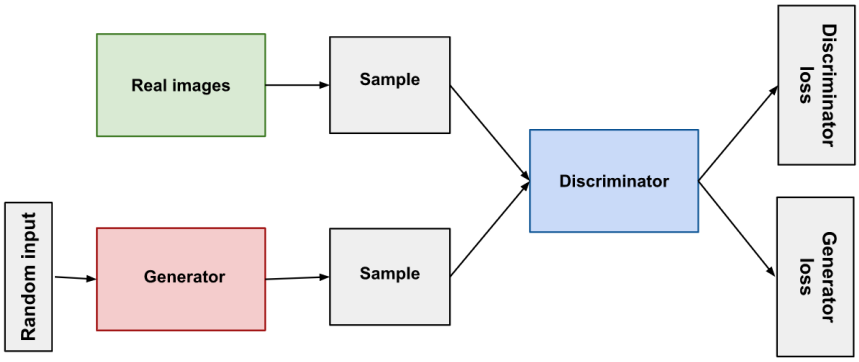

Figura 14. Arquitectura GAN. Fuente: (Google Developers, 2019)

Tanto el generador como el discriminador son redes neuronales. La salida que produce la red del generador se conecta directamente a la entrada de la red del discriminador. Mediante la técnica de "*backpropagation*", la clasificación del discriminador proporciona una señal que el generador utiliza para actualizar sus pesos.

Este tipo de redes se han utilizado en trabajos como "*MidiNet*" (Yang et al., 2017), donde se han utilizado redes CNN para la generación de melodías (en formato MIDI). Además del generador, se utiliza un discriminador para aprender las distribuciones de las melodías, lo que la convierte en una Red Generativa Adversaria (GAN). Adicionalmente, se proponen mecanismos condicionales para explotar el conocimiento previo, de modo que el modelo pueda generar melodías desde cero siguiendo una secuencia de acordes o condicionando la melodía de los compases anteriores con una melodía inicial de cebado, entre otras posibilidades. El modelo resultante puede ampliarse para generar música con múltiples canales MIDI (varias pistas).

La evaluación se realizó mediante un estudio de usuario donde se comparaba la melodía de ocho compases generada por "*MidiNet*" y por los modelos "*Melody_rnn*" de Google Magenta. Los resultados mostraron que "*MidiNet*" tenía un rendimiento comparable al de los modelos

"*Melody_rnn*" en cuanto a que eran realistas y agradables de escuchar, pero las melodías generadas por "*MidiNet*" eran mucho más interesantes.

Otro ejemplo interesante es el sistema "*Musegan*" (Dong et al., 2018). En este sistema se proponen tres modelos para la generación simbólica de música multipista en el marco de las Redes Generativas Adversarias (GAN). Los tres modelos, que difieren en los supuestos subyacentes y, en consecuencia, en las arquitecturas de red, se denominan modelo de interferencia, modelo de compositor y modelo híbrido. Los tres modelos propuestos se entrenaron con un dataset de más de cien mil compases de música rock con el fin de generar piano-rolls de cinco pistas (bajo, batería, guitarra, piano y cuerdas).

Este estudio demostró que estos 3 modelos podían generar música coherente de cuatro compases partiendo de cero (es decir, sin entradas humanas). También se ampliaron los modelos para la generación de música cooperativa entre humanos e Inteligencia Artificial, consistente en que, dada una pista específica compuesta por un humano, el sistema era capaz de generar el resto de las pistas instrumentales de forma automática para acompañarla.

2.3.5 Aprendizaje por Refuerzo (RL)

El Aprendizaje por Refuerzo (RL) es un tipo de técnica de aprendizaje automático que permite a un agente aprender en un entorno interactivo por ensayo y error utilizando el feedback que proporciona sus acciones.

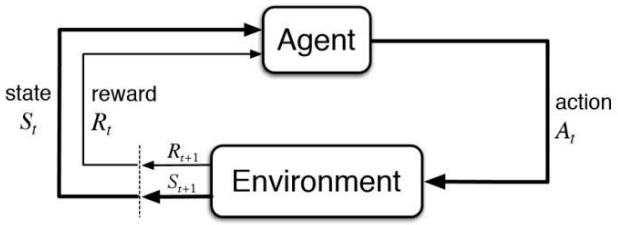

Figura 15. Aprendizaje por Refuerzo. Fuente: (Bhatt, 2018)

Las áreas de utilización del RL son aquellas que requieran muchos datos, como los juegos y la robótica. Otras aplicaciones del RL incluyen motores de resumen de texto, agentes de diálogo (texto, voz) y generación de contenidos musicales.

Este tipo de arquitectura se utiliza en el sistema "*Bach2Bach*" (Kotecha, 2018) donde se aplica una arquitectura de RL que predice y genera música polifónica alineada con las reglas musicales con las que ha sido entrenada. El modelo está basado en una red LSTM entrenada

con un "*núcleo*" similar a un kernel convolucional. Para fomentar la exploración e imponer una mayor coherencia global a la música generada, se adopta un enfoque de Deep Learning por refuerzo (DQN). Los resultados de este sistema, cuando se analizan cuantitativa y cualitativamente, muestran que este enfoque utilizando RL funciona bien en la composición de música polifónica.

Otro trabajo interesante ha sido el realizado por el equipo de Natasha Jaques (Jaques et al., 2017) en el que se utiliza una LSTM que se entrena con un gran corpus de canciones para predecir la siguiente nota en una secuencia musical. A continuación, esta RNN de notas se refina mediante RL, donde la función de recompensa es una combinación de recompensas basadas en reglas de la teoría musical, así como la salida de otra RNN ya entrenada. En este estudio se demuestra que al combinar LSTM y RL no sólo puede producir melodías más agradables, sino que, además, puede reducir significativamente los comportamientos no deseados y algunos fallos de la RNN.

2.4 Refinements

Son métodos o algoritmos que permiten mejorar tanto el rendimiento como los resultados que se obtienen de las redes neuronales. Los principales Refinements que se utilizan en la generación de música son:

2.4.1 Auto-Encoders (AE)

Un Auto-Encoder (AE) es un sistema de Aprendizaje no Supervisado en el que durante el entrenamiento la salida esperada es una aproximación de la entrada. El Auto-Encoder se aplica principalmente a la reducción de la dimensionalidad de los datos, a la clasificación y a la eliminación de ruido de las imágenes, así como a la detección de objetos.

Un Auto-Encoder se compone de las siguientes partes según (Goodfellow et al., 2016):

- Encoder: Una red neuronal que produce una representación de espacio latente comprimido de los datos de entrada.
- Espacio latente: Captura la entrada en una representación del conocimiento. Es decir, reducir la dimensionalidad de la entrada de forma que se conserve la máxima información en ella.
- Decoder: Una reconstrucción de los datos de entrada a partir del espacio latente comprimido.

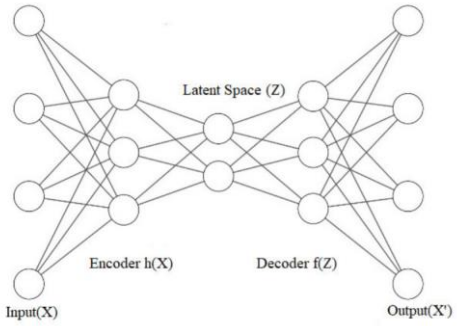

Figura 16. Arquitectura Auto-Encoder. Fuente: (Wei et al., 2020)

Como se muestra en la imagen, el codificador h codifica la información original X en un espacio latente Z. El decodificador f decodificará el espacio latente Z para recrear una aproximación de los datos originales X', de forma que X' = f (Z) = f (h (X)). Tras un entrenamiento iterativo, el Auto-Encoder intentará reproducir una copia de la entrada X como salida X'.

La aplicación de un Auto-Encoder tiene dos funciones principales, primera, la eliminación de datos, y segunda, la reducción de la dimensionalidad para eliminar las características redundantes o sin importancia. En otras palabras, la salida se hace aproximadamente igual a la entrada con algunas restricciones en el algoritmo Auto-Encoder. Estas restricciones obligan al codificador a considerar qué partes de la entrada deben preservar y qué partes se pueden descartar. Por lo tanto, el Auto-Encoder a menudo puede aprender las características significativas de los datos y descartar las características irrelevantes. Es bien sabido que un Auto-Encoder logra una reducción de la dimensionalidad similar a un PCA no lineal. En la aplicación de clasificación de un Auto-Encoder, la sección decodificadora se elimina después de entrenar el Auto-Encoder, y se sustituye por una red clasificadora.

2.4.2 Variational Auto-Encoders (VAE)

Los Variational Auto-Encoders (VAE) son modelos de Deep Learning que mezclan las redes neuronales con las distribuciones de probabilidad y que han tenido un gran éxito en muchas aplicaciones como la generación de imágenes, el subtitulado de imágenes, el diseño de proteínas, la predicción de mutaciones y los modelos de lenguaje, entre otros.

El objetivo de un VAE es aprender la distribución de los datos de entrenamiento para que, muestreando a partir de ella, se puedan generar nuevos datos. Dado que los datos de entrenamiento no tienen necesariamente una distribución matemática bien definida, se fuerza

la distribución de la salida del codificador (conocida como espacio latente) para que siga una distribución conocida como, por ejemplo, una distribución normal.

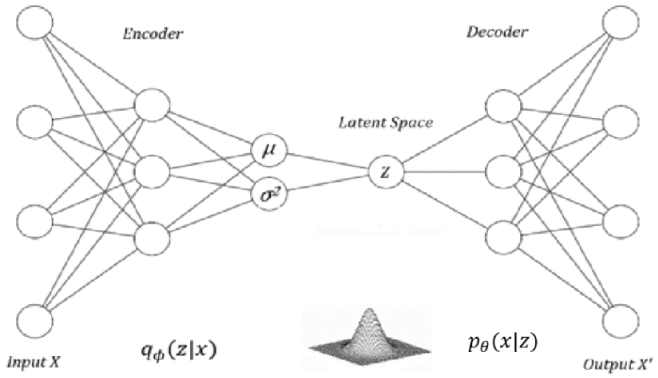

Figura 17. Arquitectura VAE. Fuente: (Wei et al., 2020)

En la figura se muestra la arquitectura de un VAE que tiene un codificador y una red de inferencia variacional, seguida del decodificador que toma muestras del espacio latente para generar la salida. La principal diferencia entre el Auto-Encoder y el VAE es que el Auto-Encoder aprende la representación comprimida de la entrada y su descompresión para ajustarse a la entrada dada. En cambio, el VAE, es un modelo bayesiano que aprende la representación comprimida del Auto-Encoder y construye los parámetros que representan la distribución de probabilidad de los datos. Puede tomar muestras de esta distribución y generar nuevas muestras de datos de entrada. Por lo tanto, el VAE es un modelo generativo, mientras que un Auto-Encoder, que sólo hace la reconstrucción, no tiene una interpretación generativa obvia.

Este concepto ha dado lugar a una enorme investigación y a variaciones en el diseño de los VAE en los últimos años, creando un campo propio, denominado "*aprendizaje de representación no supervisado*".

Este tipo de topologías también se han utilizado en la composición musical. En este sentido, se puede señalar el estudio realizado por el equipo de Adams Roberts (Roberts et al., 2018) en el que ponen de manifiesto que los VAE tienen dificultades para modelar secuencias con estructura a largo plazo. Para solventar esta problemática, se propone el uso de un decodificador jerárquico, que primero produce "*Embeddings*" para subsecuencias de la entrada y luego utiliza estos "*Embeddings*" para generar cada subsecuencia de forma independiente. Esta estructura evita el problema del "*colapso posterior*", que sigue siendo un problema para los VAE recurrentes. Se aplicó esta arquitectura a la modelización de

secuencias de notas musicales y se descubrió que el rendimiento de muestreo, interpolación y reconstrucción es mucho mejor que el de un modelo básico plano.

2.4.3 Mecanismos de Atención

El mecanismo de "A*tención*" surgió dentro del área de PLN como una mejora del sistema de traducción automática mediante redes neuronales que estaba basado en el modelo Codificador-Decodificador. Posteriormente, este mecanismo, y sus diferentes evoluciones, se utilizaron, además del área de PLN, en otras aplicaciones, como la visión por ordenador, el procesamiento del habla y la generación de contenidos artísticos.

Este mecanismo puede considerarse "*bioinspirado*" desde el ámbito de estudio de la neurociencia cognitiva ya que estos mecanismos atencionales en humanos permiten al cerebro focalizarse en una zona concreta de una imagen de entrada y prestar menos atención a otras zonas de una imagen. Este mecanismo llevado a la composición musical permite que la red se centre en determinadas notas, compases o pasajes que son más representativos para la generación de secuencias musicales, obviando o prestando menor atención al resto.

En un principio la traducción automática mediante redes neuronales estaba basaba en RNN/LSTM[3] codificadoras-decodificadoras con una arquitectura similar a esta:

- La RNN/LSTM codificadora procesa todas las palabras de la frase de entrada y la codifica en un vector de contexto. Este vector es el último estado oculto de la RNN/LSTM. En esta arquitectura todos los estados intermedios del codificador se ignoran y el estado final es el estado oculto inicial del que partirá el decodificador.

- Las unidades RNN/LSTM del decodificador producen las palabras de una frase, una tras otra.

Es decir, hay dos RNN/LSTM. La primera RNN/LSTM que denominamos "*codificador*", lee la frase de entrada y la resume en un vector de contexto. La segunda RNN/LSTM, pasa el resumen (vector de contexto) al "*decodificador*", que traduce la frase de entrada en función de los datos que ha recibido.

Los trabajos realizados por Kyunghyun Cho (Cho et al., 2014) demostraron que las RNN no pueden recordar frases y secuencias demasiado largas debido a los inconvenientes del "*vanishing gradient*" and "*exploding gradient*" (desaparición/explosión del gradiente). Básicamente, este problema consiste en que los gradientes se desvanecerán, tendiendo a 0, o explotarán, haciéndose muy grandes, por el hecho de tener que aplicar el algoritmo de

[3] RNN/LSTM indica que puede ser una RNN o bien una LTSM.

"*backpropagation*" sobre una secuencia larga de elementos, lo que da como resultado que el entrenamiento se vuelva inestable y que la red no entrene correctamente.

La solución a este problema fue el mecanismo de "*Atención*" propuesto por Bahdanau en su estudio (Bahdanau et al., 2015):

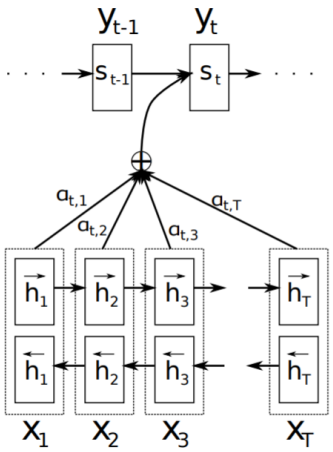

Figura 18. Mecanismo de Atención del paper original. Fuente: (Bahdanau et al., 2015)

En el modelo simple de codificador y decodificador, sólo se utilizaba el último "*hidden state*" de la LSTM codificadora como vector de contexto. Por este motivo, el mecanismo de "*Atención*" incorpora todas las palabras de la entrada (representadas por los "*hidden states*") al crear el vector de contexto y lo hace tomando simplemente la suma ponderada de los "*hidden states*". Es decir, este cálculo que se realiza determina las palabras que son más importantes para el decodificador en ese instante.

De entre todos los mecanismos de "*Atención*" existentes en la actualidad, merecen ser destacados los de Bahdanau y Luong que, aunque los principios subyacentes de la atención son los mismos en ambos, sus diferencias radican principalmente en sus topología y cálculos:

Bahdanau Atención:

Este mecanismo de "*Atención*" también se denomina "*Atención Aditiva*" y es detallado en el estudio (Bahdanau et al., 2015) donde se pretendía mejorar el modelo secuencia-secuencia

(seq-to-seq o seq2seq[4]) en la traducción automática alineando el descodificador con las frases de entrada relevantes e implementando el mecanismo de "*Atención*".

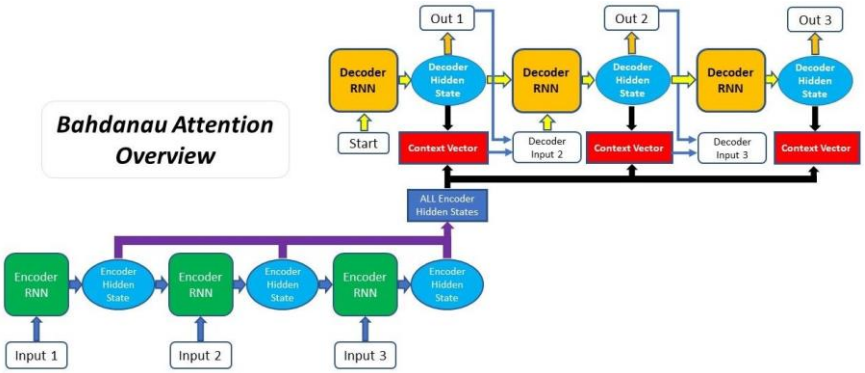

Figura 19. Mecanismo de Atención de Bahdanau modelo seq2seq. Fuente: (FloydHub, 2019)

Luong Atención:

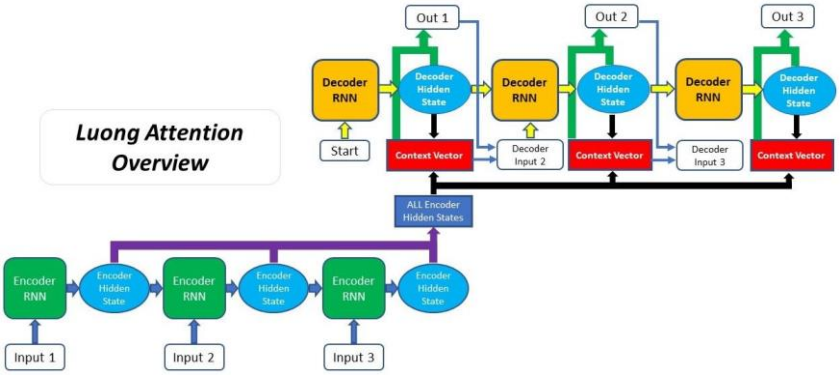

Figura 20. Mecanismo de Atención de Luong modelo seq2seq. Fuente: (FloydHub, 2019)

Este tipo de "*Atención*" fue propuesto por Thang Luong en su trabajo (Luong et al., 2015). Se suele denominar "*Atención Multiplicativa*" y se construyó sobre el mecanismo de Atención propuesto por Bahdanau.

Las dos principales diferencias entre la Atención de Luong y la de Bahdanau son:

- El modo en que se calcula el "*score*".

[4] El aprendizaje secuencia a secuencia (Seq2Seq) consiste en entrenar modelos para convertir secuencias de un dominio (por ejemplo, frases en español) en secuencias de otro dominio (por ejemplo, las mismas frases traducidas al inglés).

- La posición en la que se introduce el mecanismo de atención en el descodificador

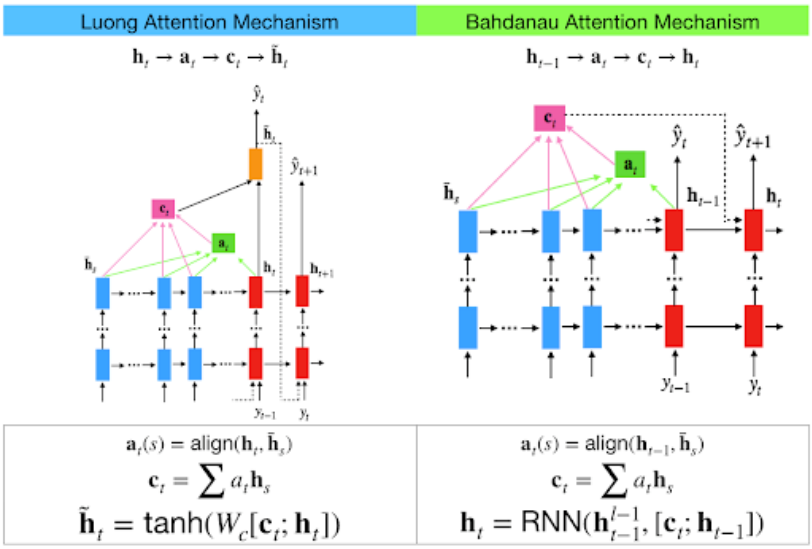

Figura 21. Diferencias entre el mecanismo de Atención de Luong y Bahdanau. Fuente: (Stackoverflow, 2020)

Entre los diferentes estudios en los que se utilizan mecanismos de "*Atención*" para la generación musical, se puede destacar el sistema "*Music Transformer*" (Huang et al., 2019) en el que utilizaron un "*Transformer*" con un mecanismo de "*Atención*" relativa modificado para generar composiciones de un minuto de duración.

Otros estudios interesantes son los realizados por Faqian Guan (Guan et al., 2019) que, partiendo de los trabajos para generar música multipista en forma de piano-rolls utilizando GAN, demostraron que las composiciones creadas siempre eran inestables y que no sonaban lo suficientemente naturales. Por lo tanto, se propuso un nuevo modelo GAN con mecanismo de "*Auto-Atención*", DMB-GAN, que podía extraer más características temporales de la música para generar piezas multi-instrumentales de forma estable y coherente, que sonasen naturales y con buena calidad.

2.4.4 Estrategias de Condicionamiento

Estas estrategias de "*Condicionamiento*", también denominadas "*Arquitecturas Condicionadas*" consisten en incluir información adicional a la entrada de la red neuronal para condicionar el comportamiento del algoritmo y conseguir mejores resultados.

La información de "*Condicionamiento*" que se introduce al algoritmo puede ser:

- Las notas de un bajo o un patrón rítmico.

- La progresión de acordes.
- El género musical
- El instrumento.

Un ejemplo de este tipo de estrategias es "*MidiNet*" (Yang et al., 2017), que como ya se vio anteriormente, se basa en redes GAN e incluye un mecanismo de "*Condicionamiento*" que permite añadir información histórica, tanto de la melodía como de los acordes de los compases anteriores.

"*Wavenet*" (Oord et al., 2016), descrita anteriormente, también hace uso de estrategias de "*Condicionamiento*" al incluirse información relevante del estilo musical y del instrumento que se utiliza para el audio en crudo (raw audio).

2.4 Proyectos en la Actualidad

Señalemos también el creciente interés de diferentes empresas digitales privadas en la generación mediante Inteligencia Artificial de contenidos artísticos, entre las que merecen ser destacadas:

Google:

Crea en 2016 el proyecto de investigación de código abierto denominado "*Magenta*" (Google, 2016) que utiliza diferentes técnicas de aprendizaje automático como herramientas en el proceso creativo. El objetivo fundamental de este proyecto es dotar a los compositores de un conjunto de utilidades o modelos en diferentes lenguajes para ayudarles a crear piezas musicales.

Los principales modelos que proporciona, y que se utilizan para la generación musical, se indican a continuación:

- **Coconet**: Mediante una red neuronal de tipo convolucional permite completar partituras escritas parcialmente.
- **Drums RNN**: Aplica el modelado del lenguaje a la composición de pistas de batería mediante una red neuronal de tipo LSTM.
- **GANSynth**: Es un algoritmo para la síntesis audio mediante el uso de redes de tipo GAN.
- **Melody RNN**: Aplica las técnicas de Deep Learning, usadas en el modelado del lenguaje, a la composición de melodías mediante una red neuronal de tipo LSTM.

- **Improv RNN**: Genera melodías como las del modelo Melody RNN, pero las condiciona a una progresión de acordes subyacente.

- **Music VAE**: Este modelo se basa en un autocodificador variacional recurrente jerárquico que proporciona diferentes modos de creación musical de forma interactiva entre los que se desatacan: el muestreo aleatorio a partir de la distribución a priori, la interpolación entre secuencias existentes y la manipulación de las secuencias existentes.

- **NSynth**: "*Neural Audio Synthesis*", es un modelo de síntesis de audio basado en WaveNet (Engel et al., 2017).

- **Performance RNN**: Aplica el modelado del lenguaje a la música polifónica utilizando una combinación de eventos de activación/desactivación de notas, cambio de tiempo y cambio de velocidad.

- **Polyphony RNN**: Aplica las técnicas de Deep Learning, usadas en el modelado del lenguaje, a la composición de música polifónica utilizando una LSTM. Se basa en la arquitectura de BachBot descrita en "*Automatic Stylistic Composition of Bach Choralies with Deep LSTM*" (Johnson & Shotton, 2017).

- **Onsets and Frames**: Modelo de transcripción automática de música de piano. Este modelo se detalla ampliamente en el estudio de Curtis Hawthorne "*Onsets and Frames: Dual-Objective Piano Transcription*" (Hawthorne et al., 2018).

- **Piano Genie**: Es un modelo que aprende a partir de una representación discreta de baja dimensionalidad de la música para piano. Se basa en una RNN codificadora que comprime las secuencias de notas de piano (88 notas o teclas) en muchas menos teclas (por ejemplo, 8). Posteriormente, otra RNN decodificadora se encarga de volver a convertir las secuencias de notas en el intervalo original de notas de piano (88 notas o teclas).

- **Pianoroll RNN-NADE:** Aplica las técnicas de Deep Learning, usadas en el modelado del lenguaje, a la composición de música polifónica utilizando una red de tipo LSTM combinada con un NADE (neural autoregressive distribution estimator). El resultado es una arquitectura denominada RNN-NADE. Este modelo se basa en la arquitectura descrita en "*Modeling Temporal Dependencies in High-Dimensional Sequences*" (Boulanger-Lewandowski et al., 2012).

- **RL Turner:** Este modelo se basa en una red neuronal de tipo LSTM que se entrena para predecir la siguiente nota en una melodía monofónica y la mejora utilizando el RL. Este modelo ha sido descrito por Natasha Jaques en "*Sequence Tutor*" (Jaques et al., 2017).

- **Score2Perf:** Es una colección de problemas Tensor2Tensor para generar interpretaciones musicales, ya sean condicionadas o no a una partitura.

Con el uso de los modelos de Google Magenta multitud de autores han conseguido llevar a cabo sus proyectos de composición musical. Destacamos algunos de ellos:

- MJ Jacob. Rapero/productor/ingeniero de Hip-Hop, se hace llamar MJx Music. Utiliza los modelos proporcionados por Google Magenta para realizar composiciones con millones de streams en Spotify y Apple Music (MJx Music, 2021).

- Aparna Kumar. Ha creado un proyecto donde se muestra de forma muy divertida los pasos de Bharatantyam (tipo de danza clásica india que se sigue practicando en la actualidad). La música se genera con Inteligencia Artificial, en base a los modelos de Google Magenta, y una bailarina ejecuta la danza sobre la composición en tiempo real (Natya*ML, 2021).

Spotify:

En 2017 se crea el "*Creator Technology Research Lab (CTRL)*" (Spotify, n.d.) que se centrará en la investigación tecnológica y en la creación de software para ayudar a los artistas hacer realidad su visión creativa.

Sony:

El Sony Computer Sciences Laboratory (CSL) (Sony CSL, 2016) creó el programa de investigación denominado "*Flow Machines*" que utiliza algoritmos de Inteligencia Artificial para la creación de composiciones musicales. Este software analizó miles de canciones para obtener información sobre el tipo de música, estilo, acordes y armonía. En base al entrenamiento aprendido pudo crear un álbum de 15 canciones de música electrónica de los años 70. Utilizando como base esta Inteligencia Artificial, se desarrollaron un conjunto de algoritmos que pueden crear diferentes composiciones adaptadas a las preferencias del compositor. La base de este sistema radica en el análisis de las características de las melodías y las propiedades estadísticas relacionadas con la armonía. De este modo, el sistema aprende a generar nuevas notas qué estén asociadas a un determinado acorde, o bien a generar nuevas notas que suelen ir después de una nota determinada.

AIVA (Artificial Intelligence Virtual Artist):

Esta empresa emergente fue creada en 2016 por los hermanos Vicent y Pierre Arnaud. Su proyecto se centra en la composición de música clásica sinfónica con algoritmos de Inteligencia Artificial, principalmente Deep Learning y RL, a partir de datasets de entrenamiento de Mozart, Bach, Beethoven, etc. (Barreau & Barreau, 2016).

En 2019 esta empresa ha sacado al mercado como producto comercial el sistema "*Music Engine*" que compone piezas cortas en diferentes estilos (pop, rock, etc.).

Amper Music:

Los productos de esta compañía permiten la generación de música mediante la fusión de la teoría musical y algoritmos de Inteligencia Artificial. Es capaz de crear música para acompañar podcast, vídeo clips, video juegos y otros tipos de contenidos en cuestión de segundos.

En la actualidad Amper Music oferta 2 productos comerciales:

- La plataforma de creación musical "*Score*"
- El API que permite a las compañías integrar las capacidades de composición musical de la Inteligencia Artificial de Amper en sus propios Sistemas.

Melodrive:

Esta empresa emergente Berlinesa creada en 2016 ha desarrollado el primer sistema basado en Inteligencia Artificial, que compone en tiempo real y desde cero, un flujo infinito de música original y emocionalmente variable. La música que se genera se adapta de forma continua al escenario y a las interacciones del usuario con su entorno, de ahí que su público objetivo sean los gamers, desarrolladores de video juegos y los streamers.

Esta empresa ha lanzado comercialmente su primer producto "*Melodrive Indie*" como un SDK que puede integrarse con el motor de video juegos o como una aplicación que proporciona música sobre los juegos y que se denomina "*Melodrive Play*".

OpenAI:

No podía faltar OpenAI entre las principales compañías que crean productos basados en Inteligencia Artificial capaces de generar música de forma automática. Entre sus creaciones se pueden destacar:

- Jukebox: Es un sistema capaz de generar piezas musicales incluyendo canto en el dominio del audio en crudo (raw data). Este sistema utiliza un VQ-VAE multi-escala que es comprimido en códigos discretos y modelado mediante Transformadores Autorregresivos. De este modo, se consiguen crear canciones en alta fidelidad de forma coherente de varios minutos de duración y en varios estilos musicales. (Dhariwal et al., 2020)
- MuseNet: Este algoritmo está basado en una red neuronal profunda que puede crear piezas musicales de 4 minutos con 10 instrumentos diferentes, y puede combinar estilos diferentes que van desde el pop hasta Bach, pasando por los Beatles.

Finalmente, hay numerosos artistas y empresas que han utilizado la Inteligencia Artificial para crear sus propias composiciones y lanzarlas al mercado discográfico:

- Benoit Carré:
    - Daddy's Car. Utilizando la Inteligencia artificial Flow Machines de Sony CSL se compuso al estilo de los Beatles. 2017.
    - Mr. Shadow. Utilizando la Inteligencia artificial Flow Machines de Sony CSL se compuso al estilo de los cantantes americanos Irving Berlín y Duke Ellington. 2017.
- Taryn Southern. Álbum I AM AI. 2018:
    - Break Free. Utilizando la Inteligencia artificial de Amper Music
    - Life Support. Utilizando la Inteligencia artificial de Amper Music
- Botnik Studios. Álbum The Songularity. 2020:
    - You can't take my door. Cantada por Elle O'Brien and Timothy Joyce. Se utilizó una red neuronal recursiva que se entrenó con música Country.
- Skygge. Utilizando la Inteligencia Artificial de Spotify CTRL. Álbum Hello World. 2018:
    - Ballad of the Shadow.
    - Sensitive. Cantada por C Duncan.
    - One Note Samba. Cantada por Pirouttes.
    - Magic man.
    - In the House of the Poetry. Cantada por Kyrie Kristmanson.
    - Hello Shadow. Cantada por Kiesza.
    - Mafia Love.
    - Paper Skin. Cantada por Jata.
    - Multi Mega Fortune. Cantada por Michael Lovett.
    - Valise. Cantada por Pirouttes.
    - Cake Woman. Cantada por Mederic Collignon y Camille Bertault.
    - Whitle Theme. Cantada por el grupo Catastrophe.
    - Je vais te manger. Cantada por Laurent Bardainne.
    - Cold Song.
- Skygge. Utilizando la Inteligencia Artificial de Spotify CTRL. Álbum American Folk Songs. 2019:

  - Black is the color. Cantada por Pete Seager.
  - Amazing Grace. Cantada por Kyrie Kristmanson.
  - Oh Shenandoah. Cantada por Pete Seager y Stephane Kerecki.
  - Song of Myself. Cantada por Peggy Seeger.
  - Wondrous Love. Cantada por Horton Barker.
- AIVA. Álbum Genesis. 2016. De las 23 piezas destacamos:
  - Symphonic Fantasy in A minor, Op. 21: Genesis.
  - Symphonic Fantasy for Orchestra in G-sharp Minor, Op. 7: The awaking.
  - Octet Nº 1 in D major, Op. 3: A little chamber music.
  - Solos para piano (Op. 1, Op. 2, Op. 3, Op 4, etc.)
- AIVA. Álbum Among the Stars. 2018. De las 24 composiciones destacamos:
  - Sinfonías fantásticas (B minor: Imperial, B minor Op. 37: The Battle from Andromeda, C-sharp minor Op. 39: Purple, F-sharp minor Op. 17: In Vogue, C major Op. 42: The age of Amazement, etc.
  - Solos para piano (Op. 24, Op. 31, Op. 32, Op 33, etc.)

Proyectos en música flamenca:

Desde el punto de vista de la generación artística específica para la música de estilo flamenco no existen apenas estudios o iniciativas en el área de la Inteligencia Artificial y, aún menos, herramientas específicas que ayuden a los compositores flamencos a realizar sus propias creaciones. Únicamente cabe destacar el desarrollo de un sistema informático realizado por la Universidad de Sevilla que permite identificar los diferentes palos (estilos flamencos) del cante jondo (Mora Merchán, 2018).

2.5 Conclusiones del Contexto y del Estado del Arte

Se ha revisado la composición algorítmica, la composición musical por ordenador y se ha realizado un amplio recorrido por las principales arquitecturas de Deep Learning que se utilizan para la generación musical, viéndose ejemplos de casos de uso de todas ellas. Se ha puesto especial énfasis en las topologías LSTM que por sus especiales características de

"*memoria*" son ideales para generar contenidos musicales. Adicionalmente, se han visto los mecanismos de "R*efinement*" y "*Condicionamiento*" más utilizados, entre el que se destaca el mecanismo de "*Atención*" que permite a las redes neuronales fijarse en aquellos contenidos más interesante para la generación de secuencias musicales.

Finalmente, se han indicado las principales compañías digitales privadas que generan contenidos artísticos mediante Inteligencia Artificial (Google, Spotify, Sony, etc.), así como diferentes artistas que han utilizado los productos de estas empresas para crear sus propias composiciones y lanzarlas al mercado discográfico.

3. Objetivos y Metodología de Trabajo

3.1. Objetivo General

El objetivo general de este trabajo es la realización de un conjunto de experimentos con distintas arquitecturas de Deep Learning con el fin de explorar diferentes técnicas de composición de música flamenca para guitarra del estilo de la Soleá.

Los experimentos utilizarán un conjunto de datos de entrenamiento consistentes en partituras y tablaturas[5] para guitarra flamenca del palo de la Soleá.

Los algoritmos utilizados y entrenados deberán generar composiciones musicales para este estilo flamenco manteniendo el compás y tonalidad propias de la Soleá.

3.2. Objetivos Específicos

Para el desarrollo de este trabajo se establecen los siguientes objetivos específicos:

1. Añadir información adicional de entrada sobre el compás[6] al algoritmo para aportar más riqueza al entrenamiento y generar música con la rítmica propia del estilo de la Soleá.

   Esto implica procesar las partituras MIDI que conforman el dataset de entrada, con el fin de generar la información de compás que se introducirá al algoritmo junto con el valor de las notas y su correspondiente duración.

   En la actualidad, existen sistemas que son capaces de generar composiciones musicales partiendo de partituras con las que se entrenaron. Estos sistemas incorporan información de las notas (valor, duración, intensidad, etc.) y, en algunos casos, información de "C*ondicionamiento*" con los acordes utilizados, el estilo musical, notas que componen la voz del bajo, etc. De este modo, al añadir información adicional de entrada, se aporta más riqueza al entrenamiento.

2. Crear mecanismos específicos de temperatura más eficaces que permitan la generación de música con la rítmica y tonalidad flamenca del estilo de la Soleá.

[5] Tipo de notación musical uy utilizada en instrumentos de cuerda (guitarra clásica, guitarra, eléctrica, bandurria, laúd, etc.) en la que se indica la cuerda y el traste donde se debe tocar cada nota de una melodía.

[6] El compás flamenco es un elemento técnico característico de la música flamenca. Es el pulso que permite seguirla y que hace que nos "*enganchemos*" a la rítmica flamenca. El compás consta de un conjunto de pulsos que conforman un determinado patrón. En ese patrón rítmico destacan algunos de ellos porque se les da más énfasis o intensidad: es el acento. Cada patrón y sus correspondientes acentos se asocian un palo flamenco diferente. Y es que el compás es la base sobre la que los artistas se mueven en el escenario, los cantaores cantan y los tocaores ejecutan su toque.

Actualmente, en los procesos de generación musical, se utilizan los mecanismos denominados de "*temperatura*" que permite añadir aleatoriedad y variabilidad a la composición musical que se genera. Estos mecanismos sólo tienen en cuenta la probabilidad de que se genere una nota en un determinado momento y carecen de otro tipo de información externa o complementaria que les permitan generar composiciones más realistas. Por lo tanto, lo que se pretende, es crear mecanismos de "*temperatura específicos*", más eficaces, que permitan una generación musical que se adapte mejor a la rítmica y tonalidad flamenca del estilo de la Soleá.

3. Analizar las diferentes estructuras de codificación de los datos que mejor se adapten para el entrenamiento. Es necesario explorar las diferentes posibilidades que van desde la codificación One-Hot utilizando un único vector de entrada con la información musical del estilo flamenco o múltiples vectores de entrada que aporten más información al algoritmo.

4. Explorar los diferentes algoritmos de Deep Learning (RNN, LSTM, LSTM bidireccionales, etc.) y sus diferentes implementaciones (Auto-Encoders, Atención, etc.) para conseguir generar música flamenca de Soleá que se ajuste al compás flamenco.

5. Utilizar el framework de Google Magenta para la generación de música flamenca del estilo de la Soleá.

   Esta biblioteca open source incluye utilidades para la manipulación de datos y para el entrenamiento de modelos de aprendizaje automático que pueden utilizarse para generar contenidos artísticos.

   La biblioteca de Google Magenta se utiliza en algunos experimentos, mientras que, en otros, se utilizan arquitecturas desarrolladas especialmente para componer música flamenca del estilo de la Soleá. En estos últimos, no incluye ninguna llamada a este framework.

6. Crear un dataset de partituras en formato MIDI a partir de partituras y tablaturas de guitarra flamenca del estilo de la Soleá.

   Para ello, las diferentes piezas son pre-procesadas para eliminar las partes rítmicas. Seguidamente, son transformadas en monofónicas, antes de obtener el dataset final que sirve de entrada para todos los experimentos que se realizan. Los procesos de depuración, filtrado y procesamiento se detallan a lo largo de este estudio en puntos posteriores.

3.3. Metodología del Trabajo

En este apartado se describen los pasos que se han dado y las herramientas utilizadas para conseguir los objetivos indicados anteriormente:

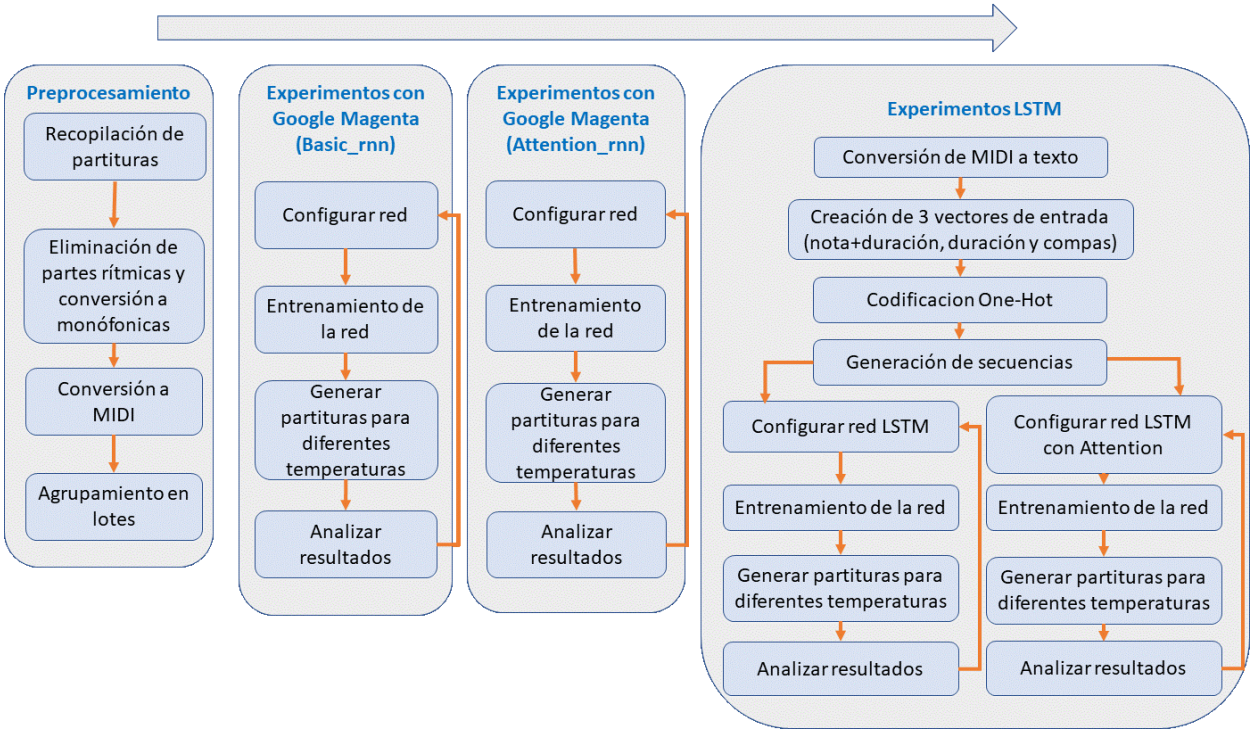

Figura 22. Resumen de la metodología a seguir. Fuente: Elaboración propia.

1. Se realiza una búsqueda de las partituras flamencas del estilo de la Soleá, con el fin de crear un dataset suficientemente grande que sirva de entrada a los diferentes algoritmos que se van a utilizar en este trabajo para generar la música flamenca.

   Todas las partituras y tablaturas recopiladas corresponden a piezas de Soleá para guitarra flamenca, también conocida como guitarra española.

   El origen de estas partituras ha sido:

   - Internet: Algunas de estas partituras han sido recopiladas de las páginas web (Perso flamenco & classic, 2021), (Guitar Pro Tabs, 2021), (TabsFlamenco.com, 2021), etc. El formato en el que se ha recopilado estas partituras ha sido:
     - Guitar-pro (.gtp y .gtpx).
     - TablEdit (.tef).
     - MIDI (.mid).
     - PowerTab (.ptb).

       Estos formatos pueden leerse desde el programa Tuxguitar.

   - Transcripciones en papel pautado: Otras partituras no se encontraban en internet y han tenido que transcribirse de libros de flamenco a formato digital mediante el uso de

los programas Tuxguitar y MuseScore. El formato en el que finalmente se encuentran estas partituras es el propio de Tuxguitar (.tg)

- Partituras del autor de este trabajo: Recopilaciones en formato digital y papel pautado procedentes de clases de guitarra flamenca con diferentes profesores a lo largo de los años. El programa utilizado para la creación de estas partituras ha sido Tuxguitar y el formato utilizado ha sido el propio de este software (.tg)

  Todas las partituras son para guitarra flamenca y por lo tanto son polifónicas, esto quiere decir que varias notas pueden tocarse simultáneamente, hasta 6 notas, ya que la guitarra española tiene 6 cuerdas.

  Las partituras están transcritas para guitarra flamenca, esto implica que las melodías están adaptadas a la digitación y a las particularidades propias de este instrumento.

2. Todas las partes que componen las Soleares y que se encuentran escritas en las partituras que forman el dataset de entrada se han respetado, a excepción de las partes rítmicas que han sido eliminadas de todas ellas por no aportar riqueza melódica (siempre son las mismas variaciones) y por ser polifónicas.

   Los compases con polifonía (varias notas tocadas simultáneamente) se han transformado en monofónicos intentando no alterar la estructura melódica de la música original:

   - En el caso de acordes (3 o más notas tocadas simultáneamente) se dejaba la nota tónica del acorde eliminando el resto.

   - Los bicordios (2 notas tocadas simultáneamente) han sido sustituidos por la nota que armónicamente sonaba mejor y que mantenía el carácter flamenco del estilo de la Soleá.

   - Transformación de falsetas con notas más rápidas que las semicorcheas. Estas notas utilizadas en adornos y pasajes virtuosos han sido cambiadas por otras de mayor duración intentando mantener la esencia de la melodía y el compás característico del estilo flamenco de la Soleá.

   - Las partituras que estaban en diferente tonalidad han sido transportadas al tono característico de la Soleá (MI, FA, SOL Lam) o como los entendidos lo conocen "*por arriba*".

   - Las falsetas que estaban repetidas, en las diferentes partituras de Soleá, se han eliminado dejando sólo una de ellas.

   - Algunas de las falsetas del dataset de entrada se han ajustado al tiempo que deberían tener (12 tiempos). Esto se ha realizado puntualmente ya que la mayor parte de las piezas de Soleá se ajustaban al compás de 12 tiempos.

3. Conversión de las partituras a formato MIDI para que puedan utilizarse como entrada a los diferentes experimentos que se van a desarrollar. Esta transformación se realiza con las funcionalidades proporcionadas por los programas Tuxguitar y MuseScore.

4. Dependiendo del tipo de experimento que se vaya a realizar, las partituras MIDI se podrán agrupar en lotes con el fin de que las piezas musicales sean más extensas. Esta unión de piezas se realiza, con el fin de pasar las validaciones que aplica Google Magenta cuando analiza la longitud mínima que debe tener las partituras y el número de notas mínimo que debe haber en cada una de ellas.

5. La información procesada y filtrada ya se puede utilizar como entrada a los diferentes experimentos que se van a realizar, con el fin de generar música flamenca del estilo de la Soleá.

   Se realizarán 2 tipos de experimentos diferentes. Los basados en el uso de Google Magenta y los basados en la creación de redes LSTM específicas para la generación de música flamenca del estilo de la Soleá:

   - **Experimentos con Google Magenta:**

     - Se partirá del dataset de partituras flamencas de Soleá en formato MIDI.

     - Los experimentos que utilicen el framework de Google Magenta no necesitarán realizar ningún procesamiento adicional de los datos de entrada como pudieran ser la división en secuencias o la codificación One-Hot (se realizarán internamente por Google Magenta).

     - El modelo "*Melody_rnn*" que proporciona Google Magenta será el que se utilice para estos experimentos:

       - Se probarán las diferentes configuraciones proporcionadas por el modelo "*Melody_rnn*". En todos los casos, el número de capas, neuronas por cada capa y tamaño del batch serán las que incorpora cada configuración por defecto.

       - Se realizará el entrenamiento de la red neuronal para las diferentes configuraciones.

       - Se generarán varias partituras con la red entrenada para diferentes configuraciones y para diferentes valores de temperatura.

       - Se analizarán los resultados obtenidos para cada configuración y para diferentes valores de temperatura, en base a los criterios de evaluación descritos en el apartado de evaluación de resultados.

- **Experimentos con redes LSTM específicas:**

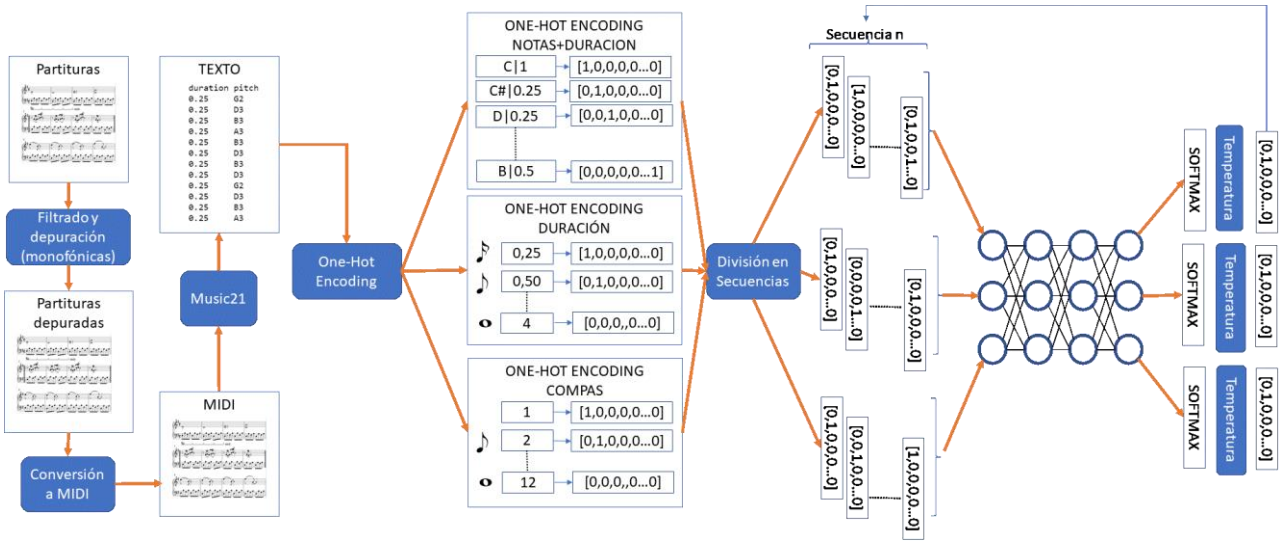

Figura 23. Flujo de transformación de datos. Fuente: Elaboración propia.

- Al igual que los experimentos realizados con el framework de Google Magenta, se partirá del dataset de partituras flamencas de Soleá en formato MIDI; pero en este caso, la información de entrada deberá procesarse para transformarla en formato texto, ya que la red LSTM no es capaz de computar eficientemente valores en formato MIDI. El proceso de conversión se realizará mediante un proceso Python y utilizando las funcionalidades de la librería Music21.

- A continuación, la información textual (nombre nota y duración) se procesará para construir 3 vectores:

  - Vector de nota+duración: Se incluirá en el mismo vector la nota y la duración concatenando ambos con un carácter especial (por ejemplo "|").

  - Vector de duración: Se incluirá la duración de cada nota en cada componente del vector.

  - Vector de compás: En base a la duración de cada nota, se desarrollará un proceso que informe para cada nota y duración, a qué compás pertenece.

- Los vectores se codificarán con One-Hot, para que la información sea procesable de forma más eficiente por el algoritmo.

- La información de los vectores codificados se dividirá en secuencias de n notas consecutivas, que serán la entrada a la red. La nota n+1, que también se representará como 3 vectores, será la salida de la red.

- Se probarán diferentes topologías (LSTM, LSTM bidireccionales, mecanismos de "*Atención*", etc.). Para cada una de ellas:
  - Se realizará el entrenamiento de la red para cada una de las arquitecturas.
  - Se generarán, para cada una de las arquitecturas, varias partituras con la red entrenada para diferentes valores de temperatura. El número de notas a generar en cada caso se indicará cuando se realice el experimento.
  - Se analizarán los resultados obtenidos para cada una de las arquitecturas y para cada uno de los valores de temperatura utilizados, en base a los criterios de evaluación descritos en el apartado de evaluación de resultados.

Evaluación de Resultados

Las melodías generadas del estilo de la Soleá son escuchadas por 3 personas diferentes (evaluadores) cuyos datos se indican en la siguiente tabla:

Tabla 1. Evaluadores de la música flamenca generada en los experimentos

Tipo Evaluador	Descripción	Conocimientos	Observaciones
Profesional	Profesor de guitarra flamenca en Madrid y profesor particular del autor de este trabajo.	Muy altos	Conoce profundamente todos los ritmos flamencos y sabe acompañar al cante al baile y da conciertos como solista.
Avanzado	Aprendiz de guitarra flamenca y compañero de clases de guitarra del autor de este trabajo. Recibe clases del mismo profesor que el autor de este trabajo.	Altos	Conoce y toca en la guitarra los principales estilos. El flamenco es una afición.
Aficionado	Persona que escucha música flamenca y que le gusta el flamenco. Amigo del autor de este trabajo.	Bajos	Conoce los principales palos flamencos y diferencia los estilos.

Estas personas evalúan cuantas falsetas (frase melódica compuesta por varios compases) de las varias que se generan en cada partitura, suenan dentro del estilo de la Soleá.

Para considerar que una falseta suena al estilo de la Soleá se deben cumplir 3 condiciones:

1. Dada una falseta, se considera que ésta va a compás cuando, empezando la melodía en el compás 1, el cierre o conclusión se realiza en el compás 12 o en un múltiplo de 12. Cada partitura que se genera contiene un número variable de falsetas y sólo algunas de ellas siguen el compás propio de la Soleá.

2. Además de seguir el compás, una falseta tiene que seguir la tonalidad y armonía flamencas de la Soleá, sin disonancias.

3. Si durante la audición de una partitura, se detectara que la música entra en un bucle, los evaluadores considerarán que las falsetas (dentro del bucle) no cumplen los criterios y que no suenan dentro del estilo de la Soleá. Este "*embuclamiento*" es más probable que se produzca cuando el valor de la temperatura de generación es 0 o muy próximo a él.

   En base a estos 3 criterios, se indica, de forma tabulada para cada uno de los experimentos, el número de falsetas que cada evaluador considera que suenan dentro del estilo flamenco de la Soleá.

   Las columnas que componen la tabla de evaluación de resultados son:

   - Temperatura: Valor del mecanismo de temperatura para la generación de partituras de Soleares.
   - Partituras de 500 notas: Total de partituras de 500 notas musicales generadas para cada temperatura.
   - Falsetas Generadas: Número total de falsetas que se han generado para un valor de temperatura. El cálculo se realiza sumando todas las falsetas de las partituras generadas para esa temperatura. Dado que cada partitura contiene notas de diferente duración, el número de falsetas que se generan es diferente (cuanto mayor sea la duración de las notas menos falsetas se generan). Adicionalmente, las falsetas no tienen por qué durar 12 compases exactamente, pudiendo durar 24, 36, etc., pero siempre un múltiplo de 12, lo cual repercute en el menor número de falsetas generadas.
   - Evaluador Profesional: Total de falsetas que suenan dentro del estilo de la Soleá, según el evaluador profesional, para todas las partituras generadas con esa temperatura.
   - Evaluador Avanzado: Total de falsetas que suenan dentro del estilo de la Soleá, según el evaluador avanzado, para todas las partituras generadas con esa temperatura.
   - Evaluador Aficionado: Total de falsetas que suenan dentro del estilo de la Soleá, según el evaluador avanzado, para todas las partituras generadas con esa temperatura.

   En la parte final de la tabla de evaluación, se incluyen los valores totales por columnas y los totales expresados en porcentajes, así como el porcentaje medio final cuyo cálculo corresponde con la media aritmética de los porcentajes de los diferentes evaluadores. Este último valor (porcentaje medio final) es el que se utilizará para poder comparar los resultados de cada uno de los experimentos. Un valor de 100% significa

que todas las falsetas generadas suenan dentro del estilo flamenco de la Soleá mientras que un valor de 0% indica todo lo contrario.

Herramientas utilizadas.

Se han utilizado las siguientes herramientas para el tratamiento de las partituras flamencas del dataset de entrada de Soleares:

- **Tuxguitar 1.5.4**: Herramienta de software desarrollada en Java-SWT que permite tanto la reproducción como la edición de partituras tablaturas. Adicionalmente, soporta la importación/exportación de los formatos propios de utilidades similares como GuitarPro (.GP3, .GP4 y .GP5). Funcionalmente este programa incluye características avanzadas como son la edición en múltiples pistas, editor de tablaturas, edición de efectos de sonido sonoros, etc. (Tuxguitar Team, 2020). Este producto es gratuito, de código abierto y mantenido por la comunidad de desarrolladores.

- **MuseScore 3**: Esta herramienta es software de edición musical que permite la reproducción e impresión de partituras. Existen versiones de esta utilidad para diferentes sistemas operativos y soporta multilenguaje. Su interfaz de usuario está basada en la filosofía WYSIWYG (lo que tienes es lo que ves). En cuanto a sus funcionalidades destacar que admite hasta 4 voces por pentagrama y un número ilimitado de pentagramas, importación/exportación a múltiples formatos (MIDI, PDF, WAV, MusicXML, etc.), inclusión de notación standard de acordes, lyrics, etc. (Musescore Org., 2018)

- **Jupyter Notebooks** (Anaconda): En estos cuadernos se desarrolla el código en Python versión 3.8 que permite la realización de los diferentes experimentos. Las principales librerías que se utilizan son:

    - Music21: Lectura y transformación de datos en formato MIDI a otros formatos. Análogamente, el proceso de transformación inverso y almacenamiento en formato MIDI de las partituras generadas también se realiza con esta librería.

    - Tensorflow 2.0 y Keras 2.4: Definición de la arquitectura de las redes neuronales, entrenamiento y generación de música flamenca del estilo de las Soleá.

Frameworks Utilizados

- **Google Magenta:** (Google, 2016) es un trabajo de investigación de código abierto que implementa librerías con diversos algoritmos de aprendizaje automático que pueden ser utilizados en la generación de contenidos musicales y artísticos. Fue iniciado por ingenieros del equipo de Google Brain, pero muchos otros han contribuido significativamente al proyecto. Magenta se distribuye como una biblioteca Python de código abierto (Github), impulsada por TensorFlow y que incluye utilidades para manipular datos de origen (principalmente música e imágenes), utilizar estos datos para entrenar modelos de aprendizaje automático y, finalmente, generar nuevos contenidos a partir de estos modelos.

Magenta proporciona multitud de modelos que pueden utilizarse para la generación de contenidos artísticos (música e imagen). Estos modelos pueden entrenarse utilizando los dataset que se deseen para que el modelo genere los contenidos en base a la información de entrada. Cabe destacar que muchos modelos ya vienen pre-entrenados y simplemente es necesario seleccionar alguno de los "*presets*" disponibles que mejor se ajuste a nuestras necesidades y empezar con la generación de contenidos artísticos.

Los diferentes modelos proporcionados por Magenta se encuentran disponibles en Github: (Google Developers, 2020). De todos los existentes, descritos en el estado del arte, utilizaremos en estos experimentos el modelo "*Melody RNN*" que se basa en una arquitectura de red neuronal del tipo LSTM.

Consideraciones del Dataset de entrada

El dataset de entrada está formado por 41 partituras en formato MIDI. Cada una de ellas contiene a su vez un número medio de falsetas entre 5 y 20.

La estructura melódica de la Soleá es muy variada pero siempre consta de las siguientes frases:

- Intro: Conjunto de compases que se ejecutan al inicio de la pieza y tienen una melodía característica.
- Llamada: Compases que indican al "*cantaor*" que puede, una vez finalizados, empezar a entonar las coplas.
- Falseta: Melodía que se intercala entre los diferentes ritmos y entre las diferentes coplas que entona el cantaor.
- Escobilla o ritmo: Secuencia rítmica compuesta de rasgueos que se intercala entre las diferentes partes de la Soleá y caracteriza a cada palo flamenco.
- Cierre remate: Conjunto de compases que cierran una interpretación o una falseta.

Estas frases rítmicas son conocidas por el "*cantaor*" y el "*tocaor*", de este modo, se establece una comunicación entre ambos que permite que se ejecute un cante por Soleá sin haber sido ensayado previamente.

El estilo flamenco de la Soleá tiene la particularidad de que consta de 12 tiempos que en partitura se van a ajustar a 4 compase de ¾ o a un múltiplo de 4. Esto implica que todo está en torno a estos 12 tiempos y no puede haber, ni uno más, ni uno menos, a lo largo de toda la pieza musical, porque esto haría que los diferentes integrantes como palmeros, bailaores y cantaores se fueran de tiempo y se perdieran durante la ejecución.

La Soleá es un palo que, además de los 12 tiempos, tiene unas acentuaciones muy particulares que se realizan en el 3, 6, 8, 10 y 12 (cada palo flamenco tiene unos acentos característicos que permiten diferenciarlo del resto).

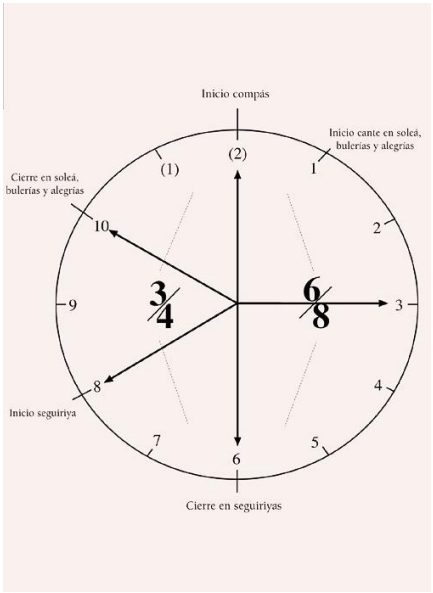

Figura 24. Acentuaciones del palo de la Soleá. Fuente: Elaboración propia

Estos acentos hacen especial a la Soleá y no se pueden representar en las partituras de forma fehaciente, en lo que a ritmo se refiere. Como ya se indicó, por comodidad y homogeneidad, se va a utilizar un compás de ¾, lo que se traduce en que se van a acentuar la primera parte de cada compás, es decir el 1, 4, 7 y 10. Esto en principio no tiene mayor problema de cara al entrenamiento en una red neuronal, pero es necesario indicar este tipo de particularidades si se desea tocar la pieza en guitarra leyendo la partitura de forma correcta, con el ritmo y acentuación adecuados.

4. Descripción Detallada de los Experimentos

En los apartados anteriores, se ha intentado dar una visión general del estado del arte en lo referente a la generación de contenidos musicales, repasando las principales topologías que se utilizan en el ámbito del Deep Learning. También, se han revisado mecanismos de "*Refinement*" y "*Condicionamiento*", como el de "*Atención*", que permiten que estas arquitecturas sean aún más eficientes.

El objetivo principal de este trabajo es explorar diferentes arquitecturas de Deep Learning con el fin de componer música flamenca para guitarra del estilo de la Soleá, que melódicamente sea agradable y que carezca de disonancias. Por este motivo, en esta segunda parte del estudio, se realizan de forma detallada los diferentes experimentos exploratorios utilizando algunas de las arquitecturas anteriormente descritas en el estado del arte.

En los siguientes puntos, y para cada uno de los experimentos, se indica: el detalle del experimento, la entrada de datos, las características de la red neuronal que se utiliza, el entrenamiento realizado (gráficas de accuracy y loss) y la generación de las piezas flamencas de Soleá (secuencias iniciales de cebado y ejemplos de composición)

4.1. Experimento 1. Magenta Melody_rnn – Basic_rnn

Descripción del experimento

En este experimento se aborda la composición de música flamenca del estilo de la Soleá utilizando las librerías creadas por Google Magenta y descritas brevemente en el punto anterior.

De los diferentes modelos que aporta este framework, se utiliza el modelo "*Melody_rnn*" en su modalidad de configuración básica "*Basic_rnn*".

Este modelo tiene la particularidad de que únicamente admite música monofónica por lo que el dataset de entrada sólo puede contener partituras MIDI, donde nunca se toquen simultáneamente 2 notas o más. Además, tampoco es posible el uso de notas más rápidas que las semicorcheas ya que las considera polifonía (aunque realmente no lo sea). Cualquier partitura del dataset de Soleares que no cumpla con estas restricciones, se descarta automáticamente por Google Magenta durante los procesos previos de validación que se realizan antes de empezar el entrenamiento.

Entradas

El dataset de entrada está formado por las partituras de Soleares en formato MIDI y con las transformaciones descriptas anteriormente implementadas. Este conjunto de entrenamiento es el mismo para todos los experimentos que se realizan en este trabajo.

Hay que destacar que, en esta configuración *"Basic_rnn"*, todas las partituras utilizadas en el entrenamiento son traspuestas por Google Magenta automáticamente al rango de los tonos MIDI en el intervalo [48, 84], al igual que las salidas que también están en este rango.

Estructura de la red neuronal:

En este experimento se utiliza el modelo "*Melody_rnn*" y la configuración "*Basic_rnn*". La configuración interna es una arquitectura de red neuronal de tipo LSTM que utiliza una codificación One-Hot como representación de las melodías de entrada.

Los parámetros de entrada para este modelo son 2 capas LSTM de 64 neuronas cada una. Se utiliza esta topología de red neuronal, porque es la que viene implementada por defecto para esta configuración y porque suele dar buenos resultados.

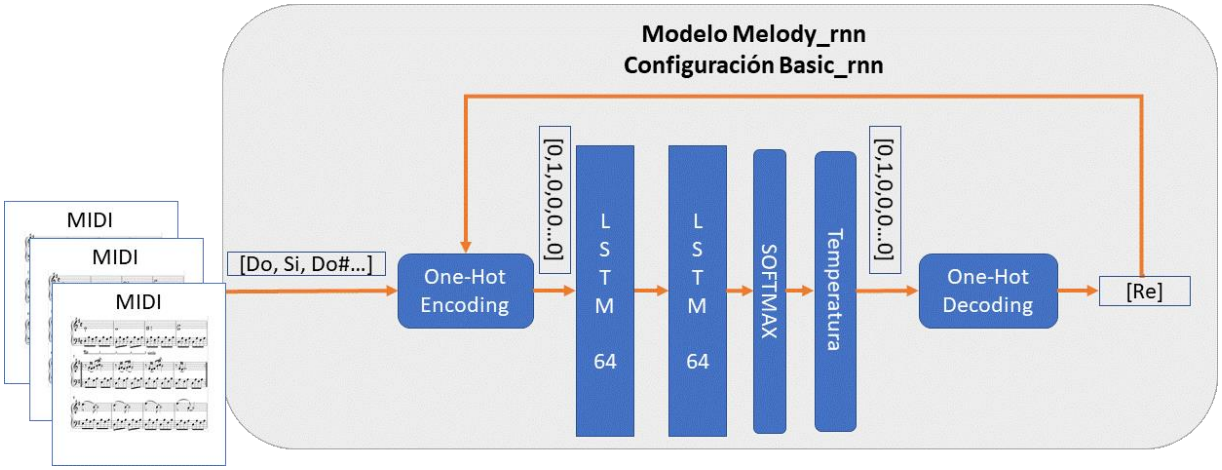

Figura 25. Configuración Basic_rnn del modelo Melody_rnn de Google Magenta. Fuente: Elaboración propia

Entrenamiento de la red:

Los parámetros de entrenamiento utilizados para esta configuración son:

- Tamaño batch: 64
- Epoch: 21.000

La red se entrenada hasta conseguir:

1. Training Accuracy = 84,4%

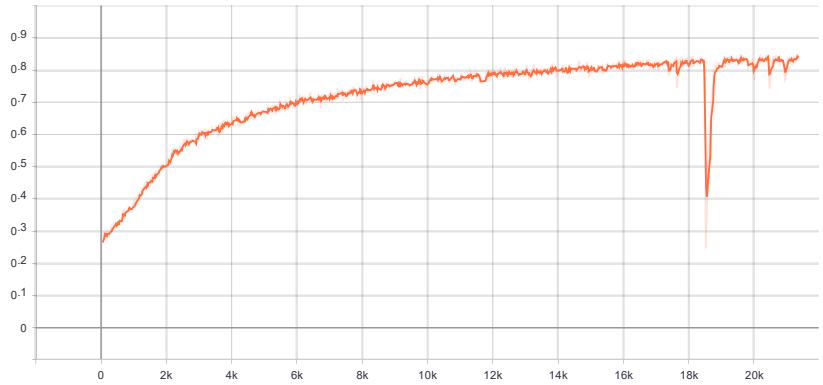

Figura 26. Accuracy del modelo. Configuración Basic_rnn. Fuente: Elaboración propia

2. Loss per steps = 0,4419

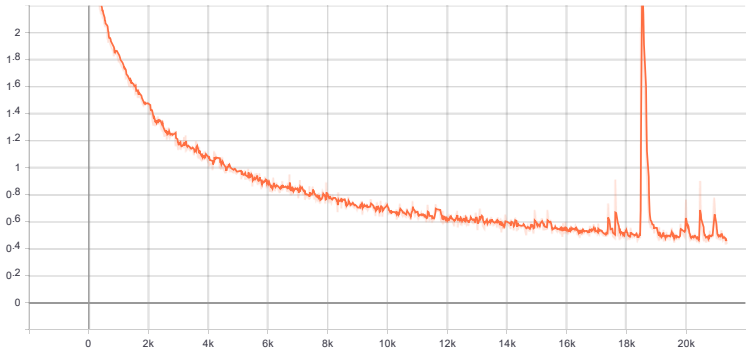

Figura 27. Loss per steps del modelo. Configuración Basic_rnn. Fuente: Elaboración propia

Generación:

Una vez realizado el entrenamiento se procede a la generación de música flamenca:

- Temperatura: Se utilizan 4 valores de temperatura diferentes:
    - Temperatura 0.
    - Temperatura 0,5
    - Temperatura 1,0.
    - Temperatura 1,5.

Para cada temperatura se generan 10 piezas diferentes de 500 notas cada una. Cada pieza tiene una secuencia de cebado diferente (notas MIDI), que se indica en la siguiente tabla:

Tabla 2. Secuencia de cebado experimento 1

# Secuencia	primer_melody (secuencia de cebado)
1	[53, 57, 62, 64]
2	[62, 60, 57, 56]
3	[57, 53, 55, 53]
4	[53, 62, 60, 57]
5	[64, 62, 56, 53]
6	[53, 57, 62, 53]
7	[62, 57, 56, 57]
8	[57, 56, 57, 53]
9	[64, 62]
10	[57, 56]

Las partituras flamencas del estilo de la Soleá, generadas en este experimento, para cada uno de los valores de temperatura, se encuentran en formato MIDI en el Anexo III de este trabajo.

4.2. Experimento 2. Magenta Melody_rnn – Attention_rnn

Descripción del experimento

En este experimento se aborda la composición de música flamenca utilizando las librerías de Google Magenta y manteniendo el mismo modelo "*Melody_rnn*" que se utilizó en el experimento anterior, pero cambiando la configuración.

Es este caso, se utiliza la configuración "*Attention_rnn*". Al igual que en la configuración "*Basic_rnn*", únicamente admite música monofónica y no son posibles notas más rápidas que las semicorcheas ya que Magenta las considera polifonía. Cualquier partitura del dataset de Soleares que no cumpla con estas restricciones, es descartada por el framework durante los procesos previos de validación que se realizan antes de empezar el entrenamiento de la red neuronal.

Entradas

El dataset de entrada está formado por las partituras de Soleá en formato MIDI. Este conjunto de entrenamiento es el mismo para todos los experimentos que se realizan en este trabajo.

En esta configuración *"Attention_rnn"*, las partituras que se utilizan en el entrenamiento no son modificadas ni alteradas por Google Magenta, tal como ocurría con la configuración "*Basic-rnn*", donde eran traspuestas al rango de los tonos MIDI en el intervalo [48, 84].

Estructura de la red neuronal:

En este experimento se utiliza el modelo "*Melody_rnn*" y la configuración "*Attention*_rnn". La configuración interna es una arquitectura de red neuronal de tipo LSTM que utiliza una codificación One-Hot como representación de las melodías de entrada y un mecanismo de "*Atención*".

Los parámetros de entrada para este modelo son 2 capas LSTM de 64 neuronas cada una. Se utiliza esta topología de red neuronal, porque es la que viene implementada por defecto para esta configuración y porque suele dar buenos resultados.

El mecanismo de "*Atención*" permite a la red neuronal recuperar fácilmente la información pasada sin necesidad de guardar esa información dentro de las neuronas de la red recurrente. Es decir, las dependencias a largo plazo son aprendidas de forma más eficiente, lo que permite la generación de melodías más extensas.

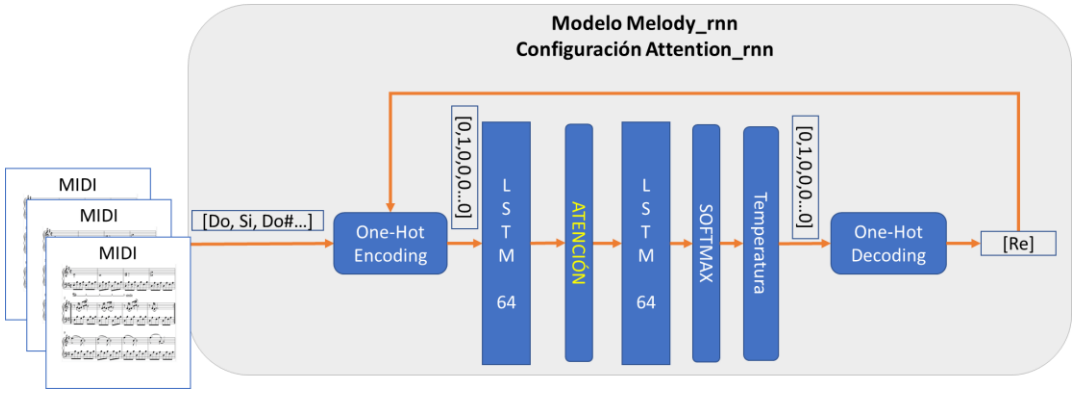

Figura 28. Configuración Attention_rnn del modelo Melody_rnn de Google Magenta. Fuente: Elaboración propia

Entrenamiento de la red:

Los parámetros de entrenamiento utilizados para esta configuración han sido:

- Tamaño batch: 64
- Epoch: 5.000

La red se entrena hasta conseguir:

3. Training Accuracy = 90,1%:

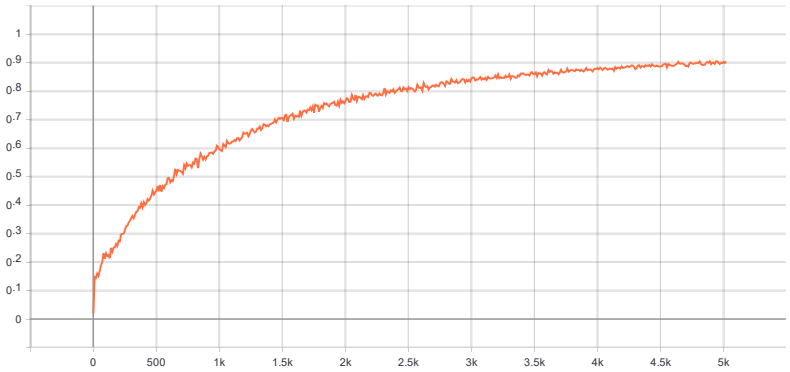

Figura 29.Accuracy del modelo. Configuración Attention_rnn. Fuente: Elaboración propia

4. Loss per steps = 0,3014:

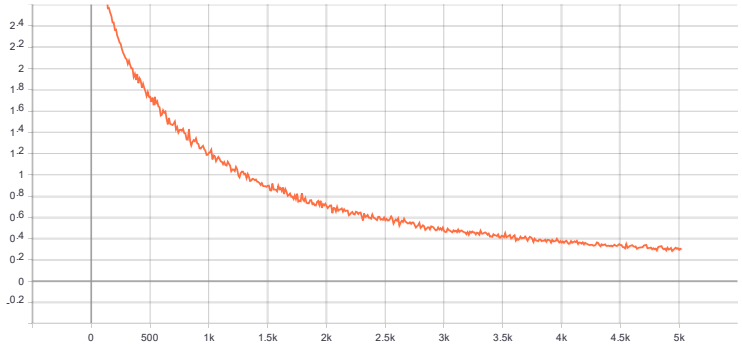

Figura 30. Loss per steps del modelo. Configuración Attention_rnn. Fuente: Elaboración propia

Generación:

Una vez realizado el entrenamiento se procede a la generación de música flamenca:

- Temperatura: Se utilizan 4 valores de temperatura diferentes:
    - Temperatura 0.
    - Temperatura 0,5
    - Temperatura 1,0.
    - Temperatura 1,5.

  Para cada temperatura se generan 10 piezas diferentes de 500 notas cada una. Cada pieza tiene una secuencia de cebado diferente (notas MIDI), que se indica en la siguiente tabla:

Tabla 3. Secuencia de cebado experimento 2

# Secuencia	primer_melody (secuencia de cebado)
1	[53, 57, 62, 64]
2	[62, 60, 57, 56]
3	[57, 53, 55, 53]
4	[53, 62, 60, 57]
5	[64, 62, 56, 53]
6	[53, 57, 62, 53]
7	[62, 57, 56, 57]
8	[57, 56, 57, 53]
9	[64, 62]
10	[57, 56]

Las partituras flamencas del estilo de la Soleá, generadas en este experimento, para cada uno de los valores de temperatura, se encuentran en formato MIDI en el Anexo III de este trabajo.

4.3. Experimento 3. LSTM Ajuste de Compás y Temperatura Variable

Descripción del experimento

El principal problema que tienen las redes neuronales a la hora de componer música muy rítmica, como puede ser la música flamenca, es la pérdida del compás a medida que se generan las notas musicales. En el caso de la Soleá, como ya se ha indicado, este compás es de 12 tiempos y la música que se genera debe estar sujeta a esta restricción.

En este experimento se incluye la información del compás como una entrada adicional a la red neuronal. Esto permite al sistema aprender qué notas van en cada tiempo del compás, y de este modo, generar música que esté sujeta a esta rítmica tan característica del flamenco.

Este experimento tiene varias novedades que se detallan y enumeran seguidamente:

1. La principal novedad que tiene este experimento es que se utiliza información rítmica del compás y que ésta se añade a la información que se extrae de las partituras MIDI. Es decir, si ya tenemos las notas que integran las piezas musicales y la duración de cada una de esas notas (o silencios), añadimos a esta información la correspondiente al compás.

   A cada tiempo del compás le corresponde la duración de una nota negra, de este modo, si tenemos 12 notas negras, cada una de ellas irá en un compás diferente. Si en lugar de tener notas negras tenemos corcheas, entonces 2 corcheas van en el mismo compás y así sucesivamente. Gráficamente sería:

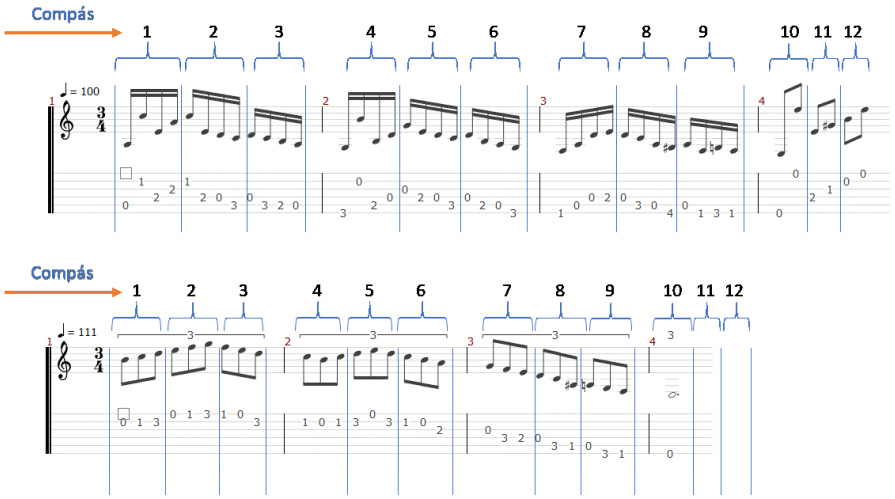

Figura 31. Formación del compás en la partitura. Fuente: Elaboración propia

Esta información rítmica se genera mediante un proceso informático que a cada nota y a su duración le asigna el compás que le corresponde. El estilo flamenco de la Soleá tiene una rítmica de 12 tiempos, por lo tanto, los valores que se asignan al compás están entre 1 y 12 a lo largo de todas las partituras del dataset de entrada (cuando llega a 12 vuelve a empezar en 1).

Esta forma de asignar el compás sería válida para otros estilos musicales como el pop, rock, vals etc.

Una vez construido el vector de compás, se integra con la información del valor de las notas y su duración.

2. Tradicionalmente se suele emplear un vector con la información de la nota y otro diferente con la información de la duración. Ambos vectores se codifican como One-Hot, se dividen en secuencias y se introducen en la red para su entrenamiento:

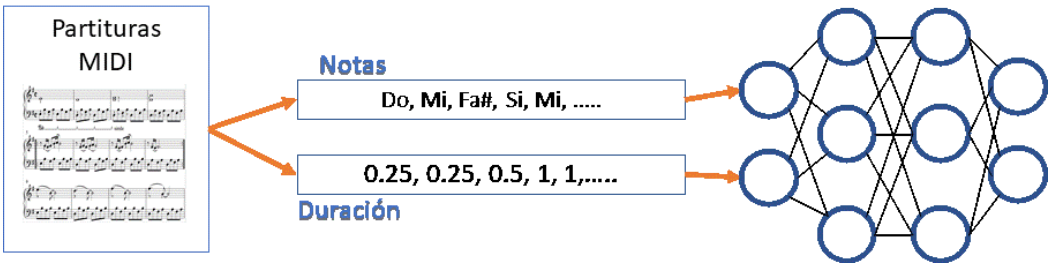

Figura 32. Modelo tradicional de 2 vectores de entrada. Fuente: Elaboración propia

Sin embargo, durante este experimento se añade otra importante novedad, consistente en incluir en el vector de notas, la concatenación de la nota y su duración. Además, mantenemos el vector de duración y, por último, añadimos el vector calculado de compás anteriormente explicado:

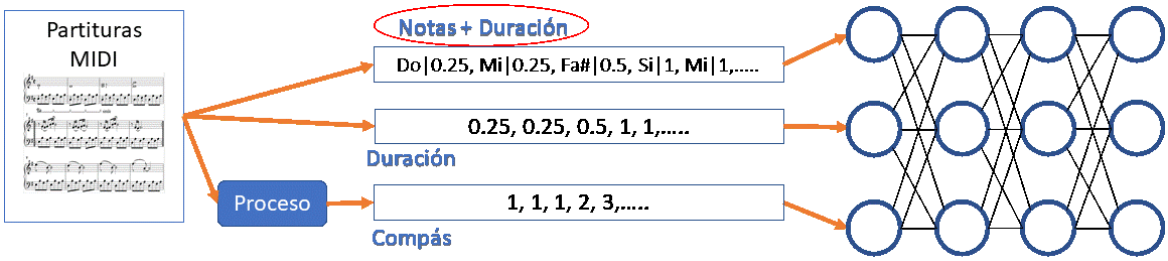

Figura 33. Modelo de 3 vectores de entrada utilizado en el experimento. Fuente: Elaboración propia

Es decir, la entrada a la red consta de 3 vectores:

- Vector de nota+duración: Se incluye en el mismo vector la nota y la duración concatenando ambos con un carácter especial (por ejemplo "|").
- Vector de duración: Se incluye la duración de cada nota en cada componente del vector.
- Vector de compás: En base a la duración de cada nota, se desarrolla un proceso que informe para cada nota y duración a que compás pertenece.

En principio, puede parecer que la información de la duración no es necesaria, puesto que ésta se concatena dentro del vector de notas, sin embargo, esto no es así, porque esta información resulta de utilidad para que la red aprenda el compás en base al vector de la duración. Si se quita este vector, el compás no podría aprenderse al estar la duración concatenada con la nota.

Por otra parte, al concatenar la nota y la duración dentro del mismo vector, estamos haciendo dependientes el valor de la nota y su duración, lo cual permite que la red aprenda que ciertas notas o secuencias de notas sólo se utilizan con determinadas duraciones. Es más, en teoría musical los valores de las notas siempre están íntimamente ligados a sus duraciones y no deben representarse de forma independiente valor y duración.

3. En este experimento se incluye otra novedad, consistente en el uso de mecanismos de "*ajuste de compás y temperatura variable*", cuyo funcionamiento se divide en 2 partes que se describen a continuación:

Funcionalidad 1:

Una vez que la red se ha entrenado, la utilizamos para generar secuencias de notas partiendo de un pequeño conjunto inicial o "*notas de cebado*". A medida que la red neuronal va generando notas, estas vuelven a incluirse como entrada para seguir con la

secuencia generativa. Esta retroalimentación se realiza de forma iterativa hasta que se genera el número de notas previamente establecido.

Puede ocurrir que, la información del compás que se genera no corresponda con la que debiera ser, lo que con toda certeza provoca que se rompa el ritmo y la melodía se vaya fuera de compas. En este caso, el mecanismo de "*ajuste de compás y temperatura variable*" actúa actualizando el valor del compás generado por la red por el que realmente le corresponde, para que la melodía siga manteniendo el compás que debe tener. Esto implica que este mecanismo debe tener control absoluto de las duraciones de las notas y silencios para saber en qué parte del compás se ubican las notas que se van generando por la red.

Gráficamente se observa que, si el compás generado coincide con el compás interno, no se realizan cambios:

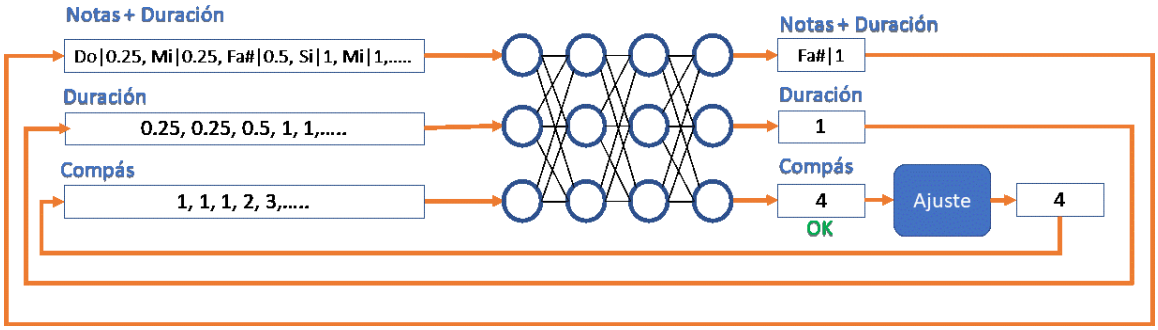

Figura 34. Mecanismo de ajuste de compás sin actuar. Fuente: Elaboración propia

Sin embargo, si el compás generado no coincide con el esperado, se actualiza con el valor correcto:

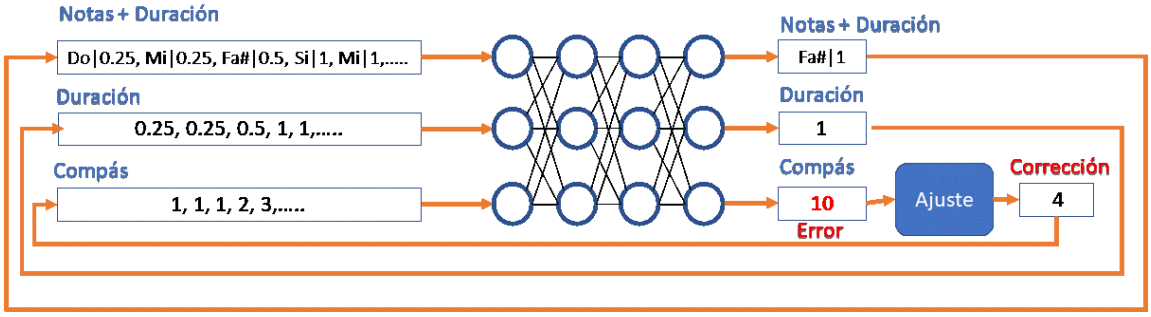

Figura 35. Mecanismo de ajuste de compás actuando. Fuente: Elaboración propia

Con este mecanismo se consigue que, aunque la red neuronal genere un valor de compás no válido, se cambie el valor por el correcto lo que fuerza a la red a seguir el compás propio de la Soleá en todas las composiciones.

Funcionalidad 2:

La temperatura es un hiperparámetro de las LSTM (y de las redes neuronales en general) que se utiliza para controlar la variabilidad de las predicciones que se generan en la capa de salida de tipo "*Softmax*". Cuando la temperatura es 0, se toman directamente los valores de probabilidad normalizados de "*Softmax*" sin cambio alguno. A medida que el valor de la temperatura aumenta, los valores más probables de "*Softmax*" se vuelven más pequeños y los más pequeños se hacen más grandes, lo que produce una distribución de probabilidad más suave sobre las clases y hace que la red se "*excite*" más fácilmente con las muestras, lo que resulta en más diversidad y también en más errores.

Utilizar la temperatura en la composición musical hace que las notas que se generan vayan variando de acuerdo con el valor de temperatura que se haya empleado. Un valor bajo, implica menor diversidad musical y un valor alto, implica mayor aleatoriedad y variabilidad.

El problema radica en que el hiperparámetro de la temperatura suele ser un valor fijo que no cambia una vez que se empiezan a generar las secuencias de notas por parte de la red neuronal.

Con el fin de evitar esto, el mecanismo de "*ajuste de compás y temperatura variable*" tiene la funcionalidad añadida de que usa un valor de temperatura diferente en función del número de compás (temperatura alta para los compases iniciales y temperatura 0 para las últimas partes). Esto hace que las melodías generadas sean más variables o aleatorias al inicio del compás, pero que finalmente se reconduzcan a un cierre típico de la Soleá al disminuir drásticamente el valor de la temperatura. Con esto se consigue generar aleatoriedad musical en la composición en los primeros tiempos, pero obligando a la melodía a ajustarse a algunos de los cierres de la Soleá característicos en los tiempos finales (10, 11, y 12). Este mecanismo tiene como entrada una lista de compases para los que la temperatura se pone a cero independientemente del valor de temperatura que se haya indicado. En este trabajo se ha usado la lista de compases de [8, 9, 10, 11, 12] con el fin de que únicamente haya aleatoriedad en los 7 primeros compases. Una vez pasados éstos, la temperatura toma el valor 0 con el fin de que la melodía se reconduzca hasta llegar al cierre en los últimos compases.

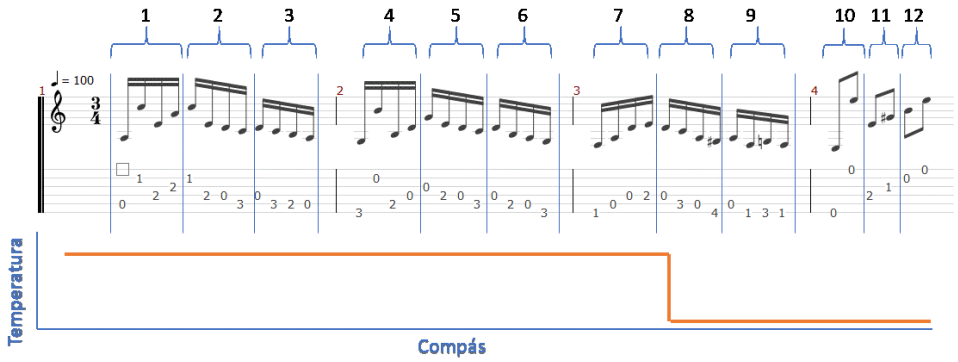

Figura 36. Evolución del valor de la temperatura en función del compás. Fuente: Elaboración propia

Los detalles acerca del mecanismo de temperatura se encuentran detallados en la codificación realizada para este experimento dentro del Anexo III.

Entradas

Para el entrenamiento de la red neuronal se utiliza como entrada el dataset de partituras flamencas en formato MIDI. Este conjunto de partituras monofónicas son las que hemos utilizado en los experimentos anteriores.

Las partituras en formato MIDI son transformadas en objetos mediante el conjunto de herramientas para el análisis musical asistido por ordenador denominado Music21 (Cuthbert, 2021). Este módulo, que se importa como una librería desde Python, realiza el tratamiento y la conversión a formato texto de acordes, notas y sus correspondientes duraciones, lo que permite que los eventos musicales sean procesados de forma más eficiente por parte de una red neuronal.

Para la representación de cada nota se utilizarán 3 vectores independientes:

1. Vector con codificación One-Hot, para las notas musicales y su duración. Está novedad es interesante ya que se introduce de forma simultánea la nota y su duración, o lo que es lo mismo, se codifica de forma conjunta la nota más la duración en One-Hot. Al utilizar melodías monofónicas y el rango de duración de las notas no superar el valor de semicorcheas, el conjunto de valores posibles[7] no afecta en exceso al entrenamiento de la red neuronal ya que este no es muy elevado.
2. Vector con codificación One-Hot para la duración de cada nota musical. Aunque el valor de la duración ya se está codificando junto con el valor de la nota, se vuelve a introducir a la entrada de la red para ayudar al vector de compás a que pueda realizar la suma de las duraciones y, de este modo, la rítmica sea más precisa y acorde con el compás de 12 tiempos de la Soleá.
3. Vector con codificación One-Hot para el compás donde se encuentra la nota. Este valor se calcula, mediante un proceso especial, a medida que se van leyendo las notas del dataset de entrada y de acuerdo con la duración de cada una de ellas.

Seguidamente se dividen los 3 vectores calculados en secuencias consecutivas de longitud n, asignando a cada secuencia de entrada cual es la siguiente nota en la partitura, es decir, la nota n+1 de las secuencias (también calculada como 3 vectores). Esto se realiza para todas las secuencias posibles que se pueden formar de forma consecutiva con las notas de todo el dataset de partituras de música flamenca del estilo de la Soleá.

El valor utilizado para n en este experimento es 16.

[7] El conjunto total de valores será (notas diferentes del dataset x duraciones posibles en el dataset)

Estructura de la red neuronal:

Se describe a continuación:

Tabla 4. Estructura de red neuronal del experimento 3

```
Model: "LSTM"
__________________________________________________________________________________________________
Layer (type)                 Output Shape              Param #     Connected to
==================================================================================================
input_1 (InputLayer)         [(None, None)]            0
__________________________________________________________________________________________________
input_2 (InputLayer)         [(None, None)]            0
__________________________________________________________________________________________________
input_3 (InputLayer)         [(None, None)]            0
__________________________________________________________________________________________________
embedding (Embedding)        (None, None, 128)         21120       input_1[0][0]
__________________________________________________________________________________________________
embedding_1 (Embedding)      (None, None, 128)         1920        input_2[0][0]
__________________________________________________________________________________________________
embedding_2 (Embedding)      (None, None, 128)         1536        input_3[0][0]
__________________________________________________________________________________________________
concatenate (Concatenate)    (None, None, 384)         0           embedding[0][0]
                                                                   embedding_1[0][0]
                                                                   embedding_2[0][0]
__________________________________________________________________________________________________
bidirectional (Bidirectional (None, None, 512)         1312768     concatenate[0][0]
__________________________________________________________________________________________________
dropout (Dropout)            (None, None, 512)         0           bidirectional[0][0]
__________________________________________________________________________________________________
lstm_1 (LSTM)                (None, 256)               787456      dropout[0][0]
__________________________________________________________________________________________________
dropout_1 (Dropout)          (None, 256)               0           lstm_1[0][0]
__________________________________________________________________________________________________
notas_y_duracion (Dense)     (None, 165)               42405       dropout_1[0][0]
__________________________________________________________________________________________________
duracion (Dense)             (None, 15)                3855        dropout_1[0][0]
__________________________________________________________________________________________________
compas_flamenco (Dense)      (None, 12)                3084        dropout_1[0][0]
==================================================================================================
Total params: 2,174,144
Trainable params: 2,174,144
Non-trainable params: 0
```

Gráficamente la topología de la red sería:

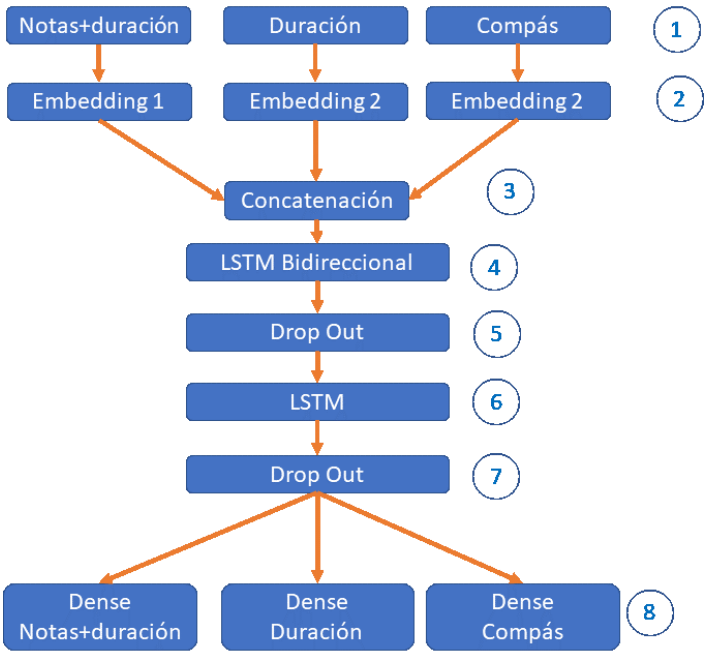

Figura 37. Arquitectura de red del experimento 3. Fuente: Elaboración propia

Hay 3 vectores de entrada (1) que corresponden con:

- Valor de las notas concatenado con su duración.
- Duraciones de las notas.
- Compás flamenco.

La capa de "*Embeddings*" (2) realizan la conversión de los valores enteros a vectores. El tamaño de cada uno de los vectores es de 128 elementos. Existe un vector de "*Embeddings*" para cada uno de los vectores de entrada.

Los vectores son concatenados (3) para conseguir un único vector que se utiliza como entrada a las capas recurrentes (LSTM).

Una capa LSTM bidireccional (4) como estructura recurrente configurada con "*Return_sequence*" a "*True*" con el fin de pasar todos los "*hidden states*" a la capa siguiente. Esta capa tiene 128 neuronas.

Capa de "*Dropout*" (5) configurada al 20% para evitar el sobreajuste (overfitting).

Una nueva capa LSTM (6) como estructura recurrente configurada con "*Return_sequence*" a "*True*" con el fin de pasar todos los "*hidden states*" a la capa siguiente. Esta capa tiene 128 neuronas.

Una nueva capa de "*Dropout*" (7) configurada al 20% para evitar el sobreajuste (overfitting).

La capa de salida (8) son 3 vectores de predicción "*Softmax*", cada uno de los cuales realiza las siguientes predicciones:

1. Vector de probabilidad con siguiente nota+duración.
2. Vector de probabilidad con la duración de la siguiente nota.
3. Vector de probabilidad de compás con la probabilidad del siguiente compás.

Entrenamiento de la red:

Los parámetros de entrenamiento utilizados para esta configuración son:

- Tamaño batch: 32
- Epoch: 100
- Loss: categorical_crossentropy

- RMSprop (learning rate = 0.001)

La red se entrena hasta conseguir:

- Training Accuracy: 95,8%:

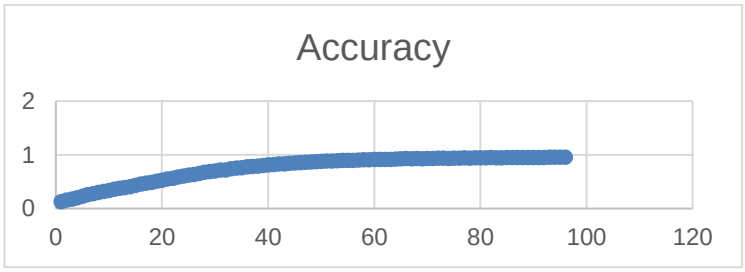

Figura 38. Evolución de los valores de accuracy en el experimento 3. Fuente: Elaboración propia

- Loss per steps: = 0,2272:

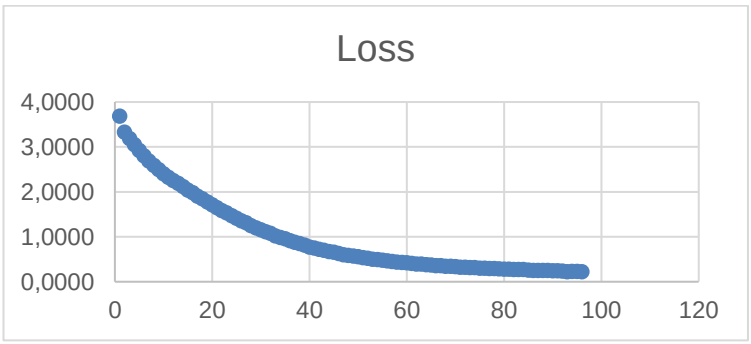

Figura 39. Evaluación de los valores de loss per step en el experimento 3. Fuente: Elaboración propia

Generación:

El mecanismo de "*ajuste de compás y temperatura variable*", descrito previamente, se aplica en este experimento para conseguir que la generación siga el patrón rítmico del estilo en cuestión, en este caso concreto el de la Soleá (12 tiempos).

Una vez realizado el entrenamiento, se procede a la generación de música flamenca:

- Temperatura: Se utilizan 4 valores de temperatura diferentes:
  - Temperatura 0.
  - Temperatura 0,5
  - Temperatura 1,0.

- Temperatura 1,5.

Para cada temperatura se generan 10 piezas diferentes de 500 notas cada una. Cada pieza tiene una secuencia de cebado diferente que se indica en la siguiente tabla:

Tabla 5. Secuencia de cebado experimento 3

# Secuencia	Secuencia de Cebado		
	Nota + Duración	Duración	Compás Flamenco
1	['A3|0.5', 'E4|0.5', 'C4|0.5']	[0.5, 0.5, 0.5]	[1, 1, 2]
2	['A4|0.25', 'E4|0.25', 'C4|0.25', 'A4|0.25']	[0.25, 0.25, 0.25, 0.25]	[1, 1, 1, 1]
3	['F3|0.25', 'B3|0.25', 'A3|0.25', 'D3|0.25']	[0.25, 0.25, 0.25, 0.25]	[1, 1, 1, 1]
4	['C4|0.25', 'B4|0.25', 'A4|0.25', 'B4|0.25']	[0.25, 0.25, 0.25, 0.25]	[1, 1, 1, 1]
5	['C5|0.25', 'F5|0.25', 'C5|0.25', 'F5|0.25']	[0.25, 0.25, 0.25, 0.25]	[1, 1, 1, 1]
6	['A4|0.25', 'C5|0.25', 'F5|0.25']	[0.25, 0.25, 0.25]	[1, 1, 1]
7	['A4|0.25', 'G4|0.25', 'A4|0.25', 'D4|0.25']	[0.25, 0.25, 0.25, 0.25]	[1, 1, 1, 1]
8	['E4|0.5', 'F4|0.5', 'E3|0.5','E4|0.5']	[0.5, 0.5, 0.5, 0.5]	[1, 1, 2, 2]
9	['G#3|0.5','B3|0.5']	[0.5, 0.5]	[1, 1]
10	['F3|0.25']	[0.25]	[1]

- Mecanismo de temperatura variable: Activo con cierre entre los compases 8 y 12 (entre 8 y 12 la temperatura toma el valor 0)

Las partituras flamencas del estilo de la Soleá, generadas en este experimento, para cada uno de los valores de temperatura, se encuentran en formato MIDI en el Anexo III de este trabajo.

4.4. Experimento 4. LSTM con Atención Ajuste de Compás y Temperatura Variable

Descripción del experimento

Este experimento es similar al que se ha realizado anteriormente, con la salvedad de que en esta configuración se va a utilizar un mecanismo de "*Atención*", el cual permite al modelo acceder más fácilmente a la información pasada sin tener que almacenar esta información en la red neuronal. Esto permite que el modelo aprenda más fácilmente las dependencias a largo plazo pudiendo generar melodías más largas.

En este experimento también se aplica el mecanismo de "*ajuste de compás y temperatura variable*" en la generación de las secuencias de salida, que se detalló en el experimento anterior.

Los detalles acerca del mecanismo de temperatura se encuentran detallados en la codificación realizada para este experimento dentro del Anexo III.

Entradas:

El dataset de entrada es el mismo conjunto de partituras que hemos utilizado hasta ahora para todos los experimentos. Al igual que en el experimento anterior, se utiliza la librería Music21 para realizar las transformaciones de notas y duraciones de formato MIDI a texto.

Seguidamente, y al igual que se hizo en el experimento previo, se crean los 3 vectores de entrada en codificación One-Hot y se generan las secuencias de n elementos consecutivos.

La generación de secuencias de entrada tiene una longitud de 16 notas

Estructura de la red neuronal:

Se describe a continuación:

Tabla 6. Estructura de red neuronal del experimento 4.

```
Model: "LSTM Attention"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            [(None, None)]       0
__________________________________________________________________________________________________
input_2 (InputLayer)            [(None, None)]       0
__________________________________________________________________________________________________
input_3 (InputLayer)            [(None, None)]       0
__________________________________________________________________________________________________
embedding_1 (Embedding)         (None, None, 128)    20608       input_1[0][0]
__________________________________________________________________________________________________
embedding_2 (Embedding)         (None, None, 128)    1792        input_2[0][0]
__________________________________________________________________________________________________
embedding_3 (Embedding)         (None, None, 128)    1536        input_3[0][0]
__________________________________________________________________________________________________
concatenate_1 (Concatenate)     (None, None, 384)    0           embedding_1[0][0]
                                                                 embedding_2[0][0]
                                                                 embedding_3[0][0]
__________________________________________________________________________________________________
bidirectional_1 (Bidirectional) (None, None, 512)    1312768     concatenate_1[0][0]
__________________________________________________________________________________________________
dropout_1 (Dropout)             (None, None, 512)    0           bidirectional_1[0][0]
__________________________________________________________________________________________________
lstm_2 (LSTM)                   (None, None, 256)    787456      dropout_1[0][0]
__________________________________________________________________________________________________
dropout_2 (Dropout)             (None, None, 256)    0           lstm_2[0][0]
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, None, 1)      257         dropout_2[0][0]
__________________________________________________________________________________________________
reshape_1 (Reshape)             (None, None)         0           dense_1[0][0]
__________________________________________________________________________________________________
activation_1 (Activation)       (None, None)         0           reshape_1[0][0]
__________________________________________________________________________________________________
repeat_vector_1 (RepeatVector)  (None, 256, None)    0           activation_1[0][0]
__________________________________________________________________________________________________
permute_1 (Permute)             (None, None, 256)    0           repeat_vector_1[0][0]
__________________________________________________________________________________________________
multiply_1 (Multiply)           (None, None, 256)    0           dropout_2[0][0]
                                                                 permute_1[0][0]
__________________________________________________________________________________________________
lambda_1 (Lambda)               (None, 256)          0           multiply_1[0][0]
__________________________________________________________________________________________________
notas_y_duracion (Dense)        (None, 161)          41377       lambda_1[0][0]
__________________________________________________________________________________________________
duracion (Dense)                (None, 14)           3598        lambda_1[0][0]
__________________________________________________________________________________________________
compas_flamenco (Dense)         (None, 12)           3084        lambda_1[0][0]
==================================================================================================
Total params: 2,172,476
Trainable params: 2,172,476
Non-trainable params: 0
```

Gráficamente la topología de la red sería:

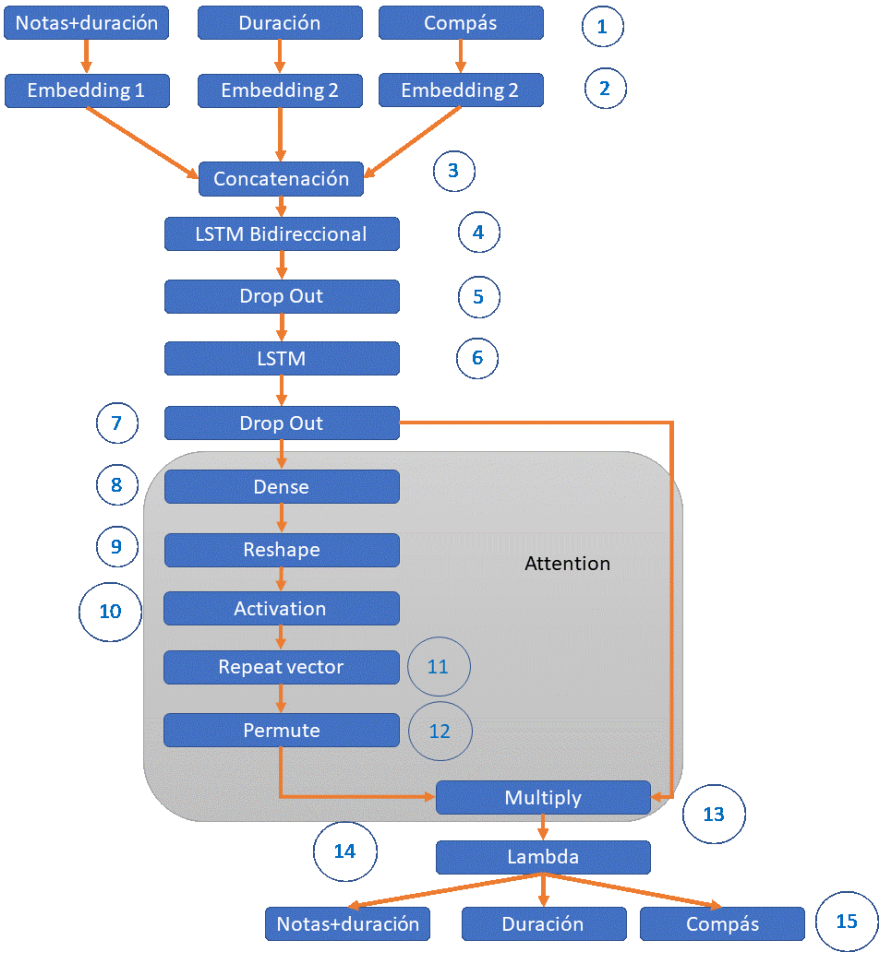

Figura 40. Arquitectura de red del experimento 4. Fuente: Elaboración propia

Hay 3 vectores de entrada (1) que corresponden con:

- Valor de las notas concatenado con su duración.
- Duraciones de las notas.
- Compás flamenco.

La capa de "*Embeddings*" (2) realiza la conversión de los valores enteros a vectores. El tamaño de cada uno de los vectores es de 128 elementos. Existe un vector de "*Embeddings*" para cada uno de los vectores de entrada.

Los vectores son concatenados (3) para conseguir un único vector que se utilizará como entrada a las capas recurrentes (LSTM).

Una capa LSTM bidireccional (4) como estructura recurrente configurada con "*Return_sequence*" a "*true*" con el fin de pasar todos los "*hidden states*" a la capa siguiente. Esta capa tiene 128 neuronas.

Capa de "*Dropout*" (5) configurada al 20% para evitar el sobreajuste (overfitting).

Una nueva capa LSTM (6) como estructura recurrente configurada con "*Return_sequence*" a "*True*" con el fin de pasar todos los "*hidden states*" a la capa siguiente. Consta de 128 neuronas.

Una nueva capa de "*Dropout*" (7) configurada al 20% para evitar el sobreajuste (overfitting).

La función de alineación es sólo una capa densa (8) con una unidad de salida y activación tanh. Se utiliza una capa de "*Reshape*" (9) para reducir la salida a un único vector de longitud igual a la de la secuencia de entrada.

Se calculan los pesos aplicando una capa de activación "*Softmax*" (10) a los valores anteriores.

Para obtener la suma de los pesos de los "*hidden states*", es necesario utilizar una capa "*Repeatvector*" (11) con pesos de las unidades LSTM para formar una matriz de dimensiones [Rnn_units, Seq_length].

Seguidamente, se transpone esta matriz mediante una capa "*Permute*" (12) para obtener una matriz de dimensiones [Seq_length, Rnn_units].

Se realiza la multiplicación (13) de esta matriz con los "*hidden states*" que proceden de la última capa LSTM que también tiene dimensiones [Seq_length, Rnn_units].

Finalmente, se utiliza una capa "*Lambda*" (14) para realizar la suma a lo largo del eje "S*eq_length*", para crear el vector de contexto de longitud "*Rnn_units*".

La capa de salida (15) son 3 vectores de predicción "*Softmax*", cada uno de los cuales realiza la siguiente predicción:

1. Vector de probabilidad con siguiente nota+duración.
2. Vector de probabilidad con la duración de la siguiente nota.
3. Vector de probabilidad de compás con la probabilidad del siguiente compás.

Entrenamiento de la red:

Los parámetros de entrenamiento utilizados para esta configuración son:

- Tamaño batch: 32
- Epoch: 106
- Loss: categorical_crossentropy
- RMSprop(learning rate = 0.001)

La red se entrenada hasta conseguir:

- Training Accuracy: 98,01%:

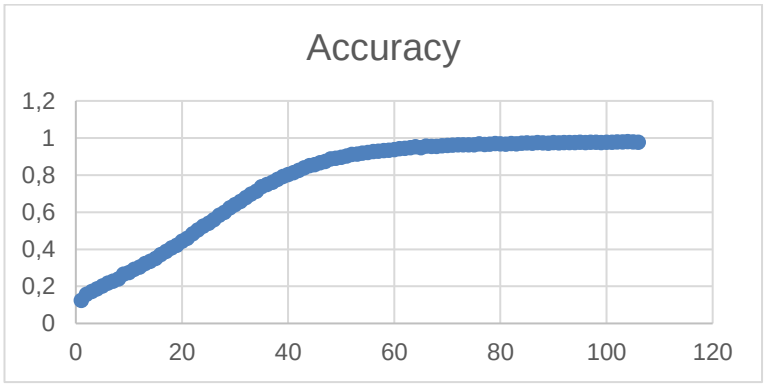

Figura 41. Evolución de los valores de accuracy en el experimento 4. Fuente: Elaboración propia.

- Loss per steps: 0,0895%:

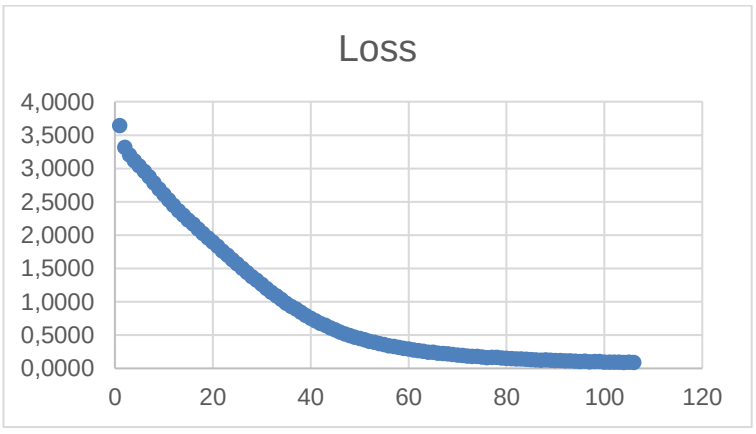

Figura 42. Evaluación de los valores de loss per step en el experimento 4. Fuente: Elaboración propia

Generación:

El mecanismo de "*ajuste de compás y temperatura variable*" también se aplica en este experimento. Recordemos que consistía en el tratamiento post-generación que se aplica al compás para conseguir que la música generada siga el patrón rítmico del estilo en cuestión, en este caso en concreto el de la Soleá (12 tiempos).

Una vez realizado el entrenamiento se procede a la generación de música flamenca:

- Temperatura: Se utilizan 4 valores de temperatura diferentes:
    - Temperatura 0.
    - Temperatura 0,5.
    - Temperatura 1,0.
    - Temperatura 1,5.

  Para cada temperatura se generan 10 piezas diferentes, de 500 notas cada una. Cada pieza tiene una secuencia de cebado diferente que se indica en la siguiente tabla:

Tabla 7. Secuencia de cebado experimento 4

# Secuencia	Secuencia de Cebado		
	Nota + Duración	Duración	Compás Flamenco
1	['A3|0.5', 'E4|0.5', 'C4|0.5']	[0.5, 0.5, 0.5]	[1, 1, 2]
2	['A4|0.25', 'E4|0.25', 'C4|0.25', 'A4|0.25']	[0.25, 0.25, 0.25, 0.25]	[1, 1, 1, 1]
3	['F3|0.25', 'B3|0.25', 'A3|0.25', 'D3|0.25']	[0.25, 0.25, 0.25, 0.25]	[1, 1, 1, 1]
4	['C4|0.25', 'B4|0.25', 'A4|0.25', 'B4|0.25']	[0.25, 0.25, 0.25, 0.25]	[1, 1, 1, 1]
5	['C5|0.25', 'F5|0.25', 'C5|0.25', 'F5|0.25']	[0.25, 0.25, 0.25, 0.25]	[1, 1, 1, 1]
6	['A4|0.25', 'C5|0.25', 'F5|0.25']	[0.25, 0.25, 0.25]	[1, 1, 1]
7	['A4|0.25', 'G4|0.25', 'A4|0.25', 'D4|0.25']	[0.25, 0.25, 0.25, 0.25]	[1, 1, 1, 1]
8	['E4|0.5', 'F4|0.5', 'E3|0.5','E4|0.5']	[0.5, 0.5, 0.5, 0.5]	[1, 1, 2, 2]
9	['G#3|0.5','B3|0.5']	[0.5, 0.5]	[1, 1]
10	['F3|0.25']	[0.25]	[1]

- Mecanismo de temperatura variable: Activo con cierre entre los compases 8 y 12 (entre 8 y 12 la temperatura toma el valor 0).

Las partituras flamencas del estilo de la Soleá, generadas en este experimento, para cada uno de los valores de temperatura, se encuentran en formato MIDI en el Anexo III de este trabajo.

5. Descripción de los Resultados

En este apartado se indican de forma detallada los resultados obtenidos para cada uno de los experimentos realizados.

Las partituras generadas en cada experimento han sido evaluadas y sus resultados tabulados según se indica en el apartado 3.3 Metodología del trabajo.

5.1. Resultados Experimento 1. Magenta Melody_rnn – Basic_rnn

En este primer experimento no se ha incluido información del compás a la entrada de la red neuronal, se ha utilizado la librería de Google Magenta con el modelo "*Melody_rnn*" y la configuración "*Basic_rnn*". Se han generado 10 partituras diferentes para cada una de las temperaturas que a continuación se indican y con un total de 500 notas musicales para cada partitura:

- **Temperatura 0:**

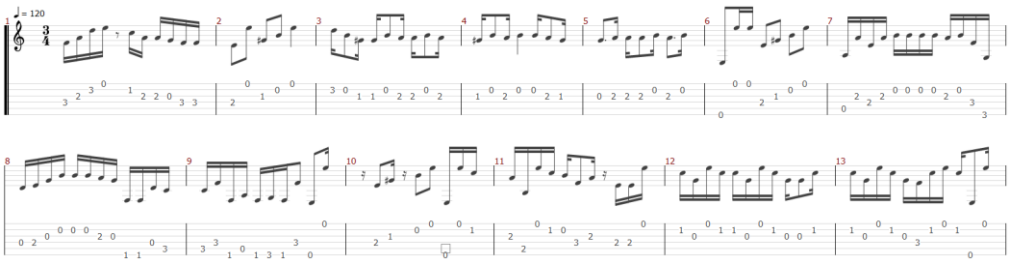

Figura 43. Extracto de partitura generada en el experimento 1 con temperatura 0. Fuente: Elaboración propia

- **Temperatura 0,5:**

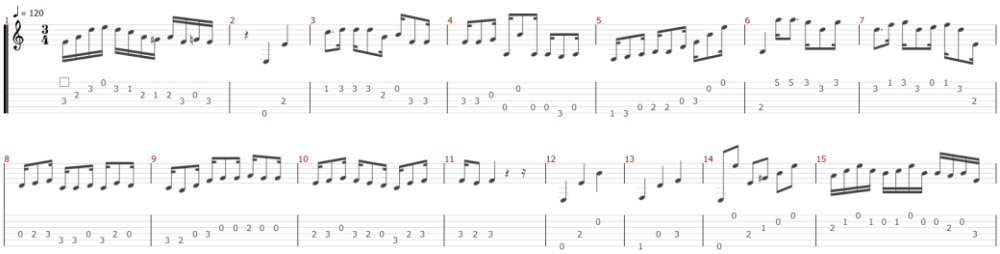

Figura 44. Extracto de partitura generada en el experimento 1 con temperatura 0,5. Fuente: Elaboración propia

- **Temperatura 1,0:**

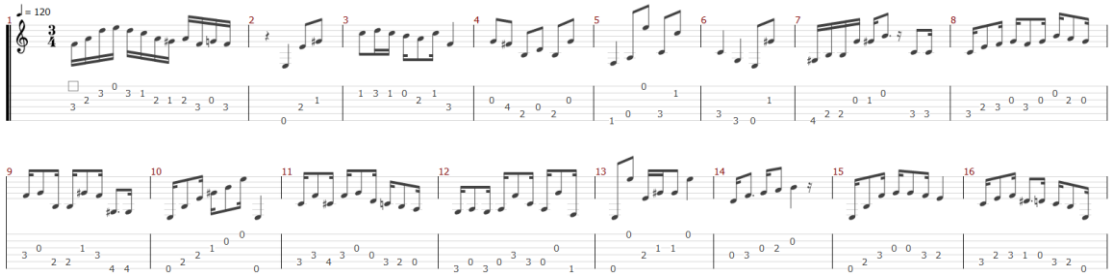

Figura 45. Extracto de partitura generada en el experimento 1 con temperatura 1,0. Fuente: Elaboración propia

- **Temperatura 1,5:**

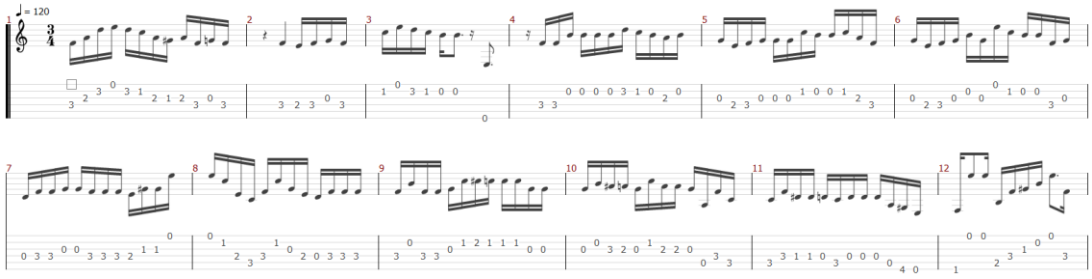

Figura 46. Extracto de partitura generada en el experimento 1 con temperatura 1,5. Fuente: Elaboración propia

El mecanismo de temperatura que proporciona Google Magenta toma valores a partir de 0, siendo éste el valor mínimo en el cual no existe aleatoriedad y en el que se escoge como salida la nota más probable de las que ofrece la red neuronal ("*Softmax*"). A medida que se incrementa la temperatura, se incrementa la variabilidad y aleatoriedad de las notas que son generadas por la red neuronal.

Todas las partituras que se han generado en este experimento, para cada una de las diferentes temperaturas, se encuentran en formato MIDI en el Anexo III de este trabajo.

Evaluación musical:

Las partituras generadas en este experimento han sido analizadas por diferentes evaluadores con el fin de obtener cuantas falsetas siguen el estilo flamenco de la Soleá. El resultado se muestra tabulado en función del valor de la temperatura de generación:

Tabla 8. Resultado de la evaluación experimento 1

Temperatura	Partituras de 500 Notas	Falsetas Generadas	Evaluador Profesional	Evaluador Avanzado	Evaluador Aficionado
0	10	100	22	24	36
0,5	10	94	14	20	30
1	10	96	14	22	28
1,5	10	102	16	20	24
TOTAL	**40**	**392**	**66**	**86**	**118**
Porcentajes por Evaluador			**16,83%**	**21,93%**	**30,10%**
Porcentaje Medio Final			**22,95%**		

5.2. Resultados Experimento 2. Magenta Melody_rnn – Attention_rnn

En este segundo experimento no se ha incluido la información del compás a la entrada de la red, se ha utilizado la librería de Google Magenta con el modelo "*Melody_rnn*" y la configuración "*Attention_rnn*". Se han generado 10 partituras diferentes para cada una de las temperaturas que a continuación se indican:

- **Temperatura 0:**

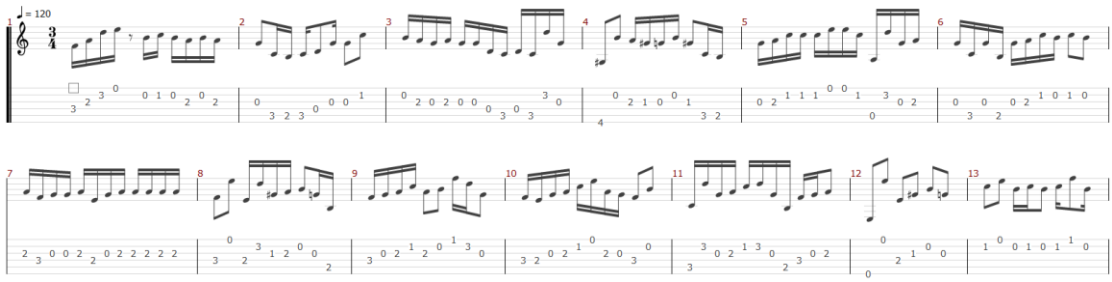

Figura 47. Extracto de partitura generada en el experimento 2 con temperatura 0. Fuente: Elaboración propia

- **Temperatura 0,5:**

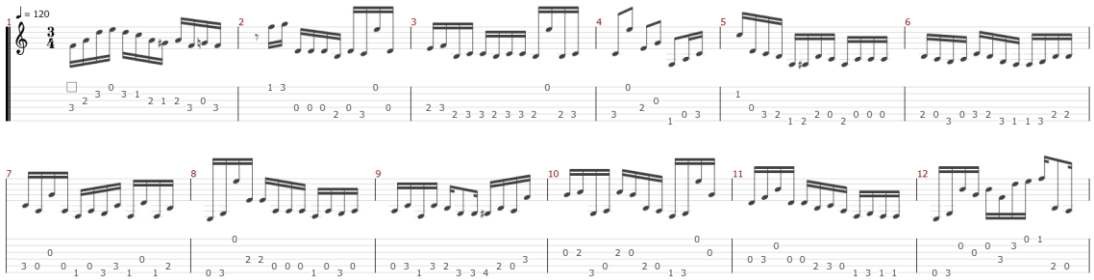

Figura 48. Extracto de partitura generada en el experimento 2 con temperatura 0,5. Fuente: Elaboración propia

- **Temperatura 1,0:**

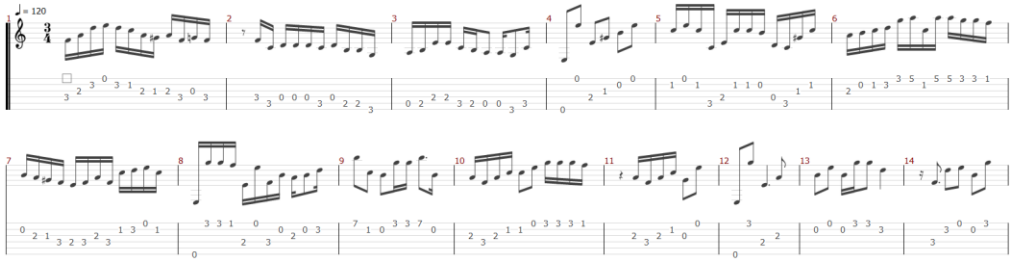

Figura 49. Extracto de partitura generada en el experimento 2 con temperatura 1,0. Fuente: Elaboración propia.

- **Temperatura 1,5:**

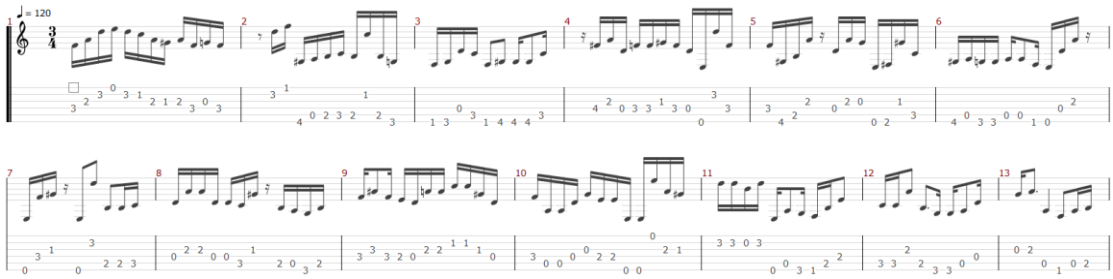

Figura 50. Extracto de partitura generada en el experimento 2 con temperatura 1,5. Fuente: Elaboración propia

Al igual que en el experimento anterior, el mecanismo de temperatura que proporciona Google Magenta toma valores a partir de 0, siendo éste el valor mínimo en el cual no existe aleatoriedad y en el que se escoge como salida la nota más probable de las que ofrece la red neuronal ("*Softmax*"). A medida que se incrementa la temperatura, se incrementa la variabilidad y aleatoriedad de las notas que son generadas por la red neuronal.

Todas las partituras que se han generado en este experimento, para cada una de las diferentes temperaturas, se encuentran en formato MIDI en el Anexo III de este trabajo.

Evaluación musical:

Las partituras generadas en este experimento han sido analizadas por diferentes evaluadores con el fin de obtener cuantas falsetas siguen el estilo flamenco de la Soleá. El resultado se muestra tabulado en función del valor de la temperatura de generación:

Tabla 9. Resultado de la evaluación experimento 2

Temperatura	Partituras de 500 Notas	Falsetas Generadas	Evaluador Profesional	Evaluador Avanzado	Evaluador Aficionado
0	10	104	12	14	16
0,5	10	102	14	18	18
1	10	92	10	14	15
1,5	10	100	10	12	15
TOTAL	**40**	**398**	**46**	**58**	**64**
Porcentajes por Evaluador			**11,55%**	**14,57%**	**16,08%**
Porcentaje Medio Final			**14,06%**		

5.3. Resultados Experimento 3. LSTM

En este tercer experimento, en el que se incluye la información acerca del compás, se ha desarrollado una red neuronal específica para la generación de música flamenca para el estilo de la Soleá. Se han generado 10 partituras diferentes para cada una de las temperaturas que a continuación se indican:

- **Temperatura 0:**

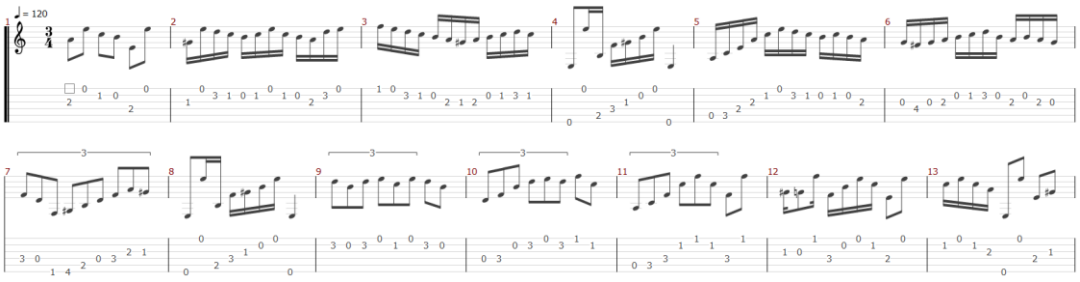

Figura 51. Extracto de partitura generada en el experimento 3 con temperatura 0. Fuente: Elaboración propia

- **Temperatura 0,5:**

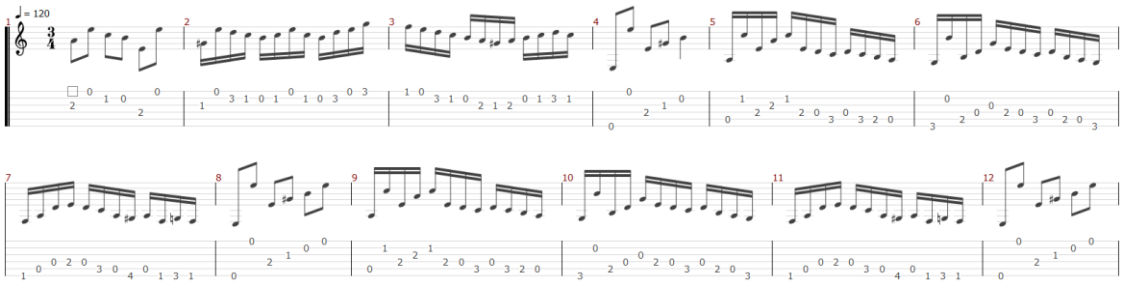

Figura 52. Extracto de partitura generada en el experimento 3 con temperatura 0,5. Fuente: Elaboración propia

- **Temperatura 1,0:**

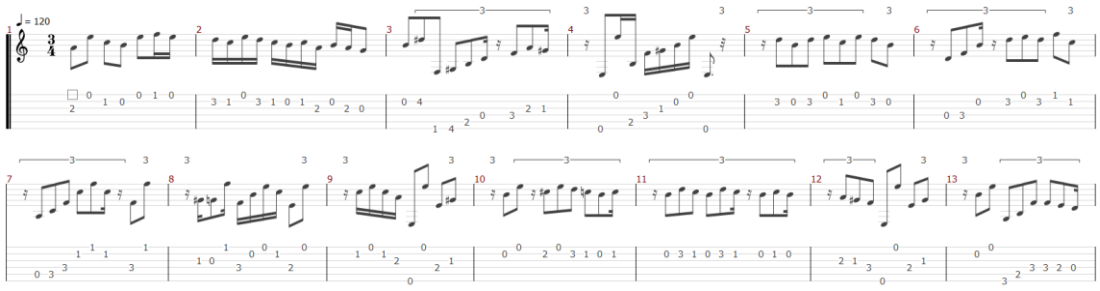

Figura 53. Extracto de partitura generada en el experimento 3 con temperatura 1,0. Fuente: Elaboración propia

- **Temperatura 1,5:**

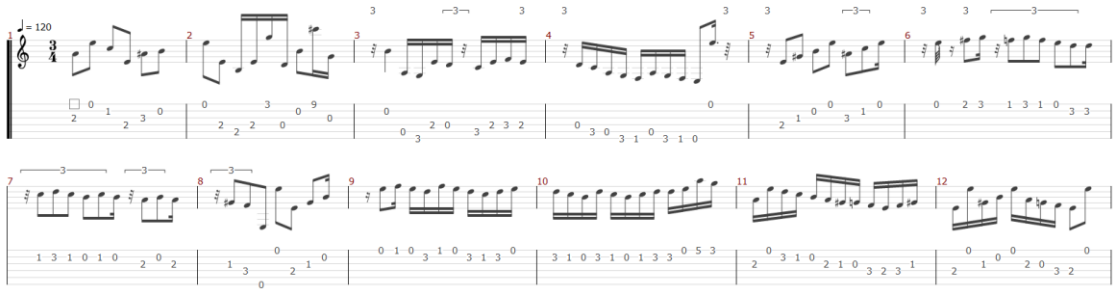

Figura 54. Extracto de partitura generada en el experimento 3 con temperatura 1,5. Fuente: Elaboración propia

Todas las partituras que se han generado en este experimento, para cada una de las diferentes temperaturas, se encuentran en formato MIDI en el Anexo III de este trabajo. Adicionalmente, en el Anexo IV, se incluyen algunos ejemplos subidos a Soundcloud.

Evaluación musical:

Las partituras generadas en este experimento han sido analizadas por diferentes evaluadores con el fin de obtener cuantas falsetas siguen el estilo flamenco de la Soleá. El resultado se muestra tabulado en función del valor de la temperatura de generación:

Tabla 10. Resultado de la evaluación experimento 3

Temperatura	Partituras de 500 Notas	Falsetas Generadas	Evaluador Profesional	Evaluador Avanzado	Evaluador Aficionado
0	10	140	30	32	30
0,5	10	142	130	134	128
1	10	135	102	105	112
1,5	10	147	38	42	46
TOTAL	**40**	**564**	**300**	**313**	**316**
Porcentajes por Evaluador			**53,19%**	**55,49%**	**56,02%**
Porcentaje Medio Final			**54,9%**		

5.4. Resultados Experimento 4. LSTM y Atención

En este cuarto experimento, en el que se incluye la información acerca del compás, se ha desarrollado una red neuronal específica para la generación de música flamenca para el estilo de la Soleá. Se han generado 10 partituras diferentes para cada una de las temperaturas que a continuación se indican:

- **Temperatura 0:**

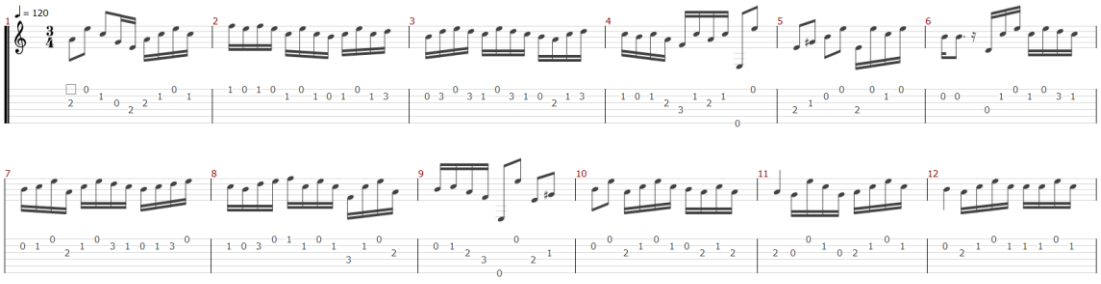

Figura 55. Extracto de partitura generada en el experimento 4 con temperatura 0. Fuente: Elaboración propia

- **Temperatura 0,5:**

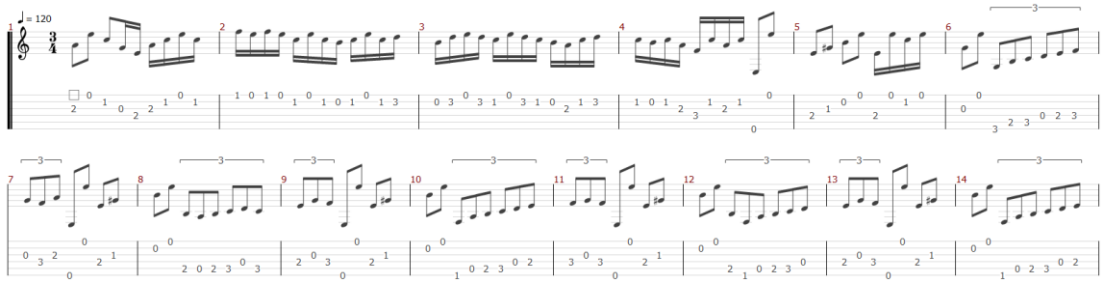

Figura 56. Extracto de partitura generada en el experimento 4 con temperatura 0,5. Fuente: Elaboración propia

- **Temperatura 1,0:**

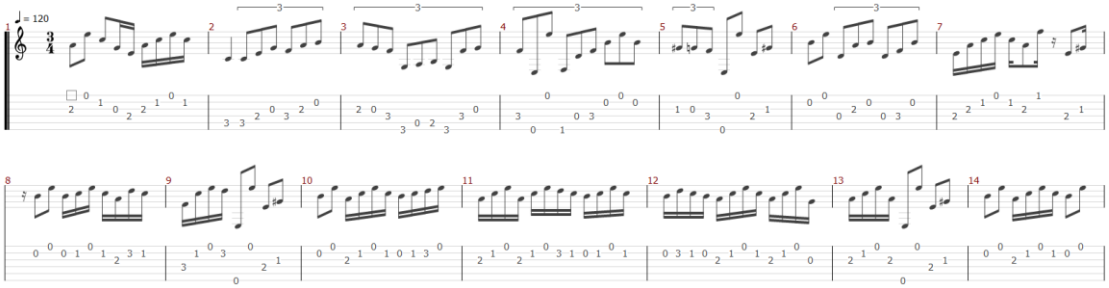

Figura 57. Extracto de partitura generada en el experimento 4 con temperatura 1,0. Fuente: Elaboración propia

- **Temperatura 1,5:**

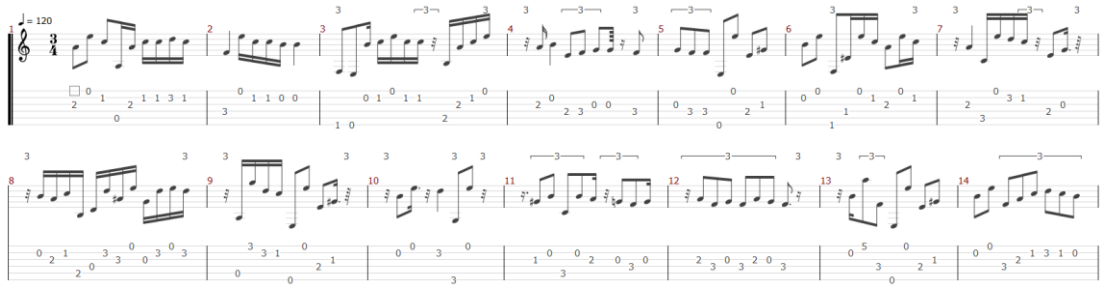

Figura 58. Extracto de partitura generada en el experimento 4 con temperatura 1,5. Fuente: Elaboración propia

Todas las partituras que se han generado en este experimento, para cada una de las diferentes temperaturas, se encuentran en formato MIDI en el Anexo III de este trabajo. Adicionalmente, en el Anexo IV, se incluyen algunos ejemplos subidos a Soundcloud.

Evaluación musical:

Las partituras generadas en este experimento han sido analizadas por diferentes evaluadores con el fin de obtener cuantas falsetas siguen el estilo flamenco de la Soleá. El resultado se muestra tabulado en función del valor de la temperatura de generación:

Tabla 11. Resultado de la evaluación experimento 4

Temperatura	Partituras de 500 Notas	Falsetas Generadas	Evaluador Profesional	Evaluador Avanzado	Evaluador Aficionado
0	10	136	90	94	98
0,5	10	142	132	136	136
1	10	138	101	105	105
1,5	10	142	42	45	46
TOTAL	**40**	**558**	**365**	**380**	**385**
Porcentajes por Evaluador			**65,41%**	**68,10%**	**68,99%**
Porcentaje Medio Final			**67,5%**		

5.5. Comparativa de los Resultados de los Experimentos

La siguiente tabla muestra el resumen de los resultados obtenidos para cada uno de los experimentos realizados:

Tabla 12. Comparativa de resultados de los experimentos.

Experimento	Tipo Experimento	Ajuste Compás Temperatura Variable	Evaluador Profesional	Evaluador Avanzado	Evaluador Aficionado	Porcentaje Medio Final
1	Magenta Melody_rnn Basic_rnn	No	**16,83%**	**21,93%**	**30,10%**	**22,95%**
2	Magenta Melody_rnn Atention_rnn	No	**11,55%**	**14,57%**	**16,08%**	**14,06%**
3	Red LSTM	Sí	**53,19%**	**55,49%**	**56,02%**	**54,9%**
4	Red LSTM con mecanismo de “Atención”	Sí	**65,41%**	**68,10%**	**68,99**	**67,5%**

En las columnas de evaluadores se indican el porcentaje de falsetas que son consideradas flamencas para cada uno de los evaluadores (profesional, avanzado y aficionado). Cada porcentaje se calcula como la media aritmética de los porcentajes obtenidos para las diferentes temperaturas.

La columna porcentaje medio final se calcula como la media aritmética de los porcentajes de los evaluadores para cada experimento. Este valor es el que se va a utilizar para comparar los diferentes resultados de cada experimento.

Al comparar el porcentaje medio final, se puede comprobar que los mejores resultados se obtienen para el experimento 4.

6. Discusión

En base a los resultados obtenidos en los diferentes experimentos realizados anteriormente, podemos destacar que:

- El framework Google Magenta no es la utilidad que mejor se adaptan a la generación de música flamenca y en concreto a la composición de melodías del estilo de la Soleá. Los resultados de porcentaje medio final de 22,95% para la configuración "*Basic_rnn*" y 14,06% para "*Attention_rnn*", muestran que las partituras que se generan no van a compás. Es decir, las falsetas no terminan con un cierre en el compás 12, algo que es fundamental para conseguir que la melodía tenga la rítmica propia de la Soleá.

- Los experimentos realizados con la configuración "*Basic_rnn*" del modelo "*Melody_rnn*" acotan el rango de generación de las melodías a los valores MIDI [48, 84], lo que en términos de música flamenca implica, por una parte, reducir el rango tonal, y por otra, generar notas que no son propias del estilo flamenco de la Soleá.

- La configuración de "*Attention_rnn*" del modelo "*Melody_rnn*" de Google Magenta permite la generación de notas en todo el rango tonal lo que hace que las melodías creadas se asemejen más al estilo de la Soleá. No obstante, al igual que la configuración *"Basic_rnn"*, las partituras generadas carecen de compás y no se ajustan a la línea melódica que debería tener una Soleá. Entendiendo por línea melódica que las notas estén dentro de los acordes propios de la cadencia de la Soleá (Mi, Fa, Sol, Lam) y que suenen de forma coherente, sin disonancias excesivas.

- Los valores de porcentaje medio final de 14,06% para la configuración de "*Attention_rnn*" del modelo "*Melody_rnn*" resulta levemente inferior al valor de 22,95% que se obtiene para la configuración "*Basic_rnn*". La razón es que, aunque se producen menos disonancias (se mejora la melodía con el mecanismo de atención), se pierde el compás propio de la Soleá y las falsetas no terminan en el tiempo 12.

- Con la utilización de los mecanismos de "*ajuste de compás y temperatura variable*" se consigue que, aunque la red neuronal genere un valor de compás no válido, se cambie el valor por el correcto, lo que fuerza a la red a seguir el compás propio de la soleá en todas las composiciones. Además, el uso de un valor de temperatura diferente en función del número de compás (temperatura alta para los compases iniciales y temperatura 0 para las últimas partes) hace que las melodías generadas sean más variables o aleatorias al inicio del compás, pero que finalmente se reconduzcan a un cierre típico de la Soleá al disminuir drásticamente el valor de la temperatura. Con esto se consigue generar aleatoriedad musical en la composición en los primeros tiempos, pero obligando a la melodía a ajustarse a algunos de los cierres de la Soleá característicos en los tiempos finales.

- La creación de 3 vectores diferentes con las secuencias de las notas que son introducidos a la entrada de la red neuronal mejora la generación de música flamenca ya que cada nota se genera de forma conjunta con la duración de ésta dentro del primer vector. Esto implica que el valor de la nota y su duración sean computados

como una unidad por la red neuronal durante el entrenamiento y que la salida del valor de la nota sea acorde a su duración. Por otra parte, la utilización del vector de duración de forma independiente ayuda a que el vector de compás pueda entrenarse correctamente puesto que puede calcular cual es el siguiente compás en base a las duraciones anteriores que ha procesado.

- Los mecanismos de "*Atención*" utilizados en el último experimento permiten a la red fijarse en las partes más importantes del dataset de entrenamiento, lo que permite aprender las dependencias a largo plazo de forma más eficiente y generar piezas de Soleá más extensas.

- Los valores de porcentaje medio final de 54,9% y 67,5% obtenidos con los experimentos 3 y 4 indican que se obtiene mejores resultados con respecto a los experimentos 1 y 2 realizados con Google Magenta, cuyos valores son 22,95% y 14,06%. Esto demuestra que, los nuevos mecanismos utilizados (ajuste de compas, temperatura variable y el vector de compas), son efectivos a la hora de generar música flamenca ya que hacen que las piezas lleven la rítmica propia de este estilo y que las melodías suenen coherentes sin disonancias excesivas.

- Dependiendo de la temperatura utilizada se obtiene mayor o menor variabilidad. Cuando se utiliza temperatura 0 las piezas se vuelven repetitivas transcurridos unos compases. En el extremo opuesto, cuando se utiliza la temperatura 1,5, la música generada es demasiado aleatoria y se producen excesivas disonancias en todos los experimentos. Las temperaturas entre 0,5 y 1,0 son las que mejores resultados ofrecen, aportando suficiente aleatoriedad para una variabilidad musical, sin perder la línea melódica propia de la Soleá.

- La diferencia entre los valores de porcentaje medio final de los experimentos 3 y 4 son pequeñas (54,9% frente a 67,5%), no obstante, se obtiene mejores resultados en el experimento 4 (67,5%) porque las melodías generadas con temperatura 0 no resultan tan repetitivas como las generadas con el experimento 3, por el uso del mecanismo de "*Atención*".

- De los resultados obtenidos se deduce que la mejor arquitectura para la generación de música flamenca es la red LSTM con "*Atención*", mecanismo de "*ajuste de compás y temperatura variable*" y utilizando temperaturas entre 0,5 y 1.

7. Conclusiones y Trabajo Futuro

7.1. Conclusiones

En este trabajo se han realización un conjunto de experimentos con distintas arquitecturas de Deep Learning con el fin de explorar diferentes técnicas de composición de música flamenca para guitarra del estilo de la Soleá.

Se han revisado las arquitecturas más importantes que se han utilizado a lo largo de los años en la generación de contenidos musicales, desde los inicios en composición algorítmica, hasta las técnicas más actuales basadas en Deep Learning, poniendo especial énfasis en las redes LSTM, en los mecanismos de "*Atención*" y en las estrategias de "*Condicionamiento*".

Una vez estudiadas las diferentes arquitecturas y mecanismos, se han realizado diferentes experimentos con el fin de probar la eficacia de cada uno de ellos en el ámbito de la generación de música flamenca del estilo de la Soleá. En este sentido, se ha trabajado con 2 tipos de arquitecturas diferentes: Google Magenta (experimentos 1 y 2) y una arquitectura desarrollada específicamente para la composición de música flamenca basada en LSTM (experimentos 3 y 4).

En la primera de ellas (Google Magenta), los valores de porcentaje medio final obtenidos en los experimentos 1 y 2 son de 22,95% y 14,06%, respectivamente. Esto indica que las composiciones flamencas de Soleá generadas por este framework no siguen el compás propio de este estilo y tienen excesivas disonancias.

En la segunda arquitectura, utilizada en los experimentos 3 y 4, se han añadido las siguientes novedades:

- Vector de compás: Permite que la red aprenda el ritmo, su cadencia, los cierres que se utilizan y cuando utilizarlos, además de las notas y duraciones que pueden ir en cada compás.
- Concatenación del valor de la nota con su duración: Permite que la red aprenda de forma conjunta que nota está asociada a qué duraciones.
- Nuevo mecanismo de "*ajuste de compás y temperatura variable*": Permite corregir el valor del compás cuando se genera una predicción errónea de este, lo que fuerza a la melodía a mantener la rítmica propia de la Soleá. Además, la idea de permitir la variabilidad de la melodía en los primeros compases y una vez pasados estos

eliminarla (temperatura variable), hace que la melodía llegue al cierre de forma controlada, generando las secuencias típicas de cierre de la Soleá. Esto incrementa la aleatoriedad musical sin perder el control en las secuencias finales tan propias del flamenco.

Estas novedades hacen que la generación de música flamenca para el estilo de la Soleá mejore significativamente en términos de compás y línea melódica, con respecto a la música generada con las librerías de Google Magenta, consiguiéndose unos porcentajes medios finales de 54,9% (experimento 3) y 67,5% (experimento 4).

Además de las novedades anteriores, en el experimento 4 se ha incluido un mecanismo de "*Atención*", que permite al modelo acceder más fácilmente a la información pasada sin tener que almacenar esta información en la red neuronal. Esto permite que el modelo aprenda más fácilmente las dependencias a largo plazo pudiendo generar melodías más largas. El porcentaje medio final que resulta de incluir el mecanismo de atención en el experimento 4 es de 67,5%, frente al 54,9% del experimento 3 (que no incluye este mecanismo). Es decir, existe una mejora significativa gracias al uso de este mecanismo de "*Atención*", por lo que se puede concluir que, la red desarrollada en el experimento 4, es la que mejor resultado ofrece en la composición de música flamenca para guitarra del estilo de la Soleá.

No obstante, podemos destacar como problema, el derivado del uso de determinados formatos utilizados para la representación de los datos de entrada a la red neuronal. Concretamente, el formato utilizado para representar el primer vector, donde se unen el valor de la nota más el valor de su duración. Esto hace que el tamaño del vocabulario sea más extenso de lo habitual (notas diferentes x duraciones diferentes), lo que supone un incremento de los recursos de computación necesarios para el entrenamiento de la red neuronal.

Finalmente, los experimentos realizados y los resultados obtenidos nos permiten concluir que se ha alcanzado el objetivo general inicial de estudiar distintas arquitecturas y técnicas de Inteligencia Artificial aplicadas a la composición de música flamenca para guitarra del estilo de la Soleá.

7.2. Líneas de trabajo futuro

Con el fin de continuar los estudios que se han empezado con este trabajo de fin de Máster, se proponen las siguientes líneas de investigación:

- En este trabajo sólo se han utilizado los modelos de Google Magenta que generan música monofónica y no se han incluido los modelos que generan música polifónica.

En estudios futuros se podrían abordar los modelos polifónicos de Magenta, como por ejemplo "*Performance_rnn*", y explorar las diversas configuraciones disponibles con el fin de generar música flamenca.

- Aplicación de los mecanismos de "*ajustes de compás y temperatura variable*" a otros palos flamencos (Tangos, Bulerías, Seguiriyas, etc.).

- Aplicación de la codificación de secuencias en 3 vectores (nota+duración, duración y compás) a otros estilos musicales (pop, rock, jazz) con el fin de controlar el ritmo propio de cada estilo.

- En este estudio se han realizado diferentes experimentos que implican la generación de música flamenca para la Soleá. Este palo flamenco se considera rítmico y sujeto a un compás de 12 tiempos. No obstante, en el flamenco existen otros estilos que se consideran de "*ejecución libre*". Esto quiere decir que no están sujetos a ningún compás y el "*tocaor*" tiene libertad absoluta para desarrollar su toque sin sujeciones rítmicas. Tal es el caso de la Taranta o la Granaína. Por lo tanto, futuras líneas de estudio pueden ir orientadas a generar música flamenca para estos estilos libres.

- El estudio se ha realizado utilizando música monofónica, transformando las partituras polifónicas en monofónicas para poder utilizarse tanto en Google Magenta como en el resto de los experimentos. Como futura línea de investigación, se propone la ejecución de los mismos experimentos, especialmente en los que no se utiliza Google Magenta, utilizando un dataset polifónico del estilo flamenco de la Soleá o de cualquier otro palo.

- Estudios más avanzados en tiempo y alcance pueden ser los derivados de la utilización de un dataset de diferentes estilos musicales, con el fin de generar música hibrida o música fusión y comprobar si los resultados obtenidos son musicalmente válidos.

- Utilización de las notas "*disonantes*" y "*fuera de compas*" para su uso como "*semillas*" para pasos posteriores o para su utilización como frases de improvisación o de disonancia intencional.

- Utilización de una 2ª red neuronal que aprenda los errores, y así trabajar con un sistema más complejo donde el módulo de "*ajuste de compás y temperatura variable*" no sea determinista, sino también basado en Inteligencia Artificial.

- Por último, se podrían plantear trabajos futuros con diferentes arquitecturas e incluso combinaciones de varios tipos de arquitecturas. Adicionalmente, se podrían realizar estudios para la generación multipista, donde cada pista puede tener asociada una arquitectura diferente.

8. Bibliografía

Agon Amado, C. A. (1998). *OpenMusic : Un langage visuel pour la composition musicale assistée par ordinateur*. 1998. http://recherche.ircam.fr/equipes/repmus/Rapports/CarlosAgon98/index.html

Alpern, A. (1995). *Techniques for Algorithmic Composition of Music*.

Arroyo-Hernández, J., Travieso, C., Ticay-Rivas, J., Mora, F., Huertas, O., & Bogantes, M. (2013). *Sistema de detección y clasificación automática de granos de polen mediante técnicas de procesado digital de imágenes*. 59–73.

Bahdanau, D., Cho, K. H., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 1–15.

Barreau, P., & Barreau, V. (2016). *AIVA - The AI composing emotional soundtrack music*. https://www.aiva.ai/

BBC Online. (2016). *Google's AI beats world Go champion in first of five matches - BBC News*. https://www.bbc.com/news/technology-35761246

Bhatt, S. (2018). *5 Things You Need to Know about Reinforcement Learning - KDnuggets*. https://www.kdnuggets.com/2018/03/5-things-reinforcement-learning.html

Bocardo, L. (2018). *Transcribiendo hacia un Lead Sheet*. https://steemit.com/spanish/@leoboccardo/transcribiendo-hacia-un-lead-sheet

Boulanger-Lewandowski, N., Bengio, Y., & Vincent, P. (2012). Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, *2*(Cd), 1159–1166.

Briot, J.-P., & Pachet, F. (2017). *Music Generation by Deep Learning - Challenges and Directions*. 1–17. https://doi.org/10.1007/s00521-018-3813-6

Briot, J. P. (2020). From artificial neural networks to deep learning for music generation: history, concepts and trends. *Neural Computing and Applications*, *33*(1), 39–65. https://doi.org/10.1007/s00521-020-05399-0

Briot, J. P., Hadjeres, G., & Pachet, F.-D. (2020). *Deep Learning Techniques for Music Generation*.

Carrera Portero, R. (2016). *el flamenco a fondo*. https://sites.google.com/site/elflamencoafondo/home

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 1724–1734. https://doi.org/10.3115/v1/d14-1179

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. 1–9. http://arxiv.org/abs/1412.3555

Cope, D. (2000). *The Algorithmic Composer* (I. A-R Editions (ed.)).

creandopartituras. (2012). *Pentagrama, claves y notas - Creando PartiturasCreando Partituras*. http://www.creandopartituras.com/pentagrama-claves-y-notas/

Cuthbert, M. (2021). *music21 Documentation*. https://web.mit.edu/music21/doc/

Dhariwal, P., Jun, H., Payne, C., Kim, J. W., Radford, A., & Sutskever, I. (2020). *Jukebox: A Generative Model for Music*. http://arxiv.org/abs/2005.00341

Di Nunzio, A. (2013). *Musicomp – Musica Informatica*. https://www.musicainformatica.org/topics/musicomp.php

Dong, H. W., Hsiao, W. Y., Yang, L. C., & Yang, Y. H. (2018). Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, 34–41.

Downie, J. S. (1993). *Music Information Retrieval (MIR)*.

Ebcioglu, K. (1988). Expert System for Harmonizing Four-Part Chorales. In *Computer Music Journal* (Vol. 12, Issue 3, pp. 43–51). https://doi.org/10.2307/3680335

Eck, D., & Schmidhuber, J. (2002). A First Look at Music Composition using LSTM Recurrent Neural Networks. *Istituto Dalle Molle Di Studi Sull Intelligenza Artificial*, *103*, 48.

El-atril. (2021). *Los midi*. http://el-atril.com/midis/El midi.htm

Eligetuviolin. (2018). *Iniciación al lenguaje musical*. https://www.eligetuviolin.com/blog/135-el-compás-iniciacion-al-lenguaje-musical-para-padres-de-musicos-3

Engel, J., Resnick, C., Roberts, A., Dieleman, S., Norouzi, M., Eck, D., & Simonyan, K. (2017). Neural audio synthesis of musical notes with WaveNet autoencoders. *34th International Conference on Machine Learning, ICML 2017*, *3*, 1771–1780.

FloydHub. (2019). *Attention Mechanism*. https://blog.floydhub.com/attention-mechanism/#bahdanau-att

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*.

Google. (2016). *Magenta*. https://magenta.tensorflow.org/

Google Developers. (2019). *Overview of GAN Structure | Generative Adversarial Networks*. https://developers.google.com/machine-learning/gan/gan_structure

Google Developers. (2020). *Google Magenta Models - GitHub*. https://github.com/magenta/magenta/blob/master/magenta/models/README.md

Griffiths, P. (1980). *The New Grove Dictionary of Music and Musicians*.

Grout, Donald Jay. Palisca, C. V. (2001). *A history of Western music*.

Guan, F., Yu, C., & Yang, S. (2019). *A GAN Model With Self-attention Mechanism To Generate Multi-instruments Symbolic Music*.

Guitar Pro Tabs. (2021). *Guitar Pro Tabs*. https://www.guitarprotabs.net/

Guttman, N. (1957). *The Silver Scale/Pitch Variations*. http://sfsound.org/tape/Guttman.html

Hadjeres, G., & Nielsen, F. (2017). *Interactive Music Generation with Positional Constraints using Anticipation-RNNs*. 1–9. http://arxiv.org/abs/1709.06404

Hadjeres, G., Pachet, F., & Nielsen, F. (2017). DeepBach: A steerable model for bach chorales generation. *34th International Conference on Machine Learning, ICML 2017*, *3*, 2187–2196.

Hawthorne, C., Elsen, E., Song, J., Roberts, A., Simon, I., Raffel, C., Engel, J., Oore, S., & Eck, D. (2018). Onsets and frames: Dual-objective piano transcription. *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018*, 50–57. https://doi.org/10.5281/zenodo.1492341

Hochreiter, S., & Schmidhuber, J. (1997). *Long Short-Term Memory*. *1780*, 1735–1780.

Huang, C. Z. A., Vaswani, A., Uszkoreit, J., Shazeer, N., Simon, I., Hawthorne, C., Dai, A. M., Hoffman, M. D., Dinculescu, M., & Eck, D. (2019). Music transformer: Generating music with long-term structure. *7th International Conference on Learning Representations, ICLR 2019*, 1–15.

Hyungui LIm, Seungyeon Rhyu, K. L. (2017). CHORD GENERATION FROM SYMBOLIC MELODY USING BLSTM NETWORKS Music and Audio Research Group , Graduate School of Convergence Science and Technology 4 Center for Super Intelligence Seoul National University , Korea. *CoRR*, *abs/1712.0*, 621–627. http://arxiv.org/abs/1712.01011

IBM. (2011). *IBM100 - Deep Blue*. https://www.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/

IBM. (2020). *What are Recurrent Neural Networks? | IBM*. https://www.ibm.com/cloud/learn/recurrent-neural-networks

Internet Archive. (n.d.). *The Illiac I*. Retrieved May 2, 2021, from https://web.archive.org/web/20120414203023/http://ems.music.uiuc.edu/history/illiac.html

Internet Archive. (2003). *Función de Evaluación*. https://web.archive.org/web/20101025020914/http://www.fenach.cl/docs/memoria/node76.html

Jaques, N., Gu, S., Bahdanau, D., Hernández-Lobato, J. M., Turner, R. E., & Eck, D. (2017). Sequence tutor: Conservative fine-tuning of sequence generation models with KL-control. *34th International Conference on Machine Learning, ICML 2017*, *4*, 2587–2596.

Johnson, M., & Shotton, J. (2017). AUTOMATIC STYLISTIC COMPOSITION OF BACH CHORALES WITH DEEP LSTM Mark Gotham University of Cambridge. *Ismir 2017*.

KDnuggets. (2021). *Using Topological Data Analysis to Understand the Behavior of Convolutional Neural Networks - KDnuggets*. https://www.kdnuggets.com/2018/06/topological-data-analysis-convolutional-neural-networks.html

Kotecha, N. (2018). *Bach2Bach: Generating Music Using A Deep Reinforcement Learning Approach*. http://arxiv.org/abs/1812.01060

Lecun, Y., Bottou, L., Bengio, Y., & Ha, P. (1998). GradientBased Learning Applied to Document Recognition. *Proceedings of the IEEE*, *November*, 1–46.

Lewis, J. P. (1988). *Creation by refinement: a creativity paradigm for gradient descent learning networks*.

Luong, M. T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing*, 1412–1421. https://doi.org/10.18653/v1/d15-1166

Makemusic. (2015). *MusicXML Documentation*. http://usermanuals.musicxml.com/MusicXML/MusicXML.htm

Maurer, J. A. (1999). *The History of Algorithmic Composition*. https://ccrma.stanford.edu/~blackrse/algorithm.html

Misterica. (2017). *El epitafio de Sícilo. La composición musical más antigua*. https://www.misterica.net/el-epitafio-de-sicilo/

MJx Music. (2021). *Spotify – MJx Music*. https://open.spotify.com/artist/3Cg0rSSfCzV3tJZQyaKlRX?si=ltIRDVzxQVKgg_4bF273DA&dl_branch=1&nd=1

Mora Merchán, J. M. (2018). *Análisis de cadenas derivadas de modelos desconocidos .* Sevilla.

Mozer, M. (1994). *Neural Network Music Composition by Prediction: Exploring the Benefits of Psychoacoustic Constraints and Multi-scale Processing.*

Musescore Org. (2018). *MuseScore*. https://musescore.org/es/manual/descripción-del-producto-musescore

Musescore Org. (2021). *Editor Piano Roll | MuseScore*. https://musescore.org/es/manual/editor-piano-roll

Musicateoria. (2011a). *Teoría de la Música: Figuras y silencios*. https://musicateoria.files.wordpress.com/2011/03/figuras-y-silencios.jpg

Musicateoria. (2011b). *Teoría de la Música: Relaciones entre notas*. https://musicateoria.files.wordpress.com/2011/03/relacion-entre-figuras.jpg

musicinformationretrieval. (2021). *chroma*. https://musicinformationretrieval.com/chroma.html

Natya*ML. (2021). *Natya*ML*. https://dance-project.glitch.me/

Nayebi, A., & Vitelli, M. (2015). GRUV: Algorithmic Music Generation using Recurrent Neural Networks. *Deep Learning for Natural Language Processing*, 1–6. https://web.stanford.edu/~anayebi/projects/CS_224D_Final_Project_Writeup.pdf

Nuñez, F. (2011). *Flamencopolis | Descubre el Flamenco*. https://www.flamencopolis.com/

Olah, C. (2015). *Understanding LSTM Networks*. http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Oord, A. van den, Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A.,

Kalchbrenner, N., Senior, A., & Kavukcuoglu, K. (2016). *WaveNet: A Generative Model for Raw Audio*. 1–15. http://arxiv.org/abs/1609.03499

Pachet, F. (2018). *Por qué la inteligencia artificial no puede crear la canción del verano*. La Vanguardia. https://www.lavanguardia.com/ciencia/20180717/45948321612/francois-pachet-musica-inteligencia-artificial.html

Palisca, C. V., & Burkholder, J. P. (2006). *Epitaph of Seikilos*.

Perso flamenco & classic. (2021). *Partituras para guitarra clasica y flamenca, flamenco , tablaturas, tab, partituras*. http://perso.flamenco.free.fr/menu-esp.php

Pons, J. (2018). *Neural Networks For Music: A Journey Through Its History | by Jordi Pons | Towards Data Science*. https://towardsdatascience.com/neural-networks-for-music-a-journey-through-its-history-91f93c3459fb

Rivera, A. S. (2019). *Qué son las claves musicales*. https://www.unprofesor.com/musica/que-son-las-claves-musicales-3481.html

Roberts, A., Engel, J., Raffel, C., Hawthorne, C., & Eck, D. (2018). A hierarchical latent vector model for learning long-term structure in music. *35th International Conference on Machine Learning, ICML 2018*, *10*, 6939–6954.

Rothman, P. (2017). *NAMM 2017: Talking MusicXML and more with Michael Good - Scoring Notes*. https://www.scoringnotes.com/news/namm-2017-talking-musicxml-and-more-with-michael-good/

Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, *45*(11), 2673–2681. https://doi.org/10.1109/78.650093

Singhal, G. (2020). *Introduction to LSTM Units in RNN | Pluralsight*. https://www.pluralsight.com/guides/introduction-to-lstm-units-in-rnn

Sony CSL. (2016). *Flow Machines – AI assisted music production*. https://www.flow-machines.com/

Spotify. (n.d.). *Innovating for Writers and Artists – Spotify for Artists*. Retrieved May 2, 2021, from https://artists.spotify.com/blog/innovating-for-writers-and-artists

Stackoverflow. (2020). *tensorflow - What is the difference between Luong attention and Bahdanau attention? - Stack Overflow*. https://stackoverflow.com/questions/44238154/what-is-the-difference-between-luong-attention-and-bahdanau-attention

Steinberg.help. (2017). *Espectrograma*. https://steinberg.help/wavelab_pro/v9.5/es/wavelab/topics/audio_files_editing/wave_window_spectrogram_r.html

Sturm, B. L., Santos, J. F., Ben-Tal, O., & Korshunova, I. (2016). *Music transcription modelling and composition using deep learning*. http://arxiv.org/abs/1604.08723

Swada, L. T. (2014). *Illiac Suite for String Quartet*. http://lydiaswada.wordpress.com/2014/02/17/illiac-suite-for-string-quartet/

Sympathy for the Lawyer. (2017). *Qué es la sincronización musical*.

https://sympathyforthelawyer.com/2017/04/25/sincronizacion-musical/

TabsFlamenco.com. (2021). *Transcriptions for Flamenco Guitar – Flamenco Guitar Transcriptions*. https://tabsflamenco.com/

Tjoa, S. (2021). *Notes on Music Information Retrieval (MIR)*. https://musicinformationretrieval.com/

Todd, P. M. (1989). Connectionist approach to algorithmic composition. *Computer Music Journal*, *13*(4), 27–43. https://doi.org/10.2307/3679551

Tuxguitar Team. (2020). *TuxGuitar 1.5.4 para Windows*. https://tuxguitar.uptodown.com/windows

Wei, R., Garcia, C., El-Sayed, A., Peterson, V., & Mahmood, A. (2020). Variations in Variational Autoencoders - A Comparative Evaluation. *IEEE Access*, *8*, 153651–153670. https://doi.org/10.1109/ACCESS.2020.3018151

Wikipedia. (2021). *Notación musical - Wikipedia, la enciclopedia libre*. https://es.wikipedia.org/wiki/Notación_musical

Yang, L. C., Chou, S. Y., & Yang, Y. H. (2017). Midinet: A convolutional generative adversarial network for symbolic-domain music generation. *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017*, 324–331.

8.1. Bibliografía Adicional

Briot, J.P. (2020). *Deep Learning Techniques for Music Generation*. Switzerland: Editorial Springer.

Foster, D. (2019). *Generative Deep Learning*. Sebastopol: Editorial O'Reilly Media.

Alexandre, D. (2020). *Music Generation with Magenta*. Birmingham: Editorial Packt Publishing.

Anexos

Anexo I. Representación de la Música

Según Jean-Pierre Briot (J. P. Briot et al., 2020), la principal división que se puede plantear a la hora elegir una representación de los datos de entrada y de salida de una pieza musical es "*subsimbólico*" (audio) frente a "*simbólico*" (notas musicales). Esta división equivale a elegir entre un formato continuo y un formato discreto de las variables de entrada y de salida.

Como se puede ver, la naturaleza de cada una de las representaciones es muy diferente, al igual que las técnicas para el procesamiento y transformación de la representación inicial. Sin embargo, el procesamiento real de estos dos tipos de representación de la información por una topología Deep Learning es básicamente el mismo.

1. Representación Subsimbólica

El primer tipo de representación de la música es la señal de audio, bien como onda o bien como su transformada.

El uso de la representación transformada de la señal de audio que se utiliza para la representación de la información implica la compresión de los datos y la selección de aquellas componentes que mejor los representan, pero a costa de perder algo de información e introducir algún sesgo.

1.1 Onda

La ventaja de la onda está en considerar la materia prima sin transformar, en bruto, con toda su resolución inicial y con todos sus matices. Las arquitecturas que procesan la señal en bruto se denominan arquitecturas de extremo a extremo. La principal desventaja para este tipo de representaciones radica en la carga computacional que debe soportar el sistema en términos de memoria y de procesamiento.

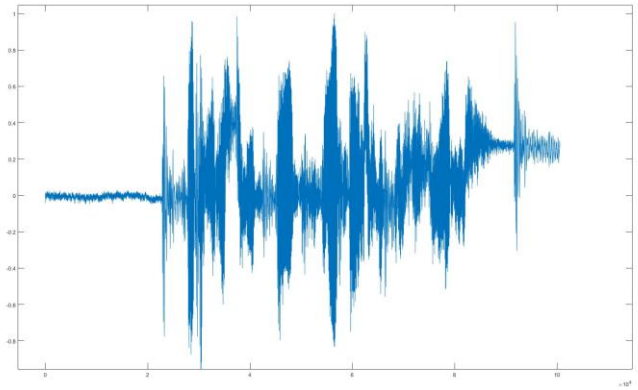

Figura A1.1 Representación de una onda de audio. Fuente: Elaboración propia

1.2 Espectrograma

El espectrograma es una representación visual de la intensidad ("*sonoridad*") de una señal a lo largo del tiempo en varias frecuencias presentes en una forma de onda concreta. Son, por tanto, gráficos bidimensionales, con una tercera dimensión representada por los colores.

Las dimensiones de las que consta un espectrograma son:

- El eje horizontal representa el tiempo, que va de izquierda (más antiguo), a la derecha (más actual).
- El eje vertical representa la frecuencia o tono (en el ámbito musical), con las frecuencias más bajas en la parte inferior y las frecuencias más altas en la parte superior.
- El color representa la amplitud (“*energía*” o "*sonoridad*") de una frecuencia concreta en un momento determinado. Los tonos azulados corresponden a amplitudes bajas y los tonos más rojizos, a amplitudes más altas.

De este modo, no sólo se puede representar si hay más o menos energía a determinadas frecuencias (5 Hz frente a 8 Hz), sino que también se visualiza cómo varían los diferentes niveles de energía en el transcurso del tiempo. Además de su uso para representación musical, los espectrogramas se utilizan también para mostrar las frecuencias de las ondas de audio producidas por la voz, máquinas, animales como ballenas, etc., tal y como es recogida por los micrófonos.

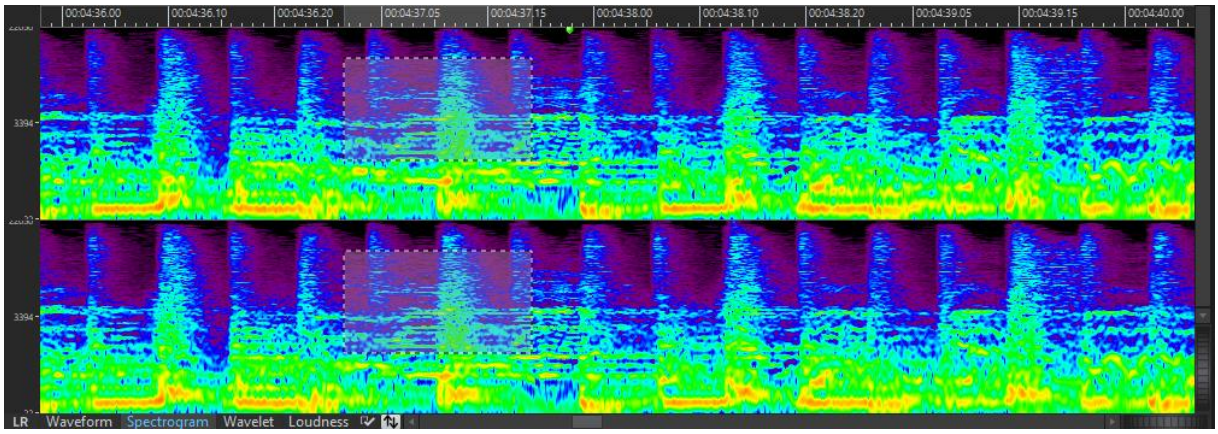

Figura A1.2. Espectrograma. Fuente: (Steinberg.help, 2017)

1.3 Cromagrama

Esta representación para el audio musical es un caso especial del espectrograma más general en el que el sonido analizado son notas musicales. Las diferencias entre ambos radican en que en el espectrograma se suele representar la frecuencia en el eje "*y*" (ordenadas), mientras que en el caso del cromagrama se muestran las notas en dicho eje.

Hay diferentes formas de convertir una grabación de audio en un cromagrama. Una de ellas puede llevarse a cabo utilizando transformadas de Fourier de tiempo corto en combinación con estrategias de "*binnin*g" o empleando bancos de filtros multietapa adecuados. Además, las propiedades de las características del croma pueden cambiarse significativamente introduciendo pasos adecuados de pre y post-procesamiento que modifiquen aspectos espectrales, temporales y dinámicos. Esto da lugar a un gran número de variantes de croma, que pueden mostrar un comportamiento muy diferente en el contexto de un escenario específico de análisis musical.

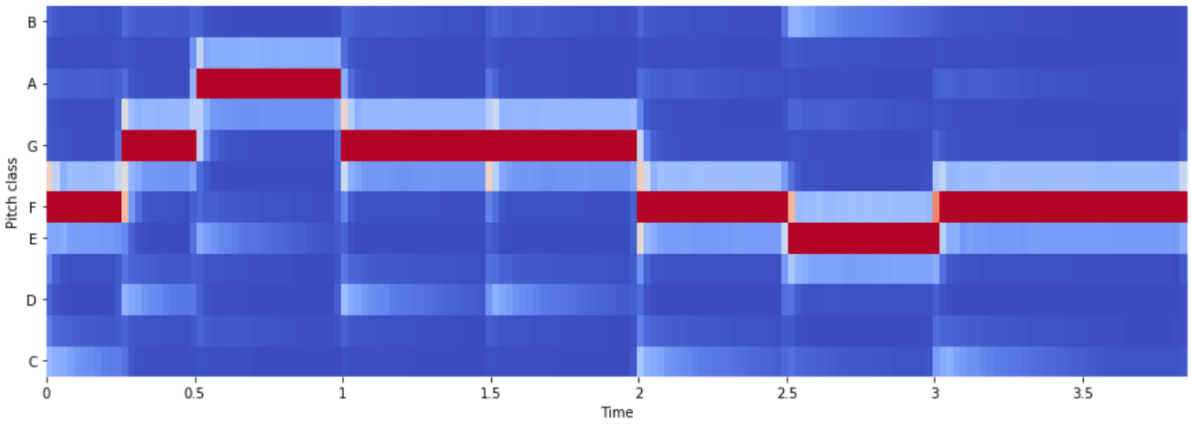

Figura A1.3. Cromagrama. Fuente: (*musicinformationretrieval*, 2021)

2. Representación Simbólica

2.1 Partitura

Es un tipo de notación, en forma impresa o manuscrita, de una obra musical en la que se detalla cómo se debe interpretar una pieza de música. Las partituras tienen un lenguaje especial conocido como notación musical que está formado por símbolos musicales. Seguidamente, de entre todos ellos se destacan los principales:

Pentagrama:

Son un conjunto de 5 líneas horizontales, separadas por 4 espacios, donde se colocan las notas musicales. Por encima y por debajo de él, se pueden añadir pequeñas líneas horizontales denominadas líneas adicionales, en las que se colocan las notas que no pueden incluirse dentro del pentagrama.

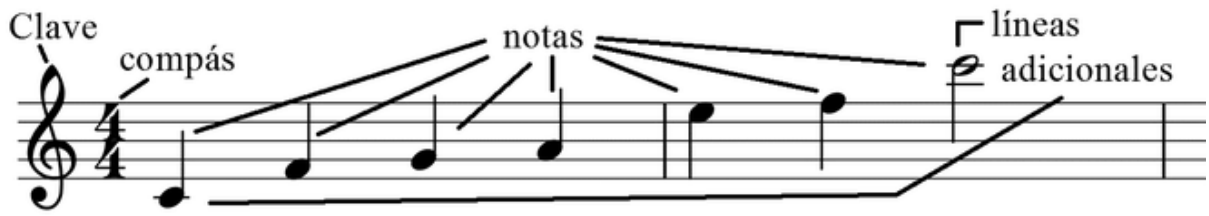

Figura A1.4. Pentagrama y líneas adicionales. Fuente: (creandopartituras, 2012)

Notas:

La nota es un símbolo que denota un sonido musical y que representa el tono y la duración de un sonido en la notación musical. Son, por tanto, los elementos fundamentales de la música a partir de los cuales se pueden representar cualquier pieza musical.

Las notas musicales son: Do / Re / Mi / Fa / Sol / La / Si.

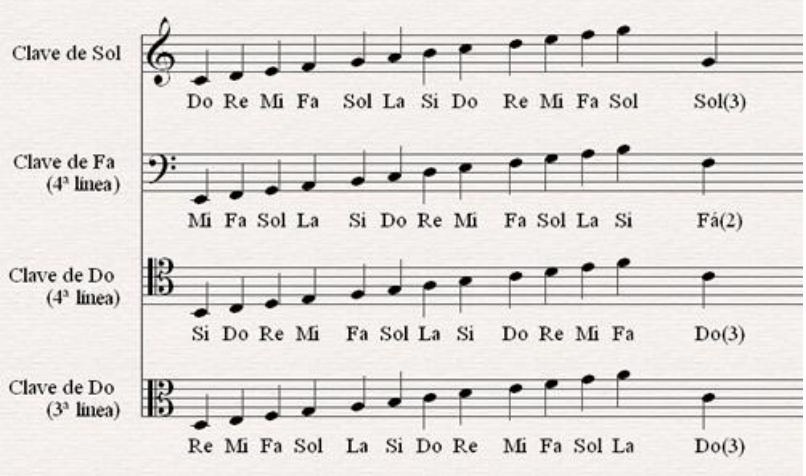

Figura A1.5. Notas musicales en las diferentes claves. Fuente: (Wikipedia, 2021)

Claves musicales:

Indican donde situar cada nota dentro del pentagrama. Existen 7 claves diferentes, siendo la más utilizada la clave de Sol, que se representa en la segunda línea del pentagrama.

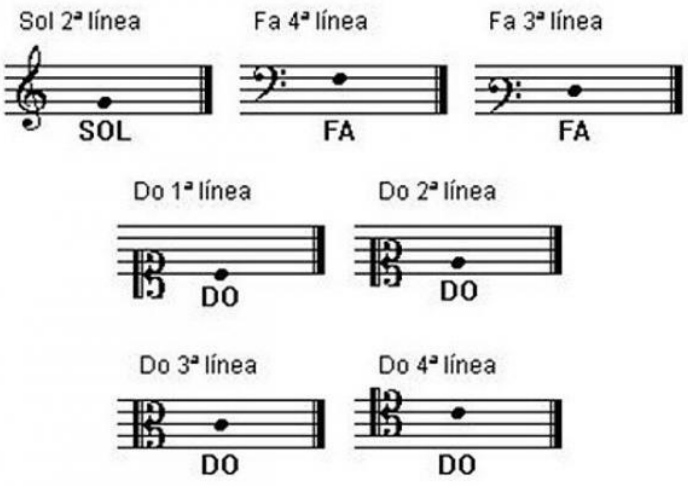

Figura A1.6. Claves musicales. Fuente: (Rivera, 2019)

Figuras y silencios:

Representan la duración de cada sonido musical y del silencio. Cada sonido tiene asociada una nota musical y una duración, del mismo modo que el silencio tiene asociado un símbolo dependiendo de su duración.

NOMBRE	FIGURA	SILENCIO	VALOR/PULSOS
Redonda	𝅝	𝄻	4 Tiempos
Blanca	𝅗𝅥	𝄼	2 Tiempos
Negra	♩	𝄽	1 Tiempo
Corchea	♪	𝄾	1 / 2 Tiempo
Semicorchea	𝅘𝅥𝅯	𝄿	1 / 4 Tiempo
Fusa	𝅘𝅥𝅰	𝅀	1/8 Tiempo
Semifusa	𝅘𝅥𝅱	𝅁	1/16 Tiempo

Figura A1.7. Notas musicales, silencios y valor. Fuente: (Musicateoria, 2011b)

La notación musical se basa en proporciones, de este modo se puede establecer una relación de proporcionalidad entre las diferentes notas en la notación musical.

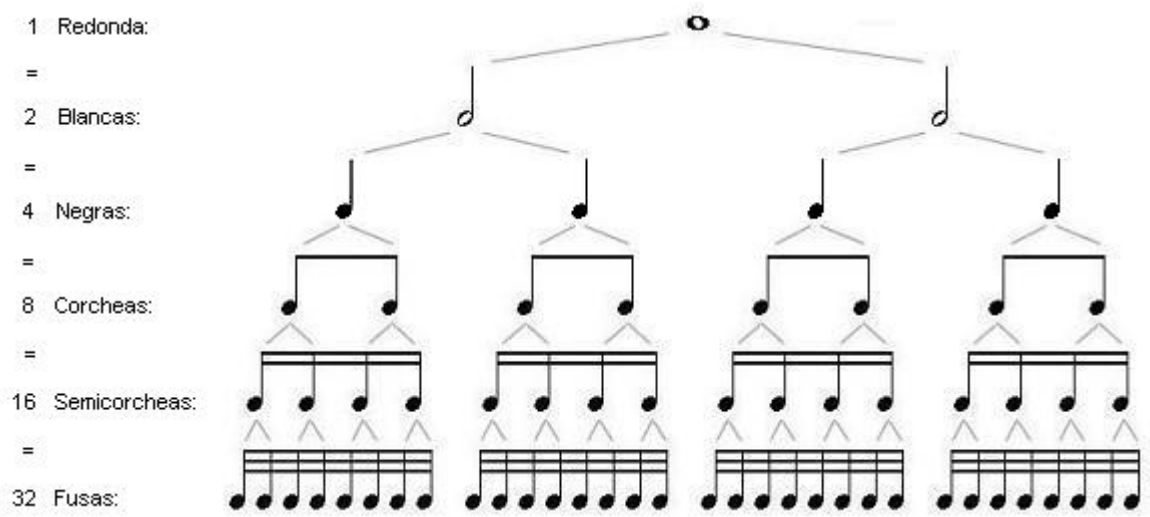

Figura A1.8. Relaciones entre notas musicales. Fuente: (Musicateoria, 2011a)

Compases:

La organización del ritmo de la música se realiza gracias al compás. Se representa al comienzo de la partitura con una figura formada por dos números. Existen diferentes tipos de

compases musicales, pudiendo ser binarios, ternarios o irregulares, por ejemplo: dos por cuatro, tres por cuatro, cuatro por cuatro, seis por ocho, etc.

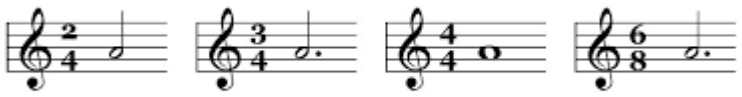

Figura A1.9. Tipos de compases musicales. Fuente: (Eligetuviolin, 2018)

El compás sirve para acentuar la melodía, es como la "*tilde*" ortográfica, pero que son invisibles en la música. Además, indica dónde poner el acento para cada estilo de música. Por ejemplo, un Pasodoble tiene un compás de 2/4 (2 por 4) lo que implica acentuar en el primer tiempo del compás y menos el segundo.

Si asociamos el compás con una fracción:

- El denominador se asocia a la figura, corchea, semicorchea, etc.:
    - 2 = Notas blancas.
    - 4 = Notas negras.
    - 8 = Notas corcheas.
    - 16= Notas semicorcheas.
    - 32 = Notas fusas.
    - 64 = Notas semifusas.
- El numerador se asocia al número de esas figuras que va a haber en cada uno de los compases:
    - 2 / 2 = 2 notas blancas por cada compás o un conjunto de notas cuya duración sea equivalente.
    - 3 / 4= 3 notas negras por cada compás o un conjunto de notas cuya duración sea equivalente.
    - 6 / 8 = 6 notas corcheas por cada compás o un conjunto de notas cuya duración sea equivalente.

2.2 MIDI

Son las siglas de "*Musical Instrument Digital Interface*". El standard MIDI permite la conexión y la comunicación entre diferentes ordenadores, instrumentos musicales electrónicos y otros dispositivos, mediante el uso de un protocolo específico.

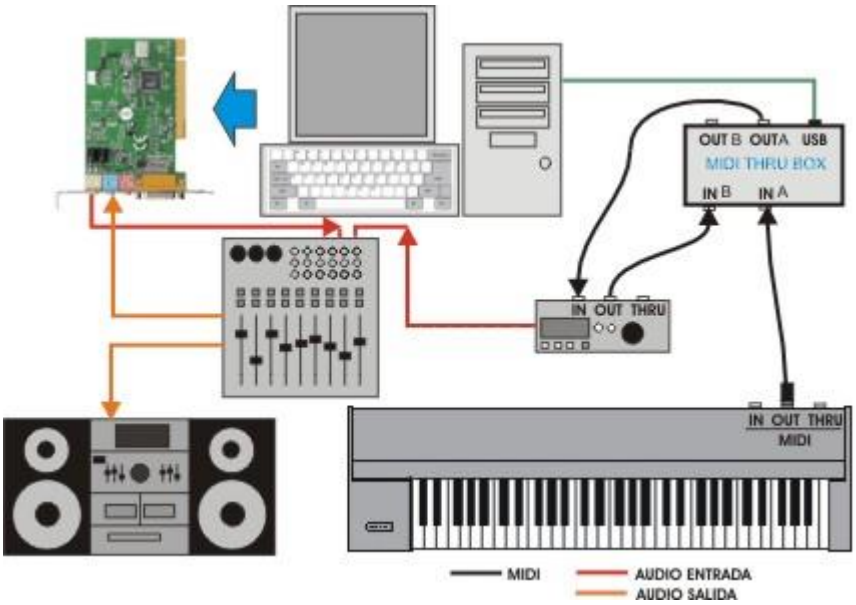

Figura A1.10. Esquema de conexiones entre diferentes equipos. Fuente: (El-atril, 2021)

El standard MIDI tienen una estructura basada en sucesiones de segmentos. Cada fichero MIDI consta de un segmento de cabecera y a continuación un número variable de segmentos de pistas.

El protocolo se basa en:

- El segmento de cabecera que contiene la información sobre:
    - Número de pistas o tracks.
    - Tiempo.
    - Formato SMF.
        - Tipo 0: Indica que toda la información se musical incluye en una única pista o track.

  - Tipo I: Indica que corresponde a multipista-síncrono. En este caso el fichero MIDI contienen la información de las diferentes pistas que se van a tocar de forma sincronizada.
  - Tipo II: No suelen utilizarse.

- Segmento de pista que consta de:
  - Cabecera.
  - Mensajes de eventos MIDI que pueden contener información de la nota que se debe tocar (tono, intensidad, etc.).
  - Eventos de control o sincronización.

De los mensajes de eventos más importantes, atendiendo a la información que contienen son:

- Note-on: Indica la nota que se está tocando. Contiene información sobre:
  - N.º de canal: indica el instrumento o la pista. Es un valor numérico comprendido entre 0 y 15.
  - N.º de nota MIDI: Indica el tono de la nota. Es un número comprendido entre 0 y 127 (7 bits).

    La relación entre notas musicales y su equivalente MIDI son las siguientes:

Tabla A1.1. Relación entre notas musicales y su equivalente MIDI

Octava	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
0	0	1	2	3	4	5	6	7	8	9	10	11
1	12	13	14	15	16	17	18	19	20	21	22	23
2	24	25	26	27	28	29	30	31	32	33	34	35
3	36	37	38	39	40	41	42	43	44	45	46	47
4	48	49	50	51	52	53	54	55	56	57	58	59
5	60	61	62	63	64	65	66	67	68	69	70	71
6	72	73	74	75	76	77	78	79	80	81	82	83
7	84	85	86	87	88	89	90	91	92	93	94	95
8	96	97	98	99	100	101	102	103	104	105	106	107
9	108	109	110	111	112	113	114	115	116	117	118	119
10	120	121	122	123	124	125	126	127				

    - Velocidad: Indica como de fuerte se toca una nota. Para un teclado quiere decir la velocidad con la que se pulsa una tecla y que corresponde con el volumen de esa nota. Toma valores enteros entre 0 y 127.

- Note off: Indica la finalización de una nota. En este caso, la velocidad corresponde a la rapidez con que la nota es soltada.

Los mensajes de eventos asociados a notas se embeben dentro de fragmentos de pistas junto con un campo delta-time que porta información relativa al tiempo para saber en qué instante debe tocarse cada nota.

2.3 Piano Roll

Es un formato basado en los pianos automáticos y que permite tanto la representación de melodías monofónicas como polifónicas. Físicamente, es un rollo continuo de papel con perforaciones, cada una de las cuales representa la pulsación o ejecución de una nota musical. La longitud de la perforación correspondería a la duración de la nota que previamente se pulsó. Este mecanismo es similar al castizo organillo que todavía se toca en algunas plazas madrileñas para reclamo turístico.

La representación musical mediante piano roll es una de las más utilizadas, aunque tenga ciertas limitaciones. Una de ellas, si la comparamos con la representación MIDI, sería la falta de información en los eventos correspondientes a note-off (liberación de la nota pulsada), esto tiene como consecuencia que no se pueda distinguir entre una nota de larga duración y varias notas tocadas repetidamente.

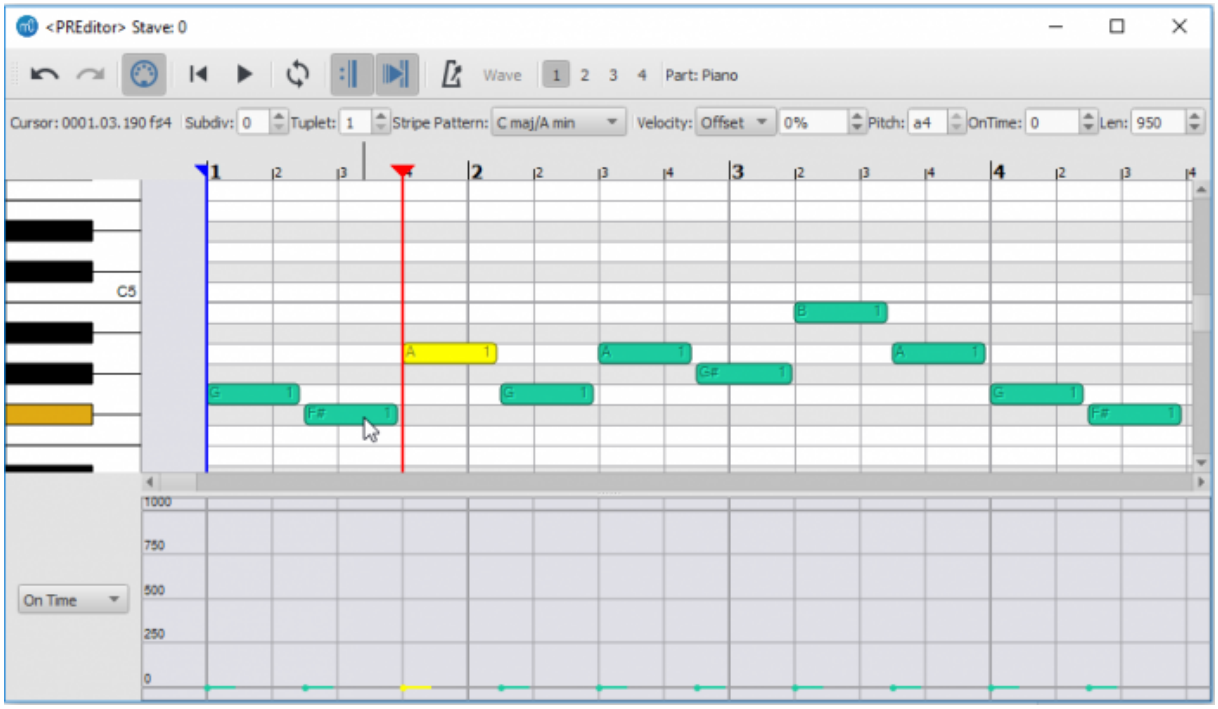

Figura A1.11. Notación piano roll del programa MuseScore. Fuente: (Musescore Org., 2021)

2.4 Texto

Una pieza musical puede codificarse en un formato "*textual*" y procesada como texto. Un ejemplo de este tipo de representaciones es la notación ABC que se ha convertido en un standard para la música folk y tradicional americana.

- Las primeras 6 líneas corresponden a los metadatos:
    - T: Es el título de la pieza.
    - M: Es el compás.
    - L: Es la longitud por defecto de la nota.
    - K: Es la clave.
- Después de esta cabecera, se encuentra la melodía que se codifica utilizando la notación inglesa (A=La, C=Do, D=Re, etc.).

Uno de los inconvenientes de la notación ABC es que solo pueden representarse melodías monofónicas.

```
X:1
T:The Legacy Jig
M:6/8
L:1/8
R:jig
K:G
GFG BAB | gfg gab | GFG BAB | d2A AFD |
GFG BAB | gfg gab | age edB |1 dBA AFD :|2 dBA ABd |:
efe edB | dBA ABd | efe edB | gdB ABd |
efe edB | d2d def | gfe edB |1 dBA ABd :|2 dBA AFD |]
```

Figura A1.12. Extracto de canción en notación ABC. Fuente: Elaboración propia

2.5 Markup Language

Son representaciones en formato HTML y XML. Algunos lenguajes de marcas han sido diseñados específicamente para que puedan usarse con aplicaciones musicales, como es el caso del open standard MusicXML (Makemusic, 2015), que fue desarrollado por Recordare LLC, en base a trabajos anteriores como el Humdrum de David Huron y el Musedata de Walter Hewlett. El objetivo que se persigue con este formato es facilitar el intercambio y almacenamiento de música por el software musical, como pueden ser los secuenciadores y los editores de partituras.

Esta representación no es muy adecuada para ser leída directamente por un músico y tampoco es apropiada para utilizarse como entrada a un algoritmo de Deep Learning.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE score-partwise PUBLIC
    "-//Recordare//DTD MusicXML 2.0 Partwise//EN"
    "http://www.musicxml.org/dtds/partwise.dtd">
<score-partwise version="2.0">
  <part-list>
    <score-part id="P1">
      <part-name>Music</part-name>
    </score-part>
  </part-list>
  <part id="P1">
    <measure number="1">
      <attributes>
        <divisions>1</divisions>
        <key>
          <fifths>0</fifths>
        </key>
        <time>
          <beats>4</beats>
          <beat-type>4</beat-type>
        </time>
        <clef>
          <sign>G</sign>
          <line>2</line>
        </clef>
      </attributes>
```

Figura A1.13. Fragmento de canción en MusicXML. Fuente: (Rothman, 2017)

2.6 Lead Sheet

Una Lead Sheet o Fake Sheet es tipo de notación musical que detalla los principales elementos de una pieza musical. Normalmente, se utiliza este tipo de representación en piezas populares de folk, pop, etc.

Consta de los siguientes elementos:

- La melodía se escribe en notación musical occidental moderna.
- La letra de la pieza musical se escribe en la parte de abajo del pentagrama.
- La armonía se coloca en la parte de arriba del pentagrama y para su representación se utiliza la codificación standard de los acordes.
- Opcionalmente, suele incluir información para el intérprete como puede ser el tempo y el estilo, así como autor y arreglista de la pieza.

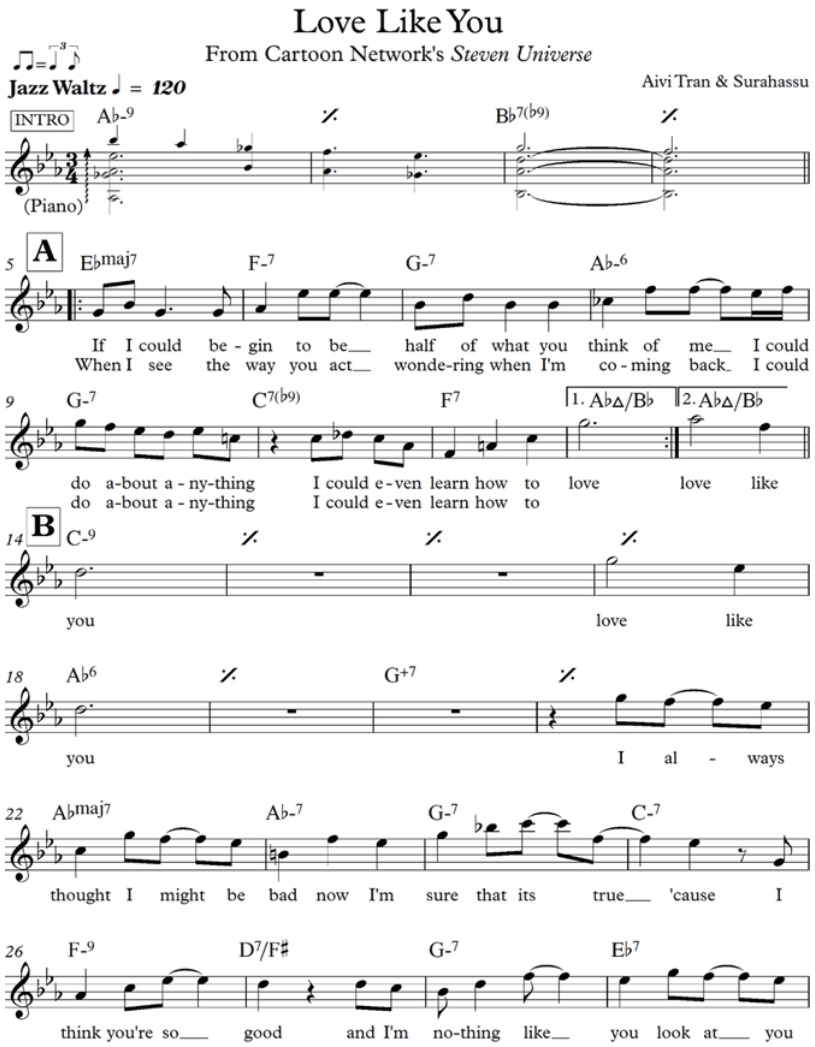

Figura A1.14. Lead Sheet. Fuente: (Bocardo, 2018)

3. Tipos de Codificación

En Aprendizaje Automático se trabajan con datos del mundo real. Esto implica que existen atributos como son los colores, los nombres de objetos, las notas musicales, o alguna clasificación específica que no son valores numéricos. Estos datos, y más concretamente su representación, afectan a los algoritmos de redes neuronales ya que éstos sólo usan valores numéricos con los que se hacen multitud de cálculos, por lo que todas las entradas que sean atributos se deben convertir a números para que puedan procesarse.

Las más utilizadas se indican a continuación:

3.1 Label Encoding

Este enfoque es sumamente simple y consiste en convertir cada etiqueta en un valor numérico. Generalmente, las etiquetas suelen ir en orden ascendente.

Tabla A1.2. Label encoding

CATEGORÍAS (Texto)	**LABEL ENCODING (Número)**
Categoría-1	0
Categoría-2	1
Categoría-3	2
…	…
Categoría-N	N - 1

El problema de usar números en la asignación de etiquetas es que se introducen relaciones y/o comparaciones entre las diferentes categorías que realmente no existen. Aunque en apariencia no hay relación alguna entre Categoría-1 y Categoría-2, podría intuirse que la Categoría-1 tiene mayor precedencia que la Categoría-2. Del mismo modo, se puede interpretar que existe algún tipo de jerarquía u orden entre ellas 0 < 1 < 2 ... < 6, incluso, podrían interpretarse relaciones del tipo Categoría-1 + Categoría-2 = Categoría-3.

3.2 One-Hot Encoding

En esta estrategia de codificación, cada valor de categoría se convierte en una nueva columna y se le asigna un valor de 1 o 0 (notación para verdadero/falso). Si consideramos el ejemplo anterior de categorías, la codificación quedaría como se indica en la siguiente tabla:

Tabla A1.3. One-Hot Encoding

CATEGORÍA	Cat-1	Cat-2	Cat-3	…	Cat-N
Categoría-1	1	0	0	…	0
Categoría-2	0	1	0	…	0
Categoría-3	0	0	1	…	0
…				…	
Categoría-N	0	0	0	…	1

Aunque la codificación One-Hot ya no plantea los problemas de relación, orden y jerarquía expuestos anteriormente en la codificación Label, tiene el inconveniente de añadir muchas más columnas al conjunto de datos, de hecho, puede hacer que el número de columnas se amplíe enormemente si existen muchos valores diferentes en una categoría, pudiendo hacer que el conjunto sea difícilmente manejable y entrenable por un algoritmo clasificador. Concretamente, el número de columnas que se procesan es igual al número de elementos diferentes que estemos utilizando. En el caso de tratamiento de textos, correspondería al tamaño del vocabulario utilizado o en el caso de tratamiento de partituras serían las diferentes notas musicales para las diferentes octavas.

Para incluir los elementos en un algoritmo de Machine Learning mediante codificación One-Hot se utiliza, por regla general, un vector de N elementos, donde N es el número de categorías diferentes de ese atributo. De este modo, se crea un vector por cada categoría diferente que tiene todas las posiciones establecidas a 0 excepto aquella asociada a la categoría que se esté representado, la cual se fija a 1.

Este tipo de representación de los atributos es utilizado frecuentemente en el tratamiento de textos, lenguaje y generación musical.

Anexo II. Música Flamenca

Según los estudios de Rocío Carrera (Carrera Portero, 2016), el flamenco es un estilo de baile y música típico de las regiones de Andalucía, Murcia y Extremadura. Las principales líneas de expresión de este arte son el baile, el toque (generalmente guitarra) y el cante. Como sabemos hoy, su historia se remonta al siglo XVIII, y aún existe debate sobre su origen, porque, aunque existen diferentes teorías, ninguna de ellas se ha podido probar de forma rigurosa y fehaciente. Aunque el flamenco se ha asociado a la cultura de la etnia gitana, quizá por sus aportaciones a este género a lo largo del tiempo, no ha sido la única, ya que esta expresión artística se ha visto influenciada por otras culturas como la africana, caribeña, además de la árabe, judía y cristiana.

De todas las hipótesis existentes sobre el origen del flamenco, la más extendida es la que propone el origen morisco, que junto al resto de culturas que se dieron en Andalucía en sus orígenes (musulmana, gitana, judía, etc.), hicieron posible el nacimiento de esta expresión artística.

Elemento Básicos de la música flamenca:

Según Rocío Carrera (Carrera Portero, 2016), los elementos básicos que hacen que la música flamenca sea diferente de otras músicas o estilos son:

- La complejidad de los ritmos: Entre los que se destacan el compás de Seguiriya, el compás de Soleá, el compás de Tangos flamencos, etc.
- El ritmo: es el elemento fundamental que da sentido y personalidad al arte flamenco. Existen diferentes tipos de ritmos entre los que se destacan:
    - Los ritmos simples: Pueden ser binarios, ternarios, etc. Rumba, Sevillanas, etc.
    - Ritmos compuestos: Se forman mediante la alternancia de 2 ritmos como en el caso de Seguiriya.
    - Sin ritmo: Se utilizan en los estilos libres como la Taranta o la Granaina.
- El sonido especial de la guitarra que se utiliza en el acompañamiento del cante y del baile. Las particularidades del cante flamenco y la forma en la que se bailan cada uno de los palos flamencos.
- El sentimiento, la pasión y el significado que tiene el flamenco hace que sea una seña de identidad cultural. Como dijo Antonio Gades, "*un extracto de fuego y de veneno, eso es el flamenco*".

- La gran variedad de palos flamencos o estilos y sus relaciones e influencias rítmicas y melódicas entre ellos.

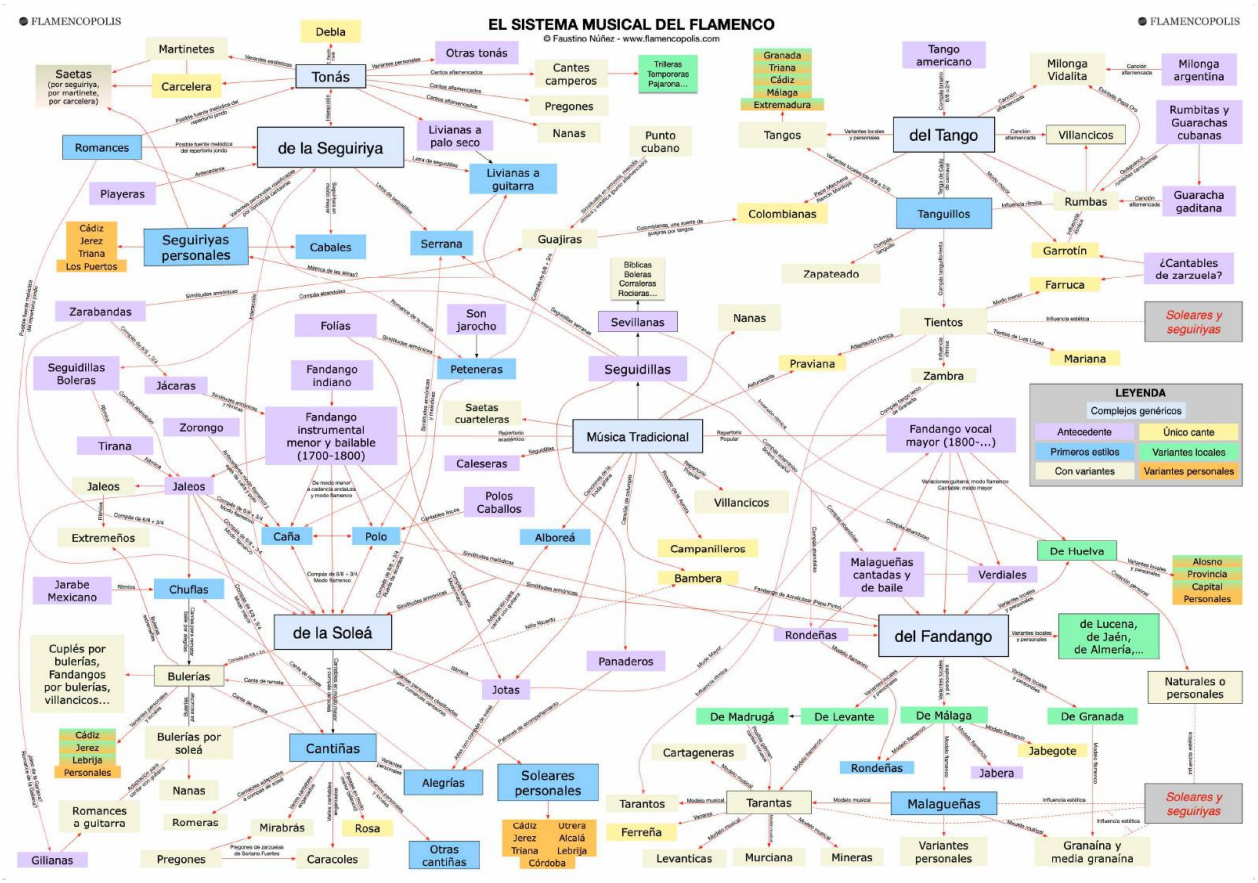

Figura A2.1. Sistema musical flamenco: Fuente: (Nuñez, 2011)

Anexo III. Código Experimentos

1. Experimento 1

En este apartado se detalla el código utilizado para el entrenamiento de la red neuronal del modelo "*Melody_rnn*" con configuración "*Basic_rnn*" y las partituras generadas durante el experimento 1 para las diferentes temperaturas:

- El código fuente realizado para este experimento se encuentra en:

  https://github.com/joseluistapiav/Experimentos_TFM_IA/tree/master/Experimento_1_Melody_rnn_Basic_rnn/Codigo_Python

- Las partituras generadas en este experimento para las diferentes temperaturas se encuentran en:

  https://github.com/joseluistapiav/Experimentos_TFM_IA/tree/master/Experimento_1_Melody_rnn_Basic_rnn/Partituras_Generadas

2. Experimento 2

En este apartado se detalla el código utilizado para el entrenamiento de la red neuronal del modelo "*Melody_rnn*" con configuración "*Attention_rnn*" y las partituras generadas durante el experimento 2 para las diferentes temperaturas:

- El código fuente realizado para este experimento se encuentra en:

  https://github.com/joseluistapiav/Experimentos_TFM_IA/tree/master/Experimento_2_Melody_rnn_Attention_rnn/Codigo_Python

- Las partituras generadas en este experimento para las diferentes temperaturas se encuentran en:

  https://github.com/joseluistapiav/Experimentos_TFM_IA/tree/master/Experimento_2_Melody_rnn_Attention_rnn/Partituras_Generadas

3. Experimento 3

En este apartado se detalla el código utilizado para el entrenamiento de la red neuronal LSTM desarrollada específicamente para este experimento y las partituras generadas para las diferentes temperaturas:

- El código fuente realizado para este experimento se encuentra en:

  https://github.com/joseluistapiav/Experimentos_TFM_IA/tree/master/Experimento_3_LSTM/Codigo_Python

- Las partituras generadas en este experimento para las diferentes temperaturas se encuentran en:

  https://github.com/joseluistapiav/Experimentos_TFM_IA/tree/master/Experimento_3_LSTM/Partituras_Generadas

4. Experimento 4

En este apartado se detalla el código utilizado para el entrenamiento de la red neuronal LSTM con mecanismo de "*Atención*" desarrollada específicamente para este experimento y las partituras generadas para las diferentes temperaturas:

- El código fuente realizado para este experimento se encuentra en:

  https://github.com/joseluistapiav/Experimentos_TFM_IA/tree/master/Experimento_4_LSTM_Attention/Codigo_Python

- Las partituras generadas en este experimento para las diferentes temperaturas se encuentran en:

  https://github.com/joseluistapiav/Experimentos_TFM_IA/tree/master/Experimento_4_LSTM_Attention/Partituras_Generadas

Anexo IV. Partituras Flamencas en Soundcloud

En este Anexo se incluyen varios ejemplos generados durante la ejecución de los experimentos para diferentes temperaturas:

- Ejemplos de música flamenca generados durante el experimento 3 para temperaturas 0,5 y 1, respectivamente:

  https://soundcloud.com/user-460041748/solea-lstm-temp-05

  https://soundcloud.com/user-460041748/solea-lstm-temp-10

- Ejemplos de música flamenca generados durante el experimento 4 para temperaturas 0,5 y 1, respectivamente:

  https://soundcloud.com/user-460041748/solea-lstm-attention-temp-05

  https://soundcloud.com/user-460041748/solea-lstm-attention-temp-10

Anexo V. Artículo de Investigación

Composición de Música Flamenca para Guitarra mediante Técnicas de Inteligencia Artificial

José Luis Tapia Velázquez

Universidad Internacional de la Rioja, Logroño (España)

Fechas: 22 de julio de 2021

Resumen

La generación de contenidos musicales a partir de información contenida en datasets perteneciente a uno o varios estilos mediante el uso de técnicas de Deep Learning es un campo de estudio dentro de la Inteligencia Artificial que está en pleno auge.

Los algoritmos y frameworks existentes en la actualidad son capaces de generar música de diferentes estilos entre los que se podrían incluir los diferentes palos flamencos, pero no tienen en cuenta las particularidades rítmicas y melódicas propias de estos, principalmente porque solo utilizan como entrada las melodías y, en algunos casos, los acordes que la acompañan.

Con el fin de poder componer música flamenca del estilo de la Soleá, se ha incluido el compás como un elemento más de entrada para que la música que se genere siga el ritmo propio de la Soleá. Adicionalmente, se utilizarán mecanismos de temperatura variable para conseguir aleatoriedad musical dentro del estilo flamenco sin perder las características melódicas propias de la Soleá.

Palabras Clave

Inteligencia Artificial, IA, composición musical, flamenco.

I. Introducción

El flamenco tiene particularidades melódicas y rítmicas que lo hacen muy especial y diferente de otros estilos musicales.

Una de estas particularidades es el "*compás*" que tienen los estilos flamencos, también denominados "*palos*", que hace que la melodía siempre siga un patrón rítmico del cual no es posible salirse. Esto implica que la melodía comience en un determinado compás, fluya y varíe durante otros compases y se cierre o finalice exactamente en el mismo compás.

Esta rítmica tan especial, no es tratada por los algoritmos de generación musical que solo utilizan como entrada las melodías de un estilo o de diferentes estilos y, en algunos casos, los acordes que la acompañan.

Con el fin de aportar la rítmica que necesitan los algoritmos para generar música de estilo flamenco, se plantea el uso de información adicional. Básicamente, consiste en añadir a la información, existente en los dataset de entrada, los datos de la rítmica del palo flamenco. Otra forma, puede ser la utilización de mecanismos dentro del ámbito de la Inteligencia Artificial para que el sistema de generación de composiciones musicales aprenda las particularidades de forma no supervisada.

Plantear un sistema de composición de música flamenca para el estilo de la Soleá, además de ser un reto, puede significar un avance en la creación de contenidos flamencos y una herramienta de ayuda para los compositores y para los lo que no conocen esta música tan singular.

II. Estado del Arte

Con la llegada del Deep Learning al mundo de la Inteligencia Artificial, se han propuesto nuevos tipos de arquitecturas y los trabajos realizados sobre la generación de música se han beneficiado de la era del Big Data caracterizada por la gran capacidad de procesamiento de las máquinas y la gran disponibilidad de datos, lo que ha permitido realizar experimentos a gran escala.

A continuación, se indican las principales arquitecturas utilizadas en Deep Learning para generar contenidos musicales:

Feedforward:

Las redes neuronales con esta topología se caracterizan por ejecutar el procesamiento de datos en una sola dirección. Este tipo de redes constan de tres capas (layers) principales de unidades independientes de cómputo llamadas neuronas:

1. Capa de entrada (input layer): Donde se reciben los datos que van a procesarse.

2. Capa intermedia (hidden layer): Donde se realiza el procesamiento propiamente dicho.

3. Capa de salida (output layer): Donde se obtienen los valores generados.

Las redes neuronales artificiales tipo feedforward (RNAf) son fáciles de programar, por lo que son muy populares entre la comunidad científica.

Si bien este tipo de redes no se utiliza de forma aislada para la generación de contenidos musicales, sí se utilizan dentro de arquitecturas más complejas como es el caso de DeepBatch [1] que combina 2 redes LSTM y 2 redes feedforward para generar música coral del estilo de Bach.

CNN

Las redes convolucionales (CNN) son un tipo de red neuronal propuestas por Yann André Lecun [2] que aplico este tipo de arquitectura al reconocimiento de dígitos escritos manualmente.

Posteriormente este tipo de redes se aplicaron en el reconocimiento de imágenes y se hicieron muy populares tras ganar AlexNet en 2012 la competición de clasificación de imágenes ImageNet (ImageNet Large Scale Visual Recognition Challenge). Hoy en día son ampliamente utilizadas en diferentes campos de la Inteligencia Artificial entre el que destaca la visión por computador (detección de objetos, segmentación, etc.).

También se han utilizado para la generación de música como es el caso de WaveNet [3]. Este modelo tenía como entrada y salida audio en crudo (raw audio) y, aunque generaba música, tenía el inconveniente de que ésta variaba cada poco tiempo en su estilo musical, el volumen de la música, la calidad, los instrumentos, etc.

RNN

Una red neuronal recurrente (RNN) es un tipo de red neuronal que utiliza datos secuenciales o de series temporales. Su particularidad y elemento característico es su memoria interna o "*hidden state*" que le permite persistir lo ocurrido anteriormente y utilizarlo para influir en la salida actual.

Si comparamos las RNN con las redes neuronales anteriores (feedforward, CNN, etc.) se puede ver que, en las primeras, la salida depende de los elementos anteriores dentro de la secuencia, mientras que, en las segundas no existe relación entre las entradas y salidas (son independientes entre sí).

Uno de los primeros de proyectos de composición musical que utilizaban las RNN fue el sistema CONCERT [4]. Uno de los hitos que se consiguieron con la creación de este sistema fue demostrar que conseguía aprender la estructura subyacente de los datos de entrenamiento, lo que le permitía componer nuevas melodías. Aunque estas composiciones eran en ocasiones agradables adolecían de falta de coherencia global.

Otros trabajos, como los realizados por Gaetan Hadjeres and Frank Nielsen [1], donde presentan una arquitectura novedosa que denominaron "*Anticipation-RNN*" que tiene las ventajas de las arquitecturas RNN, al tiempo que permite aplicar las restricciones posicionales definidas por el usuario. Utilizaron esta arquitectura para generar piezas musicales al estilo de Bach que cumplieran las restricciones de posición en las partes de soprano.

LSTM:

Las LSTM (Long Short Term Memory) son un tipo especial de RNN que fueron introducidas por Hochreiter & Schmidhuber [5] y refinadas por muchos investigadores en trabajos posteriores. Esta topología de red es capaz de aprender y memorizar las relaciones entre los datos a largo plazo.

Las LSTM están pensadas para evitar el problema de la dependencia a largo plazo. Se puede afirmar que este tipo de redes fueron diseñadas para que su comportamiento por defecto fuera almacenar y recordar información durante largos periodos de tiempo. Esto permite resolver el problema del "exploding/vanishing gradient" que sufren las RNN y que dificulta el aprendizaje de secuencias largas de datos.

Las puertas que componen las redes LSTM son:

• Puerta de olvido (forget gate): Decide qué información necesita atención y cuál puede ignorarse, o lo que es lo mismo, que información se mantiene y cual se olvida.

• Puerta de entrada (input gate): Decide qué información es relevante para actualizar en el estado actual de la célula de la unidad LSTM.

• Puerta de salida (output gate): Determina el estado oculto actual que pasará a la siguiente unidad LSTM.

Las redes LSTM se utilizan en diversas áreas de la Inteligencia Artificial, entre los que destacan la generación de contenidos musicales. De los numerosos estudios y proyectos que se realizaron hay que destacar las investigaciones de Douglas Eck y Jurgen Schmidhuber sobre la composición musical mediante redes LSTM [6] en las que destacaban que la música compuesta por las RNN adolece de una falta de estructura global ya que no pueden hacer un seguimiento de los eventos temporalmente distantes que indican la estructura musical general. Sin embargo, demostraron que las LSTM aprenden con éxito y son capaces de componer melodías nuevas en el estilo en el que se han entrenado. En los experimentos que se realizaron se utilizaron datasets de música de blues de 12 compases y sorprendentemente, una vez que la red encontraba la estructura relevante, no se desviaba de ella. Es más, la LSTM era capaz de generar música blues con buen ritmo y una estructura adecuada.

Otro ejemplo de aplicación de LSTM a la generación musical es el proyecto "*Music transcription modelling and composition*" [7] donde se aplicaron métodos de Deep Learning, especialmente redes LSTM, al modelado y la composición de transcripciones musicales. Se utilizaron unas 23.000 transcripciones de música celta expresadas en notación ABC para entrenar las redes LSTM de 3 capas con 512 neuronas por capa, con el fin de generar nuevas composiciones musicales.

Otros trabajos de interés en composición musical fueron los realizados por Aran Nayebi y Matt Vitelli y recogidos en "*Algorithmic Music Generation using Recurrent Neural Networks*" [8]. Estos estudios se centraban en la comparación del rendimiento entre GRU y LSTM para la tarea de generación algorítmica de música que utilizan como entrada ondas de audio. Los resultados indicaron que las salidas generadas por la red LSTM eran significativamente más plausibles desde el punto de vista musical que las de la GRU.

Existen diferentes trabajos donde se utilizan este tipo de redes aplicadas a la composición musical, entre ellos se pueden destacar "*Chord generation from symbolic melody using BLSTM networks*" [9], en el que se presenta un método novedoso para generar secuencias de acordes a partir de una melodía simbólica utilizando una arquitectura de red BLSTM con 2 capas y 128 neuronas cada una. La red se entrenó en una base a un dataset de más de 2.000 de partituras de música moderna (jazz, rock, etc.).

Redes GRU:

Las GRU (Gated Recurrent Unit) son un tipo especial de RNN que utilizan un sistema similar de puertas (gates) parecido al descrito anteriormente para las LSTM. Las mayores diferencias con las LSTM son que se combina el "*cell state*" y el "*hidden state*" en un solo elemento, así como la "*forget gate*" y la "*input gate*" en una única puerta.

A diferencia de la red LTSM descrita anteriormente y que consta de 3 puertas, la red GRU contiene únicamente dos puertas que controlan y deciden qué información de entrada debe pasarse a la salida:

• Puerta de actualización (update gate): Ayuda a determinar la cantidad de información pasada que debe transmitirse al futuro.

• Puerta de reinicio (reset gate): Es responsable de la memoria a corto plazo de la red. Es decir, permite al modelo decidir la cantidad de información pasada que se debe olvidar y cual debe persistir.

Existen diferentes estudios acerca de este tipo de redes y sus aplicaciones a la generación de música entre los que cabe destacar el "*Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*" [10] en el que se comparan diferentes tipos de RNN en las tareas de modelado de música polifónica y de señales de voz. Estos trabajos revelaron que las GRU eran mejores que las RNN más tradicionales, como la tanh y prácticamente iguales a las LSTM.

GAN

Las Redes Generativas Adversarias (GAN) son un tipo de redes neuronales dentro de la categoría de Deep Learning.

Una red generativa adversarial (GAN) tiene dos partes:

• El generador: Aprende a generar datos plausibles. De este modo, los datos generados se convierten en ejemplos negativos de entrenamiento para el discriminador.

• El discriminador: Aprende a distinguir los datos falsos del generador de los datos reales. El discriminador penaliza al generador por producir resultados inverosímiles

Este tipo de redes se han utilizado en trabajos como "*MidiNet*" [11] donde se han utilizado redes CNN para la generación de melodías (en formato MIDI). Además del generador, se utiliza un discriminador para aprender las distribuciones de las melodías, lo que la convierte en una red generativa adversarial (GAN). Adicionalmente, se proponen mecanismos condicionales para explotar el conocimiento previo, de modo que el modelo puede generar melodías desde cero, siguiendo una secuencia de acordes, o condicionando la melodía de los compases anteriores (por ejemplo, una melodía inicial de cebado), entre otras posibilidades. El modelo resultante puede ampliarse para generar música con múltiples canales MIDI (varias pistas).

Otro ejemplo interesante es el sistema "*Musegan*" [12]. En este sistema se proponen tres modelos para la generación simbólica de música multipista en el marco de las redes generativas adversarias (GAN). Los tres modelos, que difieren en los supuestos subyacentes y, en consecuencia, en las arquitecturas de red, se denominan modelo de interferencia, modelo de compositor y modelo híbrido. Los tres modelos propuestos se entrenaros con un dataset de más de cien mil compases de música rock con el fin de generar piano-rolls de cinco pistas (bajo, batería, guitarra, piano y cuerdas).

RL

El Aprendizaje por Refuerzo (RL) es un tipo de técnica de aprendizaje automático que permite a un agente aprender en un entorno interactivo por ensayo y error utilizando el feedback que proporciona sus acciones.

Las áreas de utilización del RL son aquellas que requieran muchos datos, como los juegos y la robótica. Otras aplicaciones del RL incluyen motores de resumen de texto, agentes de diálogo (texto, voz) y generación de contenidos musicales.

Este tipo de arquitectura se utiliza en el sistema "*Bach2Bach*" [13] donde se aplica una arquitectura de RL que predice y genera música polifónica alineada con las reglas musicales con las que se ha entrenado. El modelo está basado en una red LSTM entrenada con un "*núcleo*" similar a un kernel convolucional. Para fomentar la exploración e imponer una mayor coherencia global a la música generada, se adopta un enfoque de Deep Learning por refuerzo (DQN). Los resultados de este sistema cuando se analizan cuantitativa y cualitativamente muestran que este enfoque utilizando RL funciona bien en la composición de música polifónica.

Auto-Encoder (AE)

Es un sistema de Aprendizaje no Supervisado en el que durante el entrenamiento la salida esperada es una aproximación de la entrada. El Auto-Encoder se aplica principalmente a la reducción de la dimensionalidad de los datos, a la clasificación y a la eliminación de ruido de las imágenes, así como a la detección de objetos.

Variational Auto-Encoders (VAE)

Son modelos de Deep Learning que mezclan las redes neuronales con las distribuciones de probabilidad y que han tenido un gran éxito en muchas aplicaciones como la generación de imágenes, el subtitulado de imágenes, el diseño de proteínas, la predicción de mutaciones y los modelos de lenguaje, entre otros.

El objetivo de un VAE es aprender la distribución de los datos de entrenamiento para que, muestreando a partir de ella, se puedan generar nuevos datos. Dado que los datos de entrenamiento no tienen necesariamente una distribución matemática bien definida, se fuerza la distribución de la salida del codificador (conocida como espacio latente) para que siga una distribución conocida como, por ejemplo, una distribución normal.

Este tipo de topologías también se han utilizado también en la composición musical. En este sentido se puede señalar el estudio realizado por el equipo de Adams Roberts [14] en el que ponen de manifiesto que los VAE tienen dificultades para modelar secuencias con estructura a largo plazo. Para solventar esta problemática se propone el uso de un decodificador jerárquico, que primero produce "*Embeddings*" para subsecuencias de la entrada y luego utiliza estos "*Embeddings*" para generar cada subsecuencia de forma independiente. Esta estructura evita el problema del "*colapso posterior*", que sigue siendo un problema para los VAE recurrentes. Se aplico esta arquitectura a la modelización de secuencias de notas musicales y se descubrió que el rendimiento de muestreo, interpolación y reconstrucción es mucho mejor que el de un modelo básico plano.

Mecanismo de Atención

surgió dentro del área de PLN como una mejora del sistema de traducción automática mediante redes neuronales que estaba basado en el modelo Codificador-Decodificador. Posteriormente, este mecanismo, y sus diferentes evoluciones, se utilizaron, además del área de PLN, en otras aplicaciones, como la visión por ordenador, el procesamiento del habla y la generación de contenidos artísticos.

Este mecanismo puede considerarse "*bioinspirado*" desde el ámbito de estudio de la neurociencia cognitiva ya que estos mecanismos atencionales en humanos permiten al cerebro focalizarse en una zona concreta de una imagen de entrada y prestar menos atención a otras zonas de una imagen. Este mecanismo llevado a la composición musical permite que la red se centre en determinadas notas, compases o pasajes que son más representativos para la generación de secuencias musicales, obviando o prestando menor atención al resto.

En un principio la traducción automática mediante redes neuronales estaba basaba en RNN/LSTM codificadoras-decodificadoras con una arquitectura similar a esta:

• La RNN/LSTM codificadora procesa todas las palabras de la frase de entrada y la codifica en un vector de contexto. Este vector es el último estado oculto de la RNN/LSTM. En esta arquitectura todos los estados intermedios del codificador se ignoran y el estado final es el estado oculto inicial del que partirá el decodificador.

• Las unidades RNN/LSTM del decodificador producen las palabras de una frase, una tras otra.

Es decir, hay dos RNN/LSTM. La primera RNN/LSTM que denominamos "*codificador*", lee la frase de entrada y la resume en un vector de contexto. La segunda RNN/LSTM, pasa el resumen (vector de contexto) al "*decodificador*", que traduce la frase de entrada en función de los datos que ha recibido.

Entre los diferentes estudios en los que se utilizan mecanismos de "*Atención*" para la generación musical, se puede destacar el sistema "*Music Transformer*" [15] en el que utilizaron un "*Transformer*" con un mecanismo de "*Atención*" relativa modificado para generar composiciones de un minuto de duración.

Condicionamiento

Estas estrategias, también denominadas "*Arquitecturas Condicionadas*", consisten en incluir información adicional a la entrada de la red neuronal para condicionar el comportamiento del algoritmo y conseguir mejores resultados.

La información de "*Condicionamiento*" que se introduce al algoritmo puede ser:

• Las notas de un bajo o un patrón de rítmico.

• Una progresión de acordes.

• El género musical.

• El instrumento.

Un ejemplo de este tipo de estrategias es "*MidiNet*" [11], que como ya se vio anteriormente, se basa en redes GAN e incluye un mecanismo de "*Condicionamiento*" que permite añadir información histórica, tanto de la melodía como de los acordes de los compases anteriores.

"*Wavenet*" [3], descrita anteriormente, también hace uso de estrategias de "*Condicionamiento*" al incluirse información relevante del estilo musical y del instrumento que se utiliza para el audio en crudo (raw audio).

III. Objetivos y Metodología

El objetivo general de este trabajo es la realización de un conjunto de experimentos con distintas arquitecturas de Deep Learning con el fin de explorar diferentes técnicas de composición de música flamenca para guitarra del estilo de la Soleá.

Los experimentos utilizarán un conjunto de datos de entrenamiento consistentes en partituras y tablaturas para guitarra flamenca del palo de la Soleá.

Los algoritmos utilizados y entrenados deberán generar composiciones para este palo flamenco manteniendo el compás y tonalidad propias de este estilo.

Para el desarrollo de este artículo se establecen los siguientes objetivos específicos:

1. Añadir información adicional de entrada sobre el compás al algoritmo para aportar más riqueza al entrenamiento y generar música con la rítmica propia del estilo de la Soleá. Esto implica procesar las partituras MIDI que conforman el dataset de entrada con el fin de generar la información de compás que se introducirá al algoritmo junto con la información de las notas y su duración.

2. Crear mecanismos específicos de temperatura más eficaces que permitan la generación de música con la rítmica y tonalidad flamenca del estilo de la Soleá.

3. Analizar las diferentes estructuras de codificación de los datos que mejor se adapten para el entrenamiento. Es necesario explorar las diferentes posibilidades que van desde la codificación One-Hot utilizando un único vector de entrada con la información musical del estilo flamenco o múltiples vectores de entrada que aporten más información al algoritmo.

4. Explorar los diferentes algoritmos de Deep Learning (RNN, LSTM, LSTM bidireccionales, etc.) y sus diferentes implementaciones (Auto-Encoder, Atención, etc.) para conseguir generar música flamenca de Soleá que se ajuste al compás flamenco.

5. Utilizar el framework de Google Magenta para la generación de música flamenca del estilo de la Soleá. Esta biblioteca open source incluye utilidades para la manipulación de datos y para el entrenamiento de modelos de aprendizaje automático que pueden utilizarse para generar contenidos artísticos.

6. Crear un dataset de partituras en formato MIDI a partir de partituras y tablaturas de guitarra flamenca del estilo de la Soleá. Para ello, las diferentes piezas son pre-procesadas para eliminar las partes rítmicas. Seguidamente, son transformadas en monofónicas, antes de obtener el dataset final que sirve de entrada para todos los experimentos que se realizan. Los procesos de depuración, filtrado y procesamiento se detallan a lo largo de este estudio en puntos posteriores.

La metodología de trabajo que se ha seguido se indica a continuación:

1. Se realiza una búsqueda de las partituras flamencas del estilo de la Soleá, con el fin de crear un dataset suficientemente grande que sirva de entrada a los diferentes algoritmos que se van a utilizar en este trabajo para generar la música flamenca.

Todas las partituras y tablaturas recopiladas corresponden a piezas para guitarra flamenca del estilo de la Soleá.

El origen de estas partituras ha sido Internet, transcripciones en papel pautado, partituras del autor de este artículo.

Todas las partituras son para guitarra flamenca y por lo tanto son polifónicas.

Las partituras están transcritas y adaptadas para guitarra flamenca esto implica que las melodías están adaptadas a la digitación y a las particularidades propias del instrumento.

2. Todas las partes que componen las Soleares y que se encuentran escritas en las partituras que forman el dataset de entrada se han respetado, a excepción de las partes rítmicas que han sido eliminadas de todas ellas por no aportar riqueza melódica (siempre son las mismas variaciones) y por ser polifónicas.

Los compases con polifonía (varias notas tocadas simultáneamente) se han transformado en monofónicos intentando no alterar la estructura melódica de la música original.

3. Conversión de las partituras a formato MIDI para que puedan utilizarse como entrada a los diferentes experimentos que se van a desarrollar. Esta transformación se realiza con las funcionalidades proporcionadas por los programas Tuxguitar y MuseScore.

4. Dependiendo del tipo de experimento que se vaya a realizar, las partituras MIDI se podrán agrupar en lotes con el fin de que las piezas musicales sean más extensas. Esta unión de piezas se realiza, con el fin de pasar las validaciones que aplica Google Magenta cuando analiza la longitud mínima que debe tener las partituras y el número de notas mínimo que debe haber en cada una de ellas.

5. La información procesada y filtrada ya se puede utilizar

como entrada a los diferentes experimentos que se van a realizar, con el fin de generar música flamenca del estilo de la Soleá.

Se realizarán 2 tipos de experimentos diferentes. Los basados en el uso de Google Magenta y los basados en la creación de redes LSTM específicas para la generación de música flamenca del estilo de la Soleá:

Experimentos con Google Magenta:

Se partirá del dataset de partituras flamencas de Soleá en formato MIDI.

Los experimentos que utilicen el framework de Google Magenta no necesitarán realizar ningún procesamiento adicional de los datos de entrada como pudieran ser la división en secuencias o la codificación One-Hot (se realizarán internamente por Google Magenta).

El modelo "*Melody_rnn*" que proporciona Google Magenta será el que se utilice para estos experimentos:

- Se probarán las diferentes configuraciones proporcionadas por el modelo "*Melody_rnn*". En todos los casos, el número de capas, neuronas por cada capa y tamaño del batch serán las que incorpora cada configuración por defecto.

- Se realizará el entrenamiento de la red para las diferentes configuraciones.

- Se generarán varias partituras con la red entrenada para diferentes configuraciones y para diferentes valores de temperatura.

- Se analizarán los resultados obtenidos para cada configuración y para diferentes valores de temperatura, en base a los criterios de evaluación descritos en el apartado de evaluación de resultados.

Experimentos con redes LSTM específicas:

Al igual que los experimentos realizados con el framework de Google Magenta, se partirá del dataset de partituras flamencas de Soleá en formato MIDI; pero en este caso, la información de entrada deberá procesarse para transformarla en formato texto, ya que la red LSTM no es capaz de computar eficientemente valores en formato MIDI. El proceso de conversión se realizará mediante un proceso Python y utilizando las funcionalidades de la librería Music21.

A continuación, la información textual (nombre nota y duración) se procesará para construir 3 vectores:

1- Vector de nota+duración: Se incluirá en el mismo vector la nota y la duración concatenando ambos con un carácter especial (por ejemplo "|").

2- Vector de duración: Se incluirá la duración de cada nota en cada componente del vector:

3- Vector de compás: En base a la duración de cada nota, se desarrollará un proceso que informe para cada nota y duración, a que compás pertenece.

Los vectores se codificarán con One-Hot, para que la información sea procesable de forma más eficiente por el algoritmo.

La información de los vectores codificados se dividirá en secuencias de n notas consecutivas, que serán la entrada a la red. La nota n+1, que también se representará como 3 vectores, será la salida de la red.

Se probarán diferentes topologías (LSTM, LSTM bidireccionales, mecanismos de "*Atención*", etc.). Para cada una de ellas:

- Se realizará el entrenamiento de la red. Para cada una de las arquitecturas.

- Se generarán, para cada una de las arquitecturas, varias partituras con la red entrenada para diferentes valores de temperatura. El número de notas a generar en cada caso se indicará cuando se realice el experimento.

- Se analizarán los resultados obtenidos para cada una de las arquitecturas y para cada uno de los valores de temperatura utilizados, en base a los criterios de evaluación descritos en el apartado de evaluación de resultados.

Evaluación de Resultados

Las melodías generadas del estilo de la Soleá son escuchadas por 3 personas diferentes (evaluadores). Estas personas evalúan cuantas falsetas (frase melódica compuesta por varios compases) de las varias que se generan en cada partitura, suenan dentro del estilo de la Soleá.

Para considerar que una falseta suena al estilo de la Soleá se deben cumplir 3 condiciones:

1. Dada una falseta, se considera que ésta va a compás cuando, empezando la melodía en el compás 1, el cierre o conclusión se realiza en el compás 12 o en un múltiplo de 12. Cada partitura que se genera contiene un número variable de falsetas y sólo algunas de ellas siguen el compás propio de la Soleá.

2. Además de seguir el compás, una falseta tiene que seguir la tonalidad y armonía flamencas de la Soleá, sin disonancias.

3. Si durante la audición de una partitura, se detectara que la música entra en un bucle, los evaluadores considerarán que las falsetas (dentro del bucle) no cumplen los criterios y que no suenan dentro del estilo de la Soleá. Este "*embuclamiento*" es más probable que se produzca cuando el valor de la temperatura de generación es 0 o muy próximo a él.

En base a estos 3 criterios, se indica, de forma tabulada para cada uno de los experimentos, el número de falsetas que cada evaluador considera que suenan dentro del estilo flamenco de la Soleá.

IV. Contribución

Este experimento aporta varias novedades que se detallan seguidamente:

La principal novedad que tiene este experimento es que se va a utilizar información rítmica del compás y que se añadirá a la que se extrae de las partituras MIDI. Es decir, si ya tenemos las notas que integran las piezas musicales y la duración de cada una de esas notas (o silencios), añadiremos a esta información la correspondiente al compás.

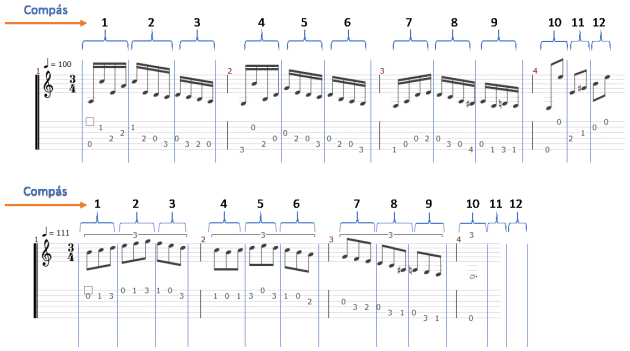

Figura 1. Formación del compás en la partitura. Fuente: Elaboración propia

Esta información rítmica se genera mediante un proceso informático que a cada nota y a su duración le asigna el compás que le corresponde. Al ser la Soleá un compás de 12 tiempos éste irá contabilizándose de 12 en 12 a lo largo de todas las partituras del dataset de entrada (cuando llega a 12 vuelve a empezar en 1).

Esta forma de asignar el compás sería válida para otros estilos musicales como el pop, rock, vals etc.

Construido el vector de compás, se integrará con la información de las notas y su duración.

Otra contribución consiste en incluir en el vector de notas, la concatenación de la nota y su duración. Además, mantendremos el vector de duración y por último añadiremos el vector calculado de compás:

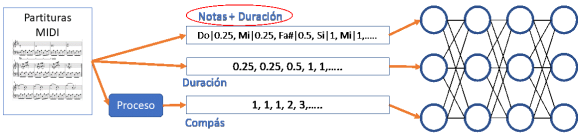

Figura 2. Modelo de 3 vectores de entrada utilizado en el experimento. Fuente: Elaboración propia

Por lo tanto, la entrada a la red constará de 3 vectores:

1. Vector de nota+duración: Se incluirá en el mismo vector la nota y la duración concatenando ambos con un carácter especial (por ejemplo "|").

2. Vector de duración: Se incluirá la duración de cada nota en cada componente del vector:

3. Vector de compás: En base a la duración de cada nota, se desarrollará un proceso que informe para cada nota y duración a que compás pertenece.

En principio puede parecer que la información de la duración no es necesaria puesto que esta se concatena dentro del vector de notas, sin embargo, esto no es así porque esta información resulta de utilidad para que la red aprenda el compás en base al vector de la duración. Si quitáramos este vector, el compás no podría aprenderse al estar la duración concatenada con la nota.

Por otra parte, al concatenar la nota y la duración dentro del mismo vector estamos haciendo dependientes el valor de la nota y su duración, lo cual permite que la red aprenda que ciertas notas o secuencias de notas solo se utilizan con determinadas duraciones. Es más, en teoría musical los valores de las notas siempre están íntimamente ligados a sus duraciones y no deben representarse de forma independiente valor y duración.

Otra contribución destacable es el uso de mecanismos de "*ajuste de compás y temperatura variable*" cuyo funcionamiento se divide en 2 partes que se describen a continuación:

Funcionalidad 1:

Una vez que la red se ha entrenado, la utilizamos para generar secuencias de notas partiendo de un pequeño conjunto inicial o "notas de cebado". A medida que la red neuronal va generando notas, estas vuelven a incluirse como entrada para seguir con la secuencia generativa. Esta retroalimentación se realiza de forma iterativa hasta que se genera el número de notas previamente establecido.

Puede ocurrir que, la información del compás que se genera no corresponda con la que debiera ser, lo que con toda certeza provoca que se rompa el ritmo y la melodía se vaya fuera de compas. En este caso, el mecanismo de "*ajuste de compás y temperatura variable*" actúa actualizando el valor del compás generado por la red por el que realmente le corresponde, para que la melodía siga manteniendo el compás que debe tener. Esto implica que este mecanismo debe tener control absoluto de las duraciones de las notas y silencios para saber en qué parte del compás se ubican las notas que se van generando por la red.

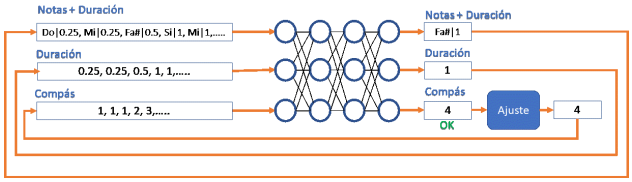

Figura 3. Mecanismo de ajuste de compás actuando. Fuente: Elaboración propia

Con este mecanismo se consigue que, aunque la red neuronal genere un valor de compás no válido, se cambie el valor por el correcto lo que fuerza a la red a seguir el compás propio de la soleá en todas las composiciones.

Funcionalidad 2:

La temperatura es un hiperparámetro de las LSTM (y de las redes neuronales en general) que se utiliza para controlar la variabilidad de las predicciones que se generan en la capa de salida de tipo "*Softmax*". Cuando la temperatura es 0, se toman directamente los valores de probabilidad normalizados de "*Softmax*" sin cambio alguno. A medida que el valor de la temperatura aumenta los valores más probables de "*Softmax*" se vuelve más pequeños y los más pequeños se hacen más grandes, lo que produce una distribución de probabilidad más suave sobre las clases, y hace que la red se "*excite*" más fácilmente con las muestras, lo que resulta en más diversidad y también en más errores.

Utilizar la temperatura en la composición musical hace que las notas que se generan vayan variando de acuerdo con el valor de temperatura que se haya empleado. Un valor bajo implica menor diversidad musical y un valor alto implica mayor aleatoriedad y variabilidad.

El problema radica en que el hiperparámetro de la temperatura suele ser un valor fijo que no cambia una vez que se empiezan a generar las secuencias de notas por parte de la red neuronal.

Con el fin de evitar esto, el mecanismo de "*ajuste de compás y temperatura variable*" tiene la funcionalidad añadida de que usa un valor de temperatura diferente en función del número de compás (temperatura alta para los compases iniciales y temperatura 0 para las últimas partes). Esto hace que las melodías generadas sean más variables o aleatorias al inicio del compás, pero que finalmente se reconduzcan a un cierre típico de la Soleá al disminuir drásticamente el valor de la temperatura. Con esto se consigue generar aleatoriedad musical en la composición en los

primeros tiempos, pero obligando a la melodía a ajustarse a algunos de los cierres de la Soleá característicos en los tiempos finales (10, 11, y 12). Este mecanismo tendrá como entrada una lista de compases para los que la temperatura se podrá a cero independientemente del valor de temperatura que se haya indicado. En este trabajo se han usado la lista de compases de [8,9,10,11,12] con el fin de que únicamente haya aleatoriedad en los 7 primeros compases. Una vez pasados estos la temperatura toma el valor 0 con el fin de que la melodía se reconduzca hasta llegar al cierre en los últimos compases.

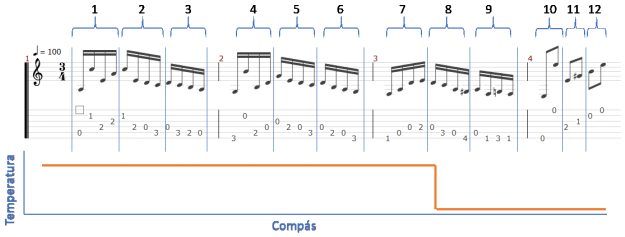

Figura 4. Evolución del valor de la temperatura en función del compás. Fuente: Elaboración propia

V. Resultados

Los experimentos realizados han consistido en la generación de música flamenca del estilo de la Soleá y utilizando diferentes arquitecturas.

En cada experimento se han generado 10 piezas musicales del estilo de la Soleá para 4 temperaturas diferentes (0, 0.5, 1 y 1.5). Las piezas generadas han sido evaluadas por una persona conocedora de este tipo de música. Se ha analizado cuantas falsetas (fragmentos melódicos) siguen el compás y la melodía propias de la Soleá para el conjunto de partituras generadas para cada temperatura.

Experimento 1 - Google Magenta. "Basic_rnn"

Los resultados obtenidos utilizando el framework de Google Magenta para el modelo "*Melody_rnn*" y la configuración "*Basic_rnn*" han sido:

Temperatura	Partituras de 500 Notas	Falsetas Generadas	Evaluador Profesional	Evaluador Avanzado	Evaluador Aficionado
0	10	100	22	24	36
0,5	10	94	14	20	30
1	10	96	14	22	28
1,5	10	102	16	20	24
TOTAL	**40**	**392**	**66**	**86**	**118**
Porcentajes por Evaluador			**16,83%**	**21,93%**	**30,10%**
Porcentaje Medio Final			**22,95%**		

Tabla 1. Resultado de la evaluación del experimento 1

Experimento 2 - Google Magenta. "Attention_rnn"

Los resultados obtenidos utilizando el framework de Google Magenta para el modelo "*Melody_rnn*" y la configuración "*Attention_rnn*" han sido:

Temperatura	Partituras de 500 Notas	Falsetas Generadas	Evaluador Profesional	Evaluador Avanzado	Evaluador Aficionado
0	10	104	12	14	16
0,5	10	102	14	18	18
1	10	92	10	14	15
1,5	10	100	10	12	15
TOTAL	**40**	**398**	**46**	**58**	**64**
Porcentajes por Evaluador			**11,55%**	**14,57%**	**16,08%**
Porcentaje Medio Final			**14,06%**		

Tabla 2. Resultado de la evaluación del experimento 2

Experimento 3 - LSTM

Los resultados obtenidos utilizando una arquitectura propia basada en LSTM y utilizando el nuevo vector de compás y el mecanismo de "*ajuste de compás y temperatura variable*" han sido:

Temperatura	Partituras de 500 notas	Falsetas generadas	Evaluador Profesional	Evaluador Avanzado	Evaluador Aficionado
0	10	140	30	32	30
0,5	10	142	130	134	128
1	10	135	102	105	112
1,5	10	147	38	42	46
TOTAL	**40**	**564**	**300**	**313**	**316**
Porcentajes por Evaluador			**53,19%**	**55,49%**	**56,02%**
Porcentaje Medio Final			**54,9%**		

Tabla 3. Resultado de la evaluación del experimento 3

Experimento 4 - LSTM con mecanismo de "Atención"

Los resultados obtenidos utilizando una arquitectura propia basada en LSTM, con mecanismo de "*Atención*" y utilizando el nuevo vector de compás y el mecanismo de "*ajuste de compás y temperatura variable*" han sido:

Temperatura	Partituras de 500 notas	Falsetas generadas	Evaluador Profesional	Evaluador Avanzado	Evaluador Aficionado
0	10	136	90	94	98
0,5	10	142	132	136	136
1	10	138	101	105	105
1,5	10	142	42	45	46
TOTAL	**40**	**558**	**365**	**380**	**385**
Porcentajes por Evaluador			**65,41%**	**68,10%**	**68,99%**
Porcentaje Medio Final			**67,5%**		

Tabla 4. Resultado de la evaluación del experimento 4

VI. Discusión

En base a los resultados obtenidos en los diferentes experimentos realizados anteriormente, podemos destacar que:

• El framework Google Magenta no es la utilidad que mejor se adaptan a la generación de música flamenca y en concreto a la composición de melodías del estilo de la Soleá. Los resultados de porcentaje medio final de 22,95% para la configuración "Basic_rnn" y 14,06% para "*Attention_rnn*", muestran que las partituras que se generan no van a compás. Es decir, las falsetas no terminan con un cierre en el compás 12, algo que es fundamental para conseguir que la melodía tenga la rítmica propia de la Soleá.

• Los experimentos realizados con la configuración "*Basic_rnn*" del modelo "*Melody_rnn*" acotan el rango de generación de las melodías a los valores MIDI [48, 84], lo que en términos de música flamenca implica, por una parte, reducir el rango tonal, y por otra, generar notas que no son propias del estilo flamenco de la Soleá.

• La configuración de "*Attention_rnn*" del modelo "*Melody_rnn*" de Google Magenta permite la generación de notas en todo el rango tonal lo que hace que las melodías creadas se asemejen más al estilo de la Soleá. No obstante, al igual que la configuración "*Basic_rnn*", las partituras generadas carecen de compás y no se ajustan a la línea melódica que debería tener una Soleá. Entendiendo por línea melódica que las notas estén dentro de los acordes propios de la cadencia de la Soleá (Mi, Fa, Sol, Lam) y que la suene coherente, sin disonancias excesivas.

• Los valores de porcentaje medio final de 14,06% para la configuración de "*Attention_rnn*" del modelo "*Melody_rnn*" resulta levemente inferior al valor de 22,95% que se obtiene para la configuración "*Basic_rnn*". La razón es que, aunque se producen menos disonancias (se mejora la melodía con el mecanismo de atención), se pierde el compás propio de la Soleá y las falsetas no terminan en el tiempo 12

• Con la utilización de los mecanismos de "*ajuste de compás y temperatura variable*" se consigue que, aunque la red neuronal genere un valor de compás no válido, se cambie el valor por el correcto, lo que fuerza a la red a seguir el compás propio de la soleá en todas las composiciones. Además, el uso de un valor de temperatura diferente en función del número de compás (temperatura alta para los compases iniciales y temperatura 0 para las últimas partes) hace que las melodías generadas sean más variables o aleatorias al inicio del compás, pero que finalmente se reconduzcan a un cierre típico de la Soleá al disminuir drásticamente el valor de la temperatura. Con esto se consigue generar aleatoriedad musical en la composición en los primeros tiempos, pero obligando a la melodía a ajustarse a algunos de los cierres de la Soleá característicos en los tiempos finales.

• La creación de 3 vectores diferentes con las secuencias de las notas que son introducidos a la entrada de la red neuronal mejora la generación de música flamenca ya que cada nota se genera de forma conjunta con la duración de ésta dentro del primer vector. Esto implica que el valor de la nota y su duración sean computados como una unidad por la red neuronal durante el entrenamiento y que la salida del valor de la nota sea acorde a su duración. Por otra parte, la utilización del vector de duración de forma independiente ayuda a que el vector de compás pueda entrenarse correctamente puesto que puede calcular cual es el siguiente compás en base a las duraciones anteriores que ha procesado.

• Los mecanismos de "*Atención*" utilizados en el último experimento permiten a la red fijarse en las partes más importantes del dataset de entrenamiento, lo que permite aprender las dependencias a largo plazo de forma más eficiente y generar piezas de Soleá más extensas.

• Los valores de porcentaje medio final de 54,9% y 67,5% obtenidos con los experimentos 3 y 4 indican que se obtiene mejores resultados con respecto a los experimentos 1 y 2 realizados con Google Magenta, cuyos valores son 22,95% y 14,06%. Esto demuestra que, los nuevos mecanismos utilizados (ajuste de compas, temperatura variable y el vector de compas), son efectivos a la hora de generar música flamenca ya que hacen que las piezas lleven la rítmica propia de este estilo y que las melodías suenen coherentes sin disonancias excesivas.

• Dependiendo de la temperatura utilizada se obtiene mayor o menor variabilidad. Cuando se utiliza temperatura 0 las piezas se vuelven repetitivas transcurridos unos compases. En el extremo opuesto, cuando se utiliza la temperatura 1,5, la música generada es demasiado aleatoria y se producen excesivas disonancias en todos los experimentos. Las temperaturas entre 0,5 y 1,0 son las que mejores resultados ofrecen, aportando suficiente aleatoriedad para una variabilidad musical, sin perder la línea melódica propia de la Soleá.

• La diferencia entre los valores de porcentaje medio final de los experimentos 3 y 4 son pequeñas (54,9% frente a 67,5%), no obstante, se obtiene mejores resultados en el experimento 4 (67,5%) porque las melodías generadas con temperatura 0 no resultan tan repetitivas como las generadas con el experimento 3, por el uso del mecanismo de "*Atención*".

• De los resultados obtenidos se deduce que la mejor arquitectura para la generación de música flamenca es la red LSTM con "*Atención*", mecanismo de "ajuste de compás y temperatura variable" y utilizando temperaturas entre 0,5 y 1.

VII. Conclusiones

En este trabajo se han realización un conjunto de experimentos con distintas arquitecturas de Deep Learning con el fin de explorar diferentes técnicas de composición de música flamenca para guitarra del estilo de la Soleá.

Se han revisado las arquitecturas más importantes que se han utilizado a lo largo de los años en la generación de contenidos musicales, desde los inicios en composición algorítmica, hasta las técnicas más actuales basadas en Deep Learning, poniendo especial énfasis en las redes LSTM, en los mecanismos de "*Atención*" y en las estrategias de "*Condicionamiento*".

Una vez estudiadas las diferentes arquitecturas y mecanismos, se han realizado diferentes experimentos con el fin de probar la eficacia de cada uno de ellos en el ámbito de la generación de música flamenca del estilo de la Soleá. En este sentido, se ha trabajado con 2 tipos de arquitecturas diferentes: Google Magenta (experimentos 1 y 2) y una arquitectura desarrollada específicamente para la composición de música flamenca basada en LSTM (experimentos 3 y 4).

En la primera de ellas (Google Magenta), los valores de porcentaje medio final obtenidos en los experimentos 1 y 2 son de 22,95% y 14,06%, respectivamente. Esto indica que las composiciones flamencas de Soleá generadas por este framework no siguen el compás propio de este estilo y tienen excesivas disonancias.

En la segunda arquitectura, utilizada en los experimentos 3 y 4, se han añadido las siguientes novedades:

• Vector de compás: Permite que la red aprenda el ritmo, su cadencia, los cierres que se utilizan y cuando utilizarlos, además de las notas y duraciones que pueden ir en cada compás.

• Concatenación del valor de la nota con su duración: Permite que la red aprenda de forma conjunta que nota está asociada a qué duraciones.

• Nuevo mecanismo de "ajuste de compás y temperatura

variable": Permite corregir el valor del compás cuando se genera una predicción errónea de este, lo que fuerza a la melodía a mantener la rítmica propia de la Soleá. Además, la idea de permitir la variabilidad de la melodía en los primeros compases y una vez pasados estos eliminarla (temperatura variable), hace que la melodía llegue al cierre de forma controlada, generando las secuencias típicas de cierre de la Soleá. Esto incrementa la aleatoriedad musical sin perder el control en las secuencias finales tan propias del flamenco.

Estas novedades hacen que la generación de música flamenca para el estilo de la Soleá mejore significativamente en términos de compás y línea melódica, con respecto a la música generada con las librerías de Google Magenta, consiguiéndose unos porcentajes medios finales de 54,9% (experimento 3) y 67,5% (experimento 4).

Además de las novedades anteriores, en el experimento 4 se ha incluido un mecanismo de "Atención", que permite al modelo acceder más fácilmente a la información pasada sin tener que almacenar esta información en la red neuronal. Esto permite que el modelo aprenda más fácilmente las dependencias a largo plazo pudiendo generar melodías más largas. El porcentaje medio final que resulta de incluir el mecanismo de atención en el experimento 4 es de 67,5%, frente al 54,9% del experimento 3 (que no incluye este mecanismo). Es decir, existe una mejora significativa gracias al uso de este mecanismo de "Atención".

No obstante, podemos destacar como problema, el derivado del uso de determinados formatos utilizados para la representación de los datos de entrada a la red neuronal. Concretamente, el formato utilizado para representar el primer vector, donde se unen el valor de la nota más el valor de su duración. Esto hace que el tamaño del vocabulario sea más extenso (notas diferentes x duraciones diferentes) de lo habitual, lo que supone un incremento de los recursos de computación necesarios para el entrenamiento de la red neuronal.

Finalmente, los experimentos realizados y los resultados obtenidos nos permiten concluir que se ha alcanzado el objetivo general inicial de estudiar distintas arquitecturas y técnicas de Inteligencia Artificial aplicadas a la composición de música flamenca para guitarra del estilo de la Soleá.

REFERENCIAS

[1] Hadjeres, G., & Nielsen, F. (2017). I*nteractive Music Generation with Positional Constraints using Anticipation-RNNs*. 1–9. http://arxiv.org/abs/1709.06404.

[2] Lecun, Y., Bottou, L., Bengio, Y., & Ha, P. (1998). *GradientBased Learning Applied to Document Recognition. Proceedings of the IEEE*, November, 1–46.

[3] Oord, A. van den, Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., & Kavukcuoglu, K. (2016). *WaveNet: A Generative Model for Raw Audio*. 1–15. http://arxiv.org/abs/1609.03499.

[4] Mozer, M. (1994). *Neural Network Music Composition by Prediction: Exploring the Benefits of Psychoacoustic Constraints and Multi-scale Processing*.

[5] Hochreiter, S., & Schmidhuber, J. (1997). *Long Short-Term Memory*. 1780, 1735–1780.

[6] Eck, D., & Schmidhuber, J. (2002). *A First Look at Music Composition using LSTM Recurrent Neural Networks*. Istituto Dalle Molle Di Studi Sull Intelligenza Artificial, 103, 48.

[7] Sturm, B. L., Santos, J. F., Ben-Tal, O., & Korshunova, I. (2016). *Music transcription modelling and composition using deep learning*. http://arxiv.org/abs/1604.08723.

[8] Nayebi, A., & Vitelli, M. (2015). GRUV: *Algorithmic Music Generation using Recurrent Neural Networks. Deep Learning for Natural Language Processing*, 1–6. https://web.stanford.edu/~anayebi/projects/CS_224D_Final_Project_Writeup.pdf.

[9] Hyungui LIm, Seungyeon Rhyu, K. L. (2017). *Chord generation from symbolic melody using blstm networks Music and Audio* Research Group, Graduate School of Convergence Science and Technology 4 Center for Super Intelligence Seoul National University, Korea. CoRR, abs/1712.0, 621–627. http://arxiv.org/abs/1712.01011.

[10] Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. 1–9. http://arxiv.org/abs/1412.3555.

[11] Yang, L. C., Chou, S. Y., & Yang, Y. H. (2017). *Midinet: A convolutional generative adversarial network for symbolic-domain music generation. Proceedings of the 18th International Society for Music Information Retrieval Conference*, ISMIR 2017, 324–331.

[12] Dong, H. W., Hsiao, W. Y., Yang, L. C., & Yang, Y. H. (2018). *Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment*. 32nd AAAI Conference on Artificial Intelligence, AAAI 2018, 34–41.

[13] Kotecha, N. (2018). *Bach2Bach: Generating Music Using A Deep Reinforcement Learning Approach*. http://arxiv.org/abs/1812.01060.

[14] Roberts, A., Engel, J., Raffel, C., Hawthorne, C., & Eck, D. (2018). *A hierarchical latent vector model for learning long-term structure in music*. 35th International Conference on Machine Learning, ICML 2018, 10, 6939–6954

[15] Huang, C. Z. A., Vaswani, A., Uszkoreit, J., Shazeer, N., Simon, I., Hawthorne, C., Dai, A. M., Hoffman, M. D., Dinculescu, M., & Eck, D. (2019). *Music transformer: Generating music with long-term structure*. 7th International Conference on Learning Representations, ICLR 2019, 1–15.