Universidad Politécnica
de Madrid

Escuela Técnica Superior de
Ingenieros Informáticos

Grado en Ciencia de Datos e Inteligencia Artificial

Trabajo Fin de Grado

Modelos Neuronales para la Creación de Piezas Musicales en Piano

Autor: Rubén Sánchez Fernández

Tutor(a): Daniel Manrique Gamo

Madrid, junio de 2025

Este Trabajo Fin de Grado se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid para su defensa.

Trabajo Fin de Grado
Grado en Ciencia de Datos e Inteligencia Artificial
Título: Modelos Neuronales para la Creación de Piezas Musicales en Piano
Junio 2025

Autor: Rubén Sánchez Fernández

Tutor:
Daniel Manrique Gamo
Departamento de Inteligencia Artificial
ETSI Informáticos
Universidad Politécnica de Madrid

Agradecimientos

A mi madre, por siempre haber estado a mi lado y darme todo el cariño para poder seguir adelante. A mi padre, por ayudarme y haber sido un gran apoyo para mí. A mis abuelos, que seguro se sienten orgullosos de todo lo que he logrado hasta el momento. A mi abuela, quien siempre me ha guiado con sus consejos llenos de sabiduría

A todas aquellas amistades que he podido disfrutar grandes momentos con todas ellas. En especial, a esos amigos más cercanos, con los cuales he vivido experiencias que se quedarán conmigo para siempre y han conseguido que llegue a disfrutar más de cada instante de la vida.

Gracias a todos por hacer que los días malos se vuelvan buenos, y los buenos sean aún mejores.

"Alegre es la vida. Y corta,
pasajera.
Y es absurdo,
y es antipático y zurdo
complicarla
con un ansia de verdad
duradera
y expectante.
¿Luego?... ¡Ya!
La verdad será cualquiera.
Lo precioso es el instante
que se va."

La canción del presente, Manuel Machado (1920)

Resumen

Este Trabajo de Fin de Grado explora el reto de generar sintéticamente piezas a piano mediante Inteligencia Artificial, enfocándose en preservar la armonía y fluidez melódica. La premisa central plantea que, mediante un procesamiento eficaz de la información musical y la implementación de modelos neuronales especializados, es posible generar secuencias de notas que integren complejidad rítmica con una experiencia auditiva satisfactoria.

Se han implementado diversas arquitecturas de redes neuronales, entre las cuales se incluyen las redes neuronales recurrentes (*RNN*), con especial atención a las unidades de memoria a corto-largo plazo (*LSTM*), las unidades recurrentes cerradas (*GRU*) y modelos híbridos de los anteriores. Así mismo, se incorpora el modelo autorregresivo *WaveNet*, lo que permite una exploración más amplia de técnicas avanzadas en el modelado de secuencias temporales musicales. Cada modelo neuronal se alimenta de un conjunto de datos musicales que ha sido necesario preprocesar de forma específica, adaptando las características sonoras a los requisitos técnicos de generación melódica. El proceso de implementación común se sustenta en tres pilares: optimización de hiperparámetros, técnicas de control de sobre-entrenamiento e implementación de técnicas para estimular la variación creativa en las producciones musicales.

Este trabajo presenta asimismo una evaluación comparativa de modelos neuronales, revelando su capacidad para predecir elementos como altura tonal, subsecuencias únicas y escalas melódicas propias. El estudio se complementa con valoraciones perceptuales auditivas para medir con mayor precisión la eficacia de distintos métodos computacionales en la creación musical.

Los resultados muestran que, si bien las arquitecturas implementadas aún no alcanzan la complejidad compositiva del ser humano, son capaces de generar piezas para piano sorprendentemente expresivas, que logran un equilibrio atractivo entre innovación y coherencia melódica. En particular, las implementaciones de modelos como *WaveNet*, *GRU* y el enfoque híbrido *LSTM-GRU* han demostrado ser especialmente adecuadas para la creación de melodías sintéticas a piano, gracias a su capacidad para producir secuencias musicalmente ricas, dinámicas y estructuralmente coherentes. Los modelos neuronales desarrollados tienen aplicación en el desarrollo de herramientas pedagógicas y de asistencia creativa, sentando un punto de partida para futuras aplicaciones artísticas en el campo de la generación musical automatizada.

Abstract

This Final Degree Project explores the challenge of generating piano pieces using Artificial Intelligence , focusing on preserving harmony and melodic fluency. The central assumption is that, through efficient processing of musical information and the implementation of specialised neural models, it is possible to generate sequences of notes that integrate rhythmic complexity with a satisfactory listening experience.

Several neural network architectures have been implemented, including recurrent neural networks (*RNN*), with a focus on long short-term memory units (*LSTM*), gated recurrent units (*GRU*) and hybrid models of the latter. In addition, the *WaveNet* autoregressive model is incorporated, allowing a wider exploration of advanced techniques in the modelling of temporal music sequences. Each neural model is supplied with specific musical data processing, adapting the audio characteristics to the technical requirements of melodic generation. The common implementation process is based on three pillars: hyperparameter optimisation, overfitting control techniques and implementation of resources that stimulate creative variation in music generations.

This project also presents a comparative evaluation of neural models, revealing their ability to predict elements such as pitch, unique sub-sequences and melodic scales of their own. The study is complemented by auditory perceptual assessments to measure more accurately the effectiveness of different computational methods in music creation.

The results show that, while the implemented architectures do not yet reach human compositional complexity, they are capable of generating surprisingly expressive piano pieces that achieve an appealing balance between innovation and melodic coherence. In particular, implementations of models such as *WaveNet*, *GRU* and the hybrid *LSTM-GRU* approach have demonstrated to be particularly well suited for the creation of synthetic piano melodies, due to their ability to produce musically rich, dynamic and structurally coherent sequences. The neural models developed have application in the development of pedagogical tools and creative assistance, establishing a solid basis for future artistic applications in the field of automated music generation.

Tabla de contenidos

1 Introducción

La Inteligencia Artificial (IA) se ha integrado con fuerza en distintos sectores de la sociedad actual, transformando la metodología empleada en diversos procesos gracias a enfoques más eficientes y novedosos. En el terreno creativo, su impacto es cada vez más notorio, como ocurre en el caso de la generación de música automatizada. En este caso, es necesario aunar los algoritmos de aprendizaje automático con conocimientos musicales para tratar de logar composiciones originales que imiten géneros, estructuras armónicas y patrones rítmicos a partir del análisis de obras ya existentes [1]. Esta situación plantea nuevas preguntas en el ámbito de la ética sobre qué significa ser músico o los derechos de autor, al mismo tiempo que abre la puerta a que más personas puedan crear música sin necesidad de una formación teórica profunda.

El proceso técnico detrás de esta innovación enfrenta obstáculos particulares. A diferencia de la pintura o la literatura, la música depende de una secuencia temporal exacta donde cada nota, silencio y acorde debe encajar en fracciones de segundo precisas. Además, requiere una estructura global que mantenga coherencia emocional y lógica interna, algo especialmente complejo de replicar artificialmente. Para lograrlo, se implementan en este trabajo modelos de redes neuronales recurrentes (*RNN*) como las celdas de memoria a corto-largo plazo (*LSTM*) y unidades recurrentes cerradas (*GRU*), diseñadas para procesar secuencias temporales extensas, como las que conforman una melodía, reteniendo información contextual a lo largo del tiempo [2]. Así mismo, arquitecturas como *WaveNet* emplean capas de convolución dilatada, una técnica que permite analizar relaciones sonoras a distintas escalas temporales simultáneamente [3].

El objetivo principal de este trabajo es implementar y comparar el rendimiento de estas arquitecturas neuronales en la generación de melodías de piano sintéticas, así como se evaluar su capacidad para producir secuencias de 200 notas musicales coherentes, equilibrando complejidad armónica y fluidez melódica. Para ello, se establecen los siguientes subobjetivos:

1. Entrenamiento, configuración y adaptación de los modelos *LSTM*, *GRU*, combinaciones híbridas y *WaveNet* al conjunto de datos *MAESTRO*, compuesto por grabaciones clásicas en formato MIDI.

2. Evaluación objetiva y subjetiva, utilizando métricas como polifonía, consistencia escalar y rango tonal para cuantificar la calidad técnica de las composiciones generadas. Así mismo, acompañarlo de una escucha activa que detalle la percepción auditiva en el oyente.

3. Análisis comparativo de los resultados obtenidos con modelos actuales como *MuseNet*, *C-RNN-GAN y GANSynth* en términos de estructura musical y aceptación auditiva.

4. Explorar las implicaciones socioculturales de esta tecnología en la creación artística, considerando su potencial para reducir barreras técnicas y sus desafíos éticos en torno a la autoría.

La estructura de este Trabajo de Fin de Grado se organiza en ocho capítulos. Tras esta introducción, el Capítulo 2 profundiza en los fundamentos teóricos de las redes neuronales. El Capítulo 3 aborda conceptos musicales esenciales y analiza los modelos neuronales relacionados con la generación musical

automatizada. En el Capítulo 4 se describe la metodología empleada junto con los objetivos a alcanzar. El Capítulo 5 detalla la implementación realizada de los modelos neuronales y presenta los resultados, comparando el rendimiento de cada modelo en términos técnicos y estéticos. En el Capítulo 6 se presentan las conclusiones, destacando el modelo neuronal que obtiene los mejores resultados en la generación de melodías de piano sintéticas. El Capítulo 7 propone líneas futuras de investigación, mientras que el Capítulo 8 explora las implicaciones sociales y artísticas de esta tecnología en el ámbito musical.

2 Redes de Neuronas Artificiales

En las últimas décadas, la evolución que ha experimentado el computador ha sido imparable. Lo que comenzó como una herramienta que servía para proporcionar ayuda en problemas de cálculo destinado a un público muy limitado, se convirtió en un objeto ampliamente democratizado que ha conseguido automatizar prácticamente cualquier proceso tedioso a realizar [4].

Gracias a ello, se ha revivido el sueño de lograr que las computadoras aprendan y emulen el comportamiento humano a través del aprendizaje automático. Un sueño que tuvo uno de sus inicios en 1943 de la mano de Warren S. McCulloch y Walter Pitts proponiendo el primer modelo neuronal que consistía en recrear la actividad cerebral mediante un elemento llamado célula o neurona artificial [5]. La neurona artificial es concebida como un dispositivo binario; es decir, con dos posibles estados. Puede recibir múltiples entradas y devuelve una única salida. La neurona solo permanece activa si la entrada total ponderada o neta rebasaba un cierto umbral, e inactiva en caso contrario. Además, aunque el modelo no sea un reflejo fiel en términos biológicos, sí que consiguió asentar las bases para proseguir esta línea de investigación en años posteriores [6].

La investigación en este campo alcanza un hito en 1958, cuando el psicólogo Frank Rosenblatt planteó un modelo neuronal llamado *Perceptrón*, el cual consiste en una única capa de neuronas de entrada y una de salida [7]. Su funcionamiento para una neurona de salida i se representa en la Figura 1: los valores de entrada x_i son multiplicados por los pesos w_{ij} de las conexiones asociadas a cada una de las neuronas en la capa de entrada. Después, se calcula el sumatorio y se aplica una función de activación umbral, que devuelve un valor binario.

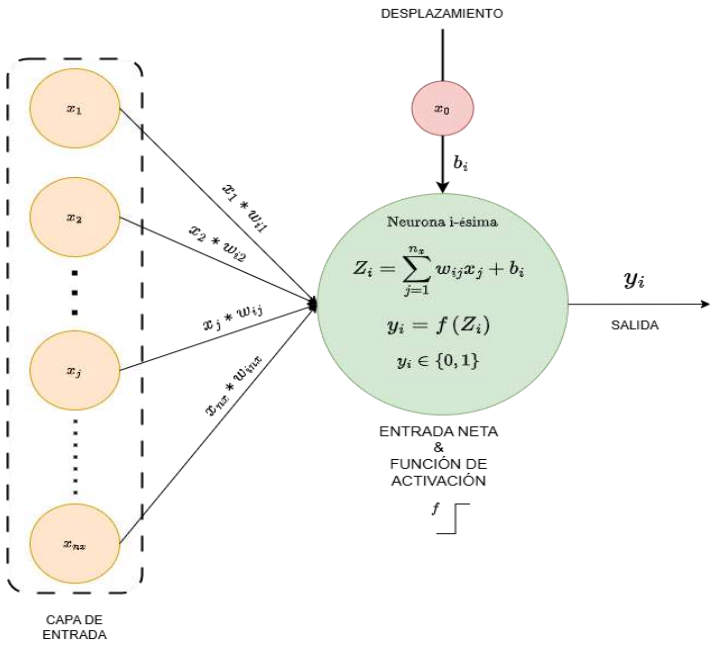


Figura 1. Diagrama del funcionamiento del Perceptrón.

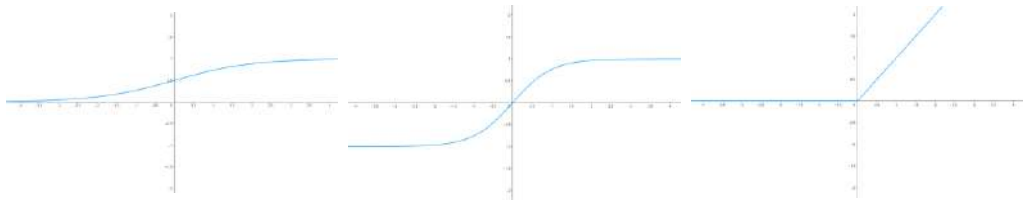
La forma en la que el Perceptrón consigue aprender es mediante la modificación de los pesos de las conexiones. De esta forma, se consigue obtener los valores de salida deseados en función de la entrada. Gráficamente, el *Perceptrón* se puede interpretar como un hiperplano que realiza una clasificación binaria

sobre regiones que son linealmente separables, como es el caso de las puertas lógicas AND (conjunción), OR (disyunción) o NOT (negación). No obstante, la puerta lógica XOR (disyunción exclusiva) no es linealmente separable, por lo que el *Perceptrón* no es capaz de realizar correctamente la clasificación binaria correspondiente a las dos posibles salidas [8].

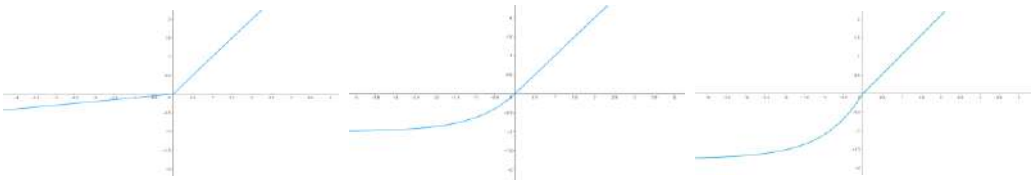
Se pueden considerar dos elementos principales para llevar a cabo el proceso de aprendizaje de un modelo neuronal:

- Los pesos de las conexiones w_{ij} representan la "fuerza" de la conexión entre dos neuronas j e i, lo cual determina su relevancia dentro de la red neuronal para obtener el valor deseado de la salida. El valor w_{i0} se corresponde con el desplazamiento b_i (véase Fig. 1), el cual lleva asociado el valor de entrada $x_0 = 1$. Representa el término independiente de la ecuación del hiperplano separador de las clases, permitiendo que no necesariamente tenga que pasar por el origen de coordenadas.
- La función de activación devuelve la salida de la red una vez computado la entrada neta. En el caso del *Perceptrón*, la función de activación es la umbral, la cual devuelve el valor 1 al superarlo y 0 en caso contrario [9].

Las funciones de activación son esenciales para que la red neuronal pueda llegar a aprender relaciones más complejas. Además, deben tener una derivada que sea fácil de calcular para que, de esta manera, el entrenamiento de la red sea más eficiente. Algunas de las funciones más utilizadas se detallan en la Figura 2.



a. Función sigmoide b. Función tangente hiperbólica c. Función *ReLU*



d. Función *Leaky ReLU* e. Función *ELU* f. Función *SELU*

Figura 2. Funciones de activación.

Como se puede apreciar, la función sigmoide (Figura 2a) es continua en todo su dominio y devuelve valores dentro del intervalo [0, 1]. Por tanto, resulta especialmente interesante en problemas de clasificación binarios. Asimismo, su derivada también es continua y fácil de calcular.

$$f(x) = \frac{1}{1+e^{-x}} \tag{2.1}$$

No obstante, como su derivada tiene forma de campana de Gauss aplanada, el aprendizaje de la red con valores de entrada neta alejados del 0 es prácticamente

nulo, lo cual supone una gran dificultad a la hora de entrenar redes de neuronas con múltiples capas ocultas (redes profundas o *deep*) y gran número de neuronas en cada capa.

La función de activación tangente hiperbólica Figura 2a es también continua en todo su dominio y devuelve valores dentro del intervalo [-1, 1]. Presenta el mismo problema que la función sigmoide:

$$f(x) = tanh(x). \tag{2.2}$$

El anterior problema planteado para las funciones de activación (2.1) y (2.2) tiene fácil solución mediante la función *ReLU* (Figura 2c), la cual devuelve el mismo valor si la entrada es mayor que 0, y 0 en caso contrario. Su derivada es bastante simple favoreciendo un aprendizaje rápido y constante. Su principal desventaja es que, al tener como salida 0 para valores negativos, favorece la aparición de "neuronas muertas"; es decir, neuronas que no participan en el proceso de aprendizaje de la red al volverse inservibles porque siempre devuelven el mismo valor nulo:

$$f(x) = \max(0, x). \tag{2.3}$$

Sin embargo, esta desventaja puede quedar solventada estableciendo una ligera pendiente para los valores negativos, lo que se denomina función *Leaky ReLU* (Figura 2d):

$$\mathrm{f}(x) = \max(\alpha * x, x), \tag{2.4}$$

donde α debe ser un número pequeño, típicamente 0.01. Existen más variantes de la función *ReLU* que consiguen evitar estos problemas, tales como la función ELU (Figura 2e):

$$\mathrm{f}(x) = \max(\alpha(e^x - 1), x), \tag{2.5}$$

que es exponencial para valores negativos, para que decrezcan de manera más suave. Gracias a esta modificación, se suele reducir el tiempo de entrenamiento de la red. Por último, la función SELU, mostrada en la Figura 2f:

$$\mathrm{f}(x) = \max\big(\lambda x, \lambda\alpha(e^x - 1)\big), \tag{2.6}$$

donde $\alpha \approx 1.6733$ y $\lambda \approx 1.0507$. Posee las propiedades de una distribución normal estándar, por lo que si se aplica a una capa, las salidas se obtienen normalizadas dentro de un cierto rango sin necesidad de aplicar técnicas de regularización [10].

2.1 Redes neuronales multicapa

Las limitaciones que presenta el perceptrón imposibilitan resolver problemas más complejos, problemas no lineales.

En la década de los 70, el matemático finlandés Seppo Linnainmaa desarrolló un método para computar el gradiente de forma automática y eficiente, actualmente conocido como retropropagación (*backpropagation*) [11]. Y en 1986, se implementó computacionalmente por los investigadores David Rumelhart, Geoffrey Hinton y Ronald Williams en su artículo *Learning representations by back-propagating errors*. En este artículo, se entrenan redes de neuronas con varias capas utilizando funciones de activación no lineales, lo que se conoce como Perceptrón multicapa (MLP, *Multilayer Perceptron*) [12].

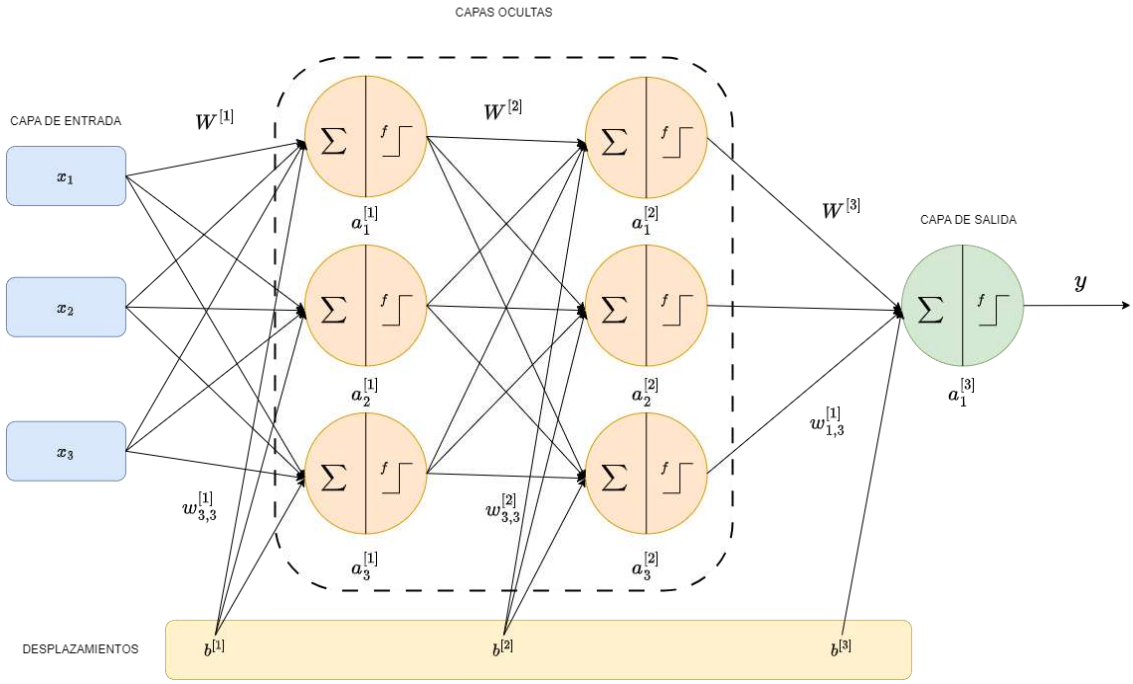


Figura 3. Diagrama del funcionamiento del Perceptrón multicapa.

La Figura 3 muestra la estructura de un perceptrón multicapa [13]. Como se puede observar, comparte con el anterior modelo (Fig. 1) la capa de entrada y la de salida, la cual devuelve el resultado final. Esta red recibe el nombre de red neuronal multicapa porque incorpora capas intermedias (capas ocultas) formadas por varias neuronas. Además, se trata de una red neuronal densa (*Dense Neural Network*) y alimentada hacia adelante (*Feedforward Neural Network*) porque cada una de las neuronas está conectada con todas las neuronas de la capa siguiente y los datos fluyen siempre de la entrada a la salida sin ciclos, respectivamente [11, 12].

Cada una de las neuronas calcula el sumatorio de sus entradas multiplicadas por los pesos de las conexiones (incluido el desplazamiento) y aplica una función de activación que es común para todas las neuronas de una misma capa. No obstante, la función de activación de la capa de salida suele variar dependiendo del problema. Para los problemas de clasificación que requieren más de dos clases, la función *Softmax* [16], mientras que para clasificación binaria, se usa la función sigmoide. En los problemas de regresión, la función de activación lineal es la más idónea [17].

Por último, notar que si en una red neuronal con capas ocultas, todas las neuronas utilizan la función de activación lineal, se puede demostrar que esta red es equivalente a otra con una única capa de entrada y otra de salida, sin capas ocultas [18].

2.2 Algoritmo de retropropagación del gradiente

El principal objetivo de las redes neuronales es aprender a realizar una tarea a partir de un conjunto de datos de entrenamiento. En el caso del paradigma de aprendizaje supervisado, se pretende que el valor de salida de la red, dado un vector de entrada, se asemeje lo más posible al valor deseado a obtener; es decir,

se busca reducir el error de predicción, definido mediante una función de pérdida. Para ello, se deben ajustar los pesos de las conexiones junto a los desplazamientos (parámetros de la red). Este es un proceso iterativo que se lleva a cabo mediante el algoritmo de retropropagación del gradiente del error. Este algoritmo permite optimizar la función de pérdida modificando los parámetros de la red [19]. Es común definir otra función de error global que permita conocer el error medio para el conjunto de datos de entrenamiento, conocido como función de coste.

Por ejemplo, en problemas de regresión, es común utilizar la suma de los errores de las neuronas de salida al cuadrado (*SSE*, *Sum of Squares Error*) como función de pérdida [20]:

$$\mathcal{L}(W) = \frac{1}{2}\left(Y - \hat{Y}\right)^2 ,$$

donde Y es el vector de salida deseada e $\hat{Y}$ el vector de salida calculada por la red para un vector de entrada.

La función de coste viene determinada por el error cuadrático medio (MSE, Mean Square Error):

$$\mathrm{MSE(W)} = \frac{1}{\mathrm{n}}\sum_{\mathrm{i=1}}^{\mathrm{n}}\left(Y - \hat{Y}\right)^2 ,$$

donde n es el cardinal del conjunto de datos de entrenamiento. Teniendo en cuenta que se busca minimizar la función de pérdida a través del descenso del gradiente y ver su variación con respecto a los pesos:

$$\Delta \mathrm{w_{ij}} = -\alpha \frac{\partial \mathcal{L}(\mathrm{W})}{\partial \mathrm{w_{ij}}} , \tag{2.7}$$

donde α es la tasa de aprendizaje (*learning rate*). Esta tasa indica el tamaño del incremento obtenido por el descenso gradiente en la dirección del mínimo [21].

A través de la regla de la cadena, se calcula la derivada de la función de pérdida:

$$\frac{\partial \mathcal{L}(W)}{\partial w_{ij}} = -\frac{\partial \mathcal{L}(W)}{\partial net_i} \cdot \frac{\partial net_i(W)}{\partial w_{ij}} = -\frac{\partial \mathcal{L}(W)}{\partial net_i} \cdot x_j , \tag{2.8}$$

donde x_j representa la entrada proveniente de la neurona j.

Se define δ_i como la derivada de la función de pérdida con respecto a la entrada neta sobre la neurona i:

$$\delta_i = -\frac{\partial \mathcal{L}(W)}{\partial net_i} . \tag{2.9}$$

Se calcula δ_i volviendo a aplicar la regla de la cadena:

$$\delta_i = -\frac{\partial \mathcal{L}(W)}{\partial y_i} \cdot \frac{dy_i(W)}{d\, net_i} = \left(Y_i - \widehat{Y_i}\right) \cdot f'(net_i) , \tag{2.10}$$

donde $f'(net_i)$ es la derivada de la función de activación sobre la entrada neta recibida.

Por tanto, partiendo de (2.7) y considerando (2.8), (2.9) y (2.10); se calculan los ajustes de los nuevos pesos, siendo:

$$\Delta w_{ij} = \alpha \cdot \delta_i x_j .$$

Los valores de δ_i para las capas ocultas se obtienen mediante la siguiente fórmula:

$$\delta_i = f'(net_i) \sum_k \delta_k W_{ki} ,$$

donde el subíndice k recorre las neuronas de la capa siguiente. Este proceso se ejecuta iterativamente actualizando los parámetros de la red neuronal hasta alcanzar una condición de parada.

Entre las posibles condiciones de parada se encuentran varias opciones importantes. Una de ellas es establecer un número máximo de iteraciones. Al alcanzarse este límite predefinido, el entrenamiento se detiene, incluso si el error no ha llegado al mínimo. Esto asegura que el proceso no se extienda indefinidamente, evitando así un uso excesivo de recursos [22].

Otra condición común es la convergencia del algoritmo. Se considera que el algoritmo de entrenamiento converge cuando la disminución del error entre iteraciones consecutivas es menor que un umbral predefinido, lo que sugiere que la red ha alcanzado un estado en el que las mejoras son marginales y, por tanto, se encuentra próxima a un error mínimo. En este contexto, si las métricas de evaluación, como la función de pérdida o la precisión, dejan de mostrar mejoras durante un número consecutivo de iteraciones el proceso de entrenamiento puede detenerse anticipadamente [23].

Adicionalmente, se puede detener el entrenamiento para evitar el sobreajuste. El sobreajuste ocurre cuando un modelo se adapta en exceso a los datos de entrenamiento, lo que reduce su capacidad de generalización a nuevos datos [24]. En redes profundas, el entrenamiento se prolonga hasta que el error en el conjunto de entrenamiento se reduzca lo máximo posible. Dado que estas redes suelen ser considerablemente más complejas en relación con el tamaño del conjunto de datos, es altamente probable que ocurra un sobreajuste. Para mitigar este problema, se emplean técnicas de regularización [24], las cuales ayudan a reducir el sobreentrenamiento de la red. Por el contrario, en redes poco profundas se calcula el rendimiento del modelo en el conjunto de validación. Si el error en este conjunto comienza a aumentar mientras el error en el conjunto de entrenamiento sigue disminuyendo, es una señal de sobreajuste. En este caso, se puede aplicar la técnica de detención temprana (*early stopping*), que implica detener el entrenamiento antes de que el modelo se sobreajuste, evaluando métricas como la función de pérdida o la precisión al final de cada iteración. Esto permite que el modelo conserve el estado que mejor equilibra su rendimiento entre el conjunto de entrenamiento y el de validación, evitando así un sobreajuste que podría afectar su capacidad de generalización [25].

Cada vez que se presenta a la red todo el conjunto de entrenamiento durante el proceso de aprendizaje, se dice que se ha completado una época (*epoch*). Durante una época, los parámetros de la red neuronal pueden haberse actualizado desde una sola vez, lo que se conoce como actualización en lote (*batch)*, o haberse actualizado cada vez que se presenta un ejemplo de entrenamiento, lo que se denomina actualización online o estocástica. Un término medio, usado frecuentemente, es la actualización en subconjuntos (*mini-batch)*, donde se acumula la actualización de pesos para cada subconjunto creado del total de los datos de entrenamiento [26].

Por último, cabe mencionar que actualmente existen una gran variedad de optimizadores (*Momentum*, *RMSProp* o *Adam*) que aceleran el proceso de aprendizaje [27].

2.3 Optimizador RMSProp

El optimizador *RMSProp* (*Root Mean Square Propagation*) fue introducido por Geoffrey Hinton durante un curso sobre redes neuronales, como parte de sus recomendaciones prácticas para el entrenamiento de redes profundas. *RMSProp* se diseñó para abordar algunas limitaciones de métodos como el Descenso de Gradiente (*SGD*) y su variante *Momentum*. El optimizador *Momentum* acelera la convergencia acumulando gradientes pasados para generar un "impulso" hacia el mínimo, manteniendo una tasa de aprendizaje constante. En cambio, *RMSProp* utiliza un enfoque diferente: adapta dinámicamente la tasa de aprendizaje para cada parámetro, en función de los gradientes recientes. Este ajuste permite un mejor control en escenarios donde las magnitudes de los gradientes varían significativamente [28].

En términos matemáticos, el procedimiento de *RMSProp* en cada iteración k se describe como:

$$s_k \leftarrow \beta s_k + (1-\beta)(\partial\theta_k)^2 \quad \forall s_k, \tag{2.11}$$

$$\theta_k \leftarrow \theta_k - \frac{\alpha}{\sqrt{s_k + \varepsilon}}\partial\theta_k \quad \forall \theta_k, \tag{2.12}$$

donde s_k es el valor que acumula el cuadrado de los gradientes a través de un promediado ponderado exponencial. El valor θ_k indica el gradiente en esa misma iteración k. El hiperparámetro Beta se utiliza para la reducción exponencial y es ajustable, aunque por defecto se le asocia 0,9. El hiperparámetro ε se utiliza para evitar la división por 0, asociándole un valor muy pequeño en torno a 10^{-7}. Se puede observar que s_k es directamente proporcional a la pendiente que haya en esa iteración k, es decir la derivada del gradiente en esa iteración $\partial\theta_k$. Además, el gradiente en esa iteración se actualiza inversamente proporcional al valor de s_k. De esta manera, cuando la pendiente en ese instante sea más pronunciada, el valor del gradiente es menor y viceversa, adaptando el tamaño del paso en cada iteración k [29].

El éxito del optimizador *RMSProp* radica en su capacidad para facilitar y agilizar el entrenamiento de redes neuronales, especialmente en escenarios donde hay dificultades para alcanzar la convergencia al mínimo. Estas dificultades suelen estar relacionadas con tasas de aprendizaje inadecuadas o con la complejidad inherente de funciones de pérdida que presentan superficies altamente irregulares o con oscilaciones significativas [30]. La Figura 4 ilustra una comparación entre *SGD* y *RMSProp*, donde se observa que éste llega a encontrar una trayectoria más directa hacia el mínimo que aquél.

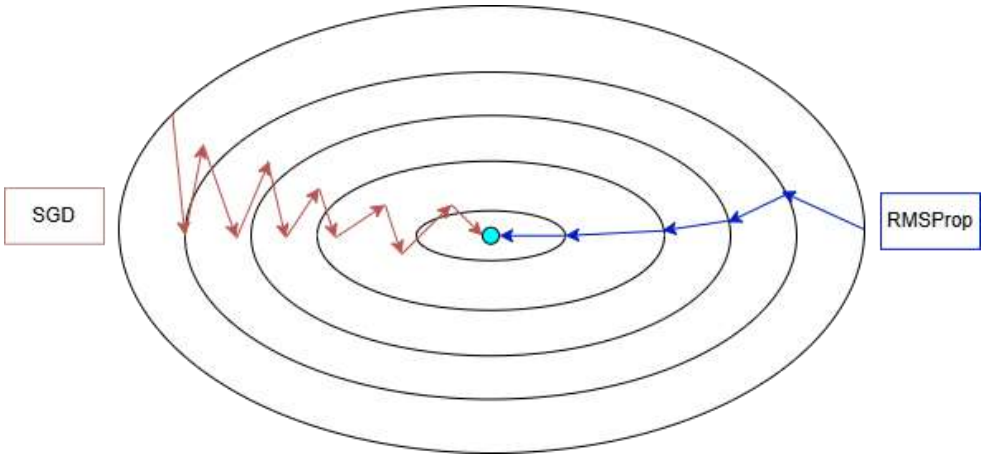


Figura 4. Comparación de las trayectorias entre SGD y RMSProp.

2.4 Optimizador Adam

El optimizador *Adam* fue desarrollado por Diederik Kingma y Jimmy Ba, quienes lo presentaron en su artículo de 2014 titulado *Adam: A Method for Stochastic Optimization*. Este método se diseñó con el propósito de combinar lo mejor de dos enfoques ya consolidados: *RMSProp* y *Momentum*. *Adam* mantiene y actualiza dos tipos de promedios móviles durante el entrenamiento de un modelo. Por un lado, calcula un promedio móvil de los gradientes, inspirado en el método *Momentum*, lo que permite al modelo aprender de forma más estable, siguiendo direcciones consistentes. Por otro lado, utiliza un promedio móvil del cuadrado de los gradientes, una idea tomada de *RMSProp*, lo que le permite emplear una tasa de aprendizaje adaptativa, dependiendo de la magnitud de los gradientes [31].

De forma analítica, *Adam* sigue los siguientes pasos en cada iteración k:

$$m_k \leftarrow \beta_1 m_k + (1-\beta_1)(\partial\theta_k) \quad \forall m_k, \tag{2.13}$$

$$s_k \leftarrow \beta_2 s_k + (1-\beta_2)(\partial\theta_k)^2 \quad \forall s_k, \tag{2.14}$$

donde m_k y s_k es el promediado móvil y el promediado móvil al cuadrado respectivamente en la iteración k. Los hiperparámetros β_1 y β_2 (generalmente 0,9 y 0,999) controlan la reducción exponencial. La reducción exponencial posee como efecto adverso el sesgo de los anteriores valores hacia 0 al inicio del entrenamiento de la red. Para remediar este efecto, se utilizan las siguientes correcciones:

$$\widehat{m_k} \leftarrow \frac{m_k}{1-\beta_1^k}, \quad \widehat{v_k} \leftarrow \frac{v_k}{1-\beta_2^k}. \tag{2.15}$$

Por último, la derivada del gradiente en cada iteración k viene dada como:

$$\theta_k \leftarrow \theta_k - \frac{\alpha}{\sqrt{\widehat{v_k}}+\varepsilon}\widehat{m_k}, \tag{2.16}$$

donde el hiperparámetro ε evita la división por 0, al que se le asocia un valor muy pequeño en torno a 10^{-8} [30].

El optimizador *Adam* ha demostrado su eficacia en diversas tareas de clasificación, superando en ocasiones a otros optimizadores. Un estudio analizó el desempeño de diversos optimizadores, como *RMSprop*, *Adam* y otros, en la

tarea de clasificar imágenes de hojas de plantas en 15 categorías diferentes. Los resultados muestran que Adam obtuvo un acierto del 98%, superando a los otros optimizadores tanto en términos de acierto como en velocidad de convergencia [32].

En otro trabajo [33], se evalúan diversas variantes del optimizador *Adam* y se comparan con métodos convencionales, como *Momentum*, en tareas de clasificación de imágenes utilizando arquitecturas de redes neuronales profundas como *AlexNet*, *VGGNet* y *ResNet*. Los experimentos demuestran que las variantes de *Adam* ofrecen un rendimiento significativamente superior en precisión y eficiencia en comparación con los optimizadores tradicionales. Sin embargo, *Adam* puede presentar limitaciones en la convergencia, especialmente en funciones no convexas o cerca de un óptimo local [34].

3 Composición musical automatizada

El afán por la creación y composición de música es tan antiguo como la historia misma. En el Antiguo Egipto ya se construían instrumentos de viento, cuerda y percusión utilizando los materiales disponibles. Además, la figura del músico gozaba de un gran reconocimiento social. Siglos después, la música volvió a estar presente a través de los monjes gregorianos en los diferentes monasterios de toda Europa, dando gran relevancia a la técnica vocal (canto gregoriano). La música barroca se perfiló durante el siglo XVII gracias a grandes compositores como Bach y Haendel en Alemania o Vivaldi en Italia. Esta forma de hacer música llegaría a su cúspide a mediados del siglo XVIII (Clasicismo) con su máximo exponente: Mozart. Asimismo, el lado más expresivo y sentimental de la música se desarrolló durante el siglo XIX, de la mano de músicos románticos como Beethoven o Chopin, que crearon algunas de las piezas musicales de piano más memorables.

La historia de la música dio un cambio drástico en el siglo XX gracias a los diferentes movimientos de vanguardia que se difundieron por Europa. Estos movimientos ayudaron a experimentar con la música y crear nuevas formas nunca vistas. Una pieza fundamental que ayudó a este desarrollo fue la llegada del computador durante la década de 1950, el cual ofrecía muchas posibilidades para crear y componer música.

El primer computador que se utilizó para reproducir música fue CSIRAC, el cual fue programado por Geoffrey Hill en la ciudad de Sídney [35]. El computador crea diferentes ritmos y melodías a través de las señales eléctricas binarias que emite en tiempo real. Estas señales son procesadas y enviadas a un amplificador conectado a un altavoz [36].

Los pioneros en usar algoritmos para la composición de música fueron Lejaren Hiller y Leonard Isaacson. Estos algoritmos se basan en emplear reglas estocásticas para generar los patrones musicales utilizados para la composición, tales como el ritmo o la armonía. En 1957, unieron sus esfuerzos para dar vida a la *Illiac Suite*, una pieza musical (cuarteto de cuerdas) que hizo historia al convertirse en la primera obra creada íntegramente por un ordenador: el *Illiac I*, de la Universidad de Illinois [37].

Otro precursor de la música por computador fue Max Matthews. Trabajó en los Laboratorios Bell donde desarrolló el programa MUSIC I en 1957. Este programa informático produce un audio a partir de una secuencia de notas que el usuario introduce, lo que se consideró revolucionario en su momento. Junto a su compañero Newmann Guttman, lo utilizó para crear *The Silver Scale*, considerada la primera pieza musical generada por computador. Asimismo, Matthews continuó implementando nuevas versiones de este programa en las que añadió nuevas funcionalidades para conseguir composiciones más elaboradas. Esta tecnología es la antecesora de los actuales sintetizadores de audio para la producción musical [38].

Max Matthews continuó investigando más acerca de este campo. En 1970, presentó junto al compositor Richard Moore un nuevo sistema para crear música llamado GROOVE. Según los autores: "*GROOVE es un sistema híbrido que interpone un ordenador digital entre un compositor-intérprete humano y un sintetizador de sonido electrónico. Todas las acciones manuales del ser humano son supervisadas por el ordenador y almacenadas en su memoria de disco.*" [39]. GROOVE actúa como los modernos controladores que usan los técnicos de sonido, ya que permite al usuario realizar muchas acciones en tiempo real.

Estas acciones consisten en mezclar diferentes secciones del audio o ajustarlas. Después, todos estos cambios se guardan en la memoria del computador para poder reproducir el audio actualizado.

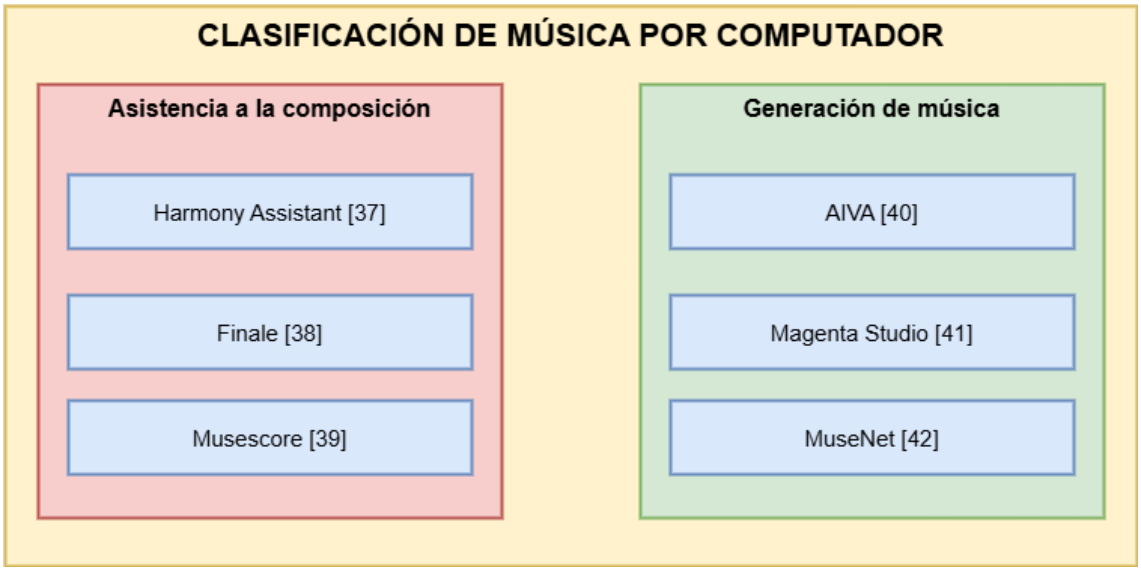


Figura 5. Diagrama que muestra la clasificación de los sistemas de música por computador en dos grandes grupos: herramientas de asistencia a la composición y algoritmos de generación de música.

Es posible dividir los equipos de música por computador en dos grandes grupos. El primero, dedicado a la asistencia en la creación, ofrece al compositor herramientas que, basadas en los acordes y las notas, le proponen progresiones melódicas que se adaptan al estilo de su obra. El segundo grupo corresponde a los algoritmos capaces de generar música nueva, los cuales suelen hacer uso de la Inteligencia Artificial. En la Figura 5, se pueden apreciar algunos programas software actuales según esta clasificación.

3.1 Fundamentos de música

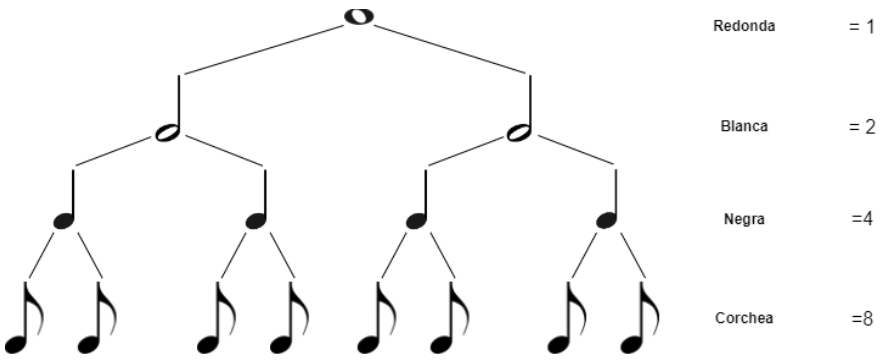


Figura 6. Árbol binario jerárquico que representa la subdivisión rítmica musical, donde cada nivel del árbol muestra cómo se divide el valor de una nota en partes más pequeñas equivalentes.

El objetivo de esta sección es sentar las bases del ámbito musical necesarias para representar fielmente una nota musical y sus relaciones con otras notas dentro de una melodía.

En primer lugar, conviene recalcar las cualidades del sonido en términos musicales. El sonido, partiendo de su origen ondulatorio, queda definido mediante las siguientes características:

- Tono (altura): hace referencia a la frecuencia de ese sonido en particular, lo que nos permite diferenciar entre sonidos agudos (altas frecuencias) y graves (bajas frecuencias).

- Duración: indica si un sonido es largo y corto. En notación musical, queda representado por las distintas figuras musicales que se pueden apreciar en la Figura 6. Las figuras musicales tienen una duración predeterminada, siendo la redonda la de mayor valor (4 unidades de tiempo), y las sucesivas tienen la mitad del valor de la figura anterior.

- Intensidad: distingue entre sonidos fuertes y suaves. En una pieza musical puede haber distintas notaciones para representar este aspecto. Desde un sonido suave (*piano*, *p*) hasta un sonido fuerte (*forte*, *f*).

- Timbre: esta característica es intrínseca a cada instrumento musical, incluida la voz humana. Por tanto, permite distinguir el sonido que produce, por ejemplo, un piano o un violín, ya que sus timbres son distintos y el oído humano es capaz de identificarlos.

En la música occidental, existen 7 notas principales (do, re, mi, fa, sol, la y si) que se organizan en escalas. Estas notas pueden ampliarse a un total de 12, gracias a la adición de semitonos entre los pares de notas si-do y mi-fa. Asimismo, es común utilizar el cifrado americano para representarlas, el cual consiste en asociar letras del alfabeto a las notas musicales, como por ejemplo C para do. La Figura 7 agrupa la anterior notación musical explicada y el cifrado americano en el pentagrama, el cual sirve de soporte para su representación.

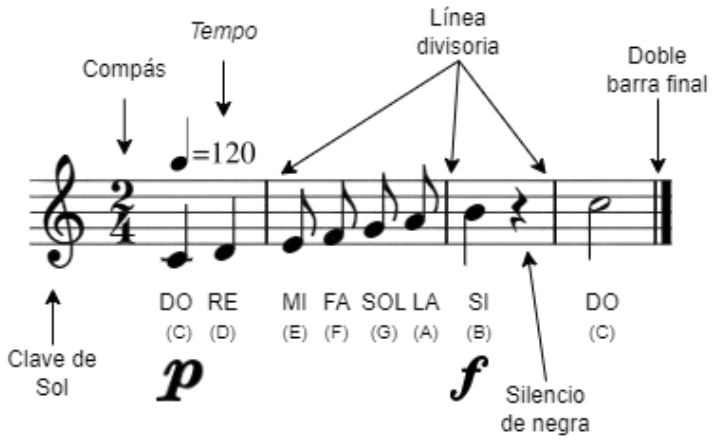


Figura 7. Pentagrama musical que muestra una melodía acompañada de notación representativa.

Otro aspecto a tener en cuenta es la clave musical. La clave es un signo que se usa como referencia para indicar la altura en específico de una nota. A partir de ella, se representan las demás notas. La clave más usada es la clave de Sol que indica que la nota sol va situada en la segunda línea del pentagrama. Como las notas se organizan en escalas consecutivas (do, re, mi, fa, sol, la y si) a diferentes alturas, resulta trivial posicionar las restantes. Otro símbolo que complementa a las notas son los silencios, que se utilizan para marcar pausas en las que no se emite ningún sonido. Además, para cada figura musical, existe un silencio equivalente que tiene la misma duración.

Una notación que es fundamental para establecer la duración de una melodía es el compás. El compás indica la partición por grupos de las figuras musicales, separados por una la línea divisoria en el pentagrama. Se representa a través de una fracción, donde el numerador indica el número de tiempos que consta cada compás, mientras que el denominador representa el tipo de figura. Por ejemplo, el compás 2/4 significa que en cada compás caben 2 negras, o cualquier combinación equivalente (véase Fig. 6). Asimismo, otra notación que tiene especial influencia en la duración de una melodía es el *Tempo*, el cual indica la velocidad en la que una obra musical tiene que interpretarse. El Tempo viene marcado por las pulsaciones por minuto (*bpm*, *beats per minute*). La figura musical acompañada por una indicación de *bpm* nos señala qué valor de nota dura una pulsación (véase Fig. 7).

Por último, un aspecto que cobra gran relevancia en música es la armonía La armonía se refiere a la combinación de varios sonidos que se producen simultáneamente (acordes), creando una sensación de equilibrio y coherencia. Su objetivo es estudiar las relaciones entre las diferentes notas que se tocan al mismo tiempo, buscando que su interacción resulte agradable, aportando riqueza al conjunto melódico. Los acordes de tríada, formados por tres notas, se construyen a partir de intervalos, que representan la distancia tonal entre las notas. Un intervalo relevante es la octava, la cual es la distancia entre dos notas del mismo nombre. Las octavas se relacionan porque la frecuencia de la nota superior es el doble de la inferior, lo que permite posicionar los distintos sonidos graves y agudos en un mismo marco tonal. Estos intervalos definen la estructura del acorde: mayor, menor, disminuido, aumentado, de quinta o de séptima; entre otros. Además, los acordes se pueden representar mediante el cifrado americano, los cuales utilizan la nota fundamental, llamada tónica, para identificar el acorde dentro de una progresión. En este caso, la tónica se acompaña de símbolos que describen el tipo de acorde: mayor (sin símbolo, por defecto), menor (*m*), entre otros. En la Figura 8, se puede observar una progresión armónica anotada con este tipo de cifrado, mostrando cómo se identifican los acordes.

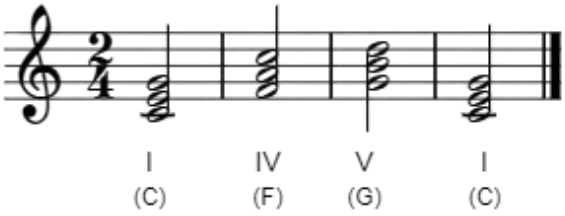


Figura 8. Representación de una progresión armónica que incluye distintos acordes. Los números romanos indican la posición de la tónica de cada acorde dentro de la escala musical, permitiendo identificar el tipo de acorde en cada momento.

3.2 Representación por computador

A pesar de que la partitura es el medio más utilizado para escribir y transmitir música, resulta ineficiente a la hora de trabajar con una melodía en computador. De esta manera, existen diversas herramientas que permiten representar los aspectos musicales anteriormente comentados.

MIDI

El formato ampliamente utilizado en sistemas informáticos para representar música es MIDI (*Musical Instrument Digital Interface*). MIDI es un protocolo de comunicación entre distintos dispositivos que hace uso de una interfaz para la transmisión de datos [46]. Cuando la comunicación se ha establecido, se empiezan a intercambiar mensajes. Estos mensajes se componen de bytes que contienen información acerca del audio.

Los mensajes del estándar MIDI se dividen en dos conjuntos principales: los mensajes de sistema, que incluyen comandos para controlar la posición de la canción, su estado de reproducción y mensajes relacionados con las características específicas del hardware utilizado, y los mensajes de canal. Estos mensajes pueden ser transmitidos por 16 canales diferentes (del 0 al 15), lo que es de gran utilidad para tratar pistas de audio individualmente [47]. Dentro de este último se encuentra la información descriptiva acerca de cada nota musical del audio. Los datos más relevantes que incluye son la velocidad de la nota, el Tempo, la intensidad, cuándo la nota es tocada (*key on*) y cuándo deja de estarlo (*key off*) [46]. Este último dato se codifica mediante 7 bits (rango entre 0 y 127), que indica el tono en específico de la nota [48]. En la Figura 9 se puede observar las relaciones entre estos valores y el nombre de la nota que corresponde a ese tono.

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

Figura 9. Representación en formato MIDI de las notas musicales.

La popularidad de este formato para la representación de música ha originado una gran cantidad de recursos web que albergan piezas musicales en este formato [49]. Este hecho es de especial relevancia porque ayuda a crear conjuntos de datos compuestos por estas melodías que se pueden aplicar al entrenamiento de diferentes modelos. De esta manera, estos modelos pueden aprender a generar música con distintos instrumentos a partir de estos archivos en formato MIDI [50] [51].

Piano roll

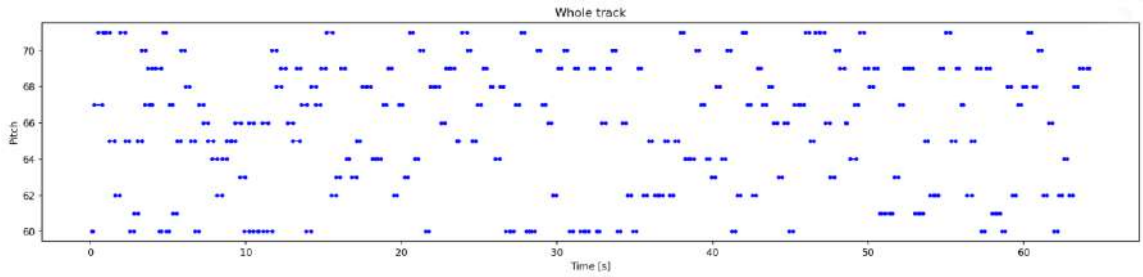


Figura 10. Representación de una pieza musical mediante piano roll.

La *pianola* es un instrumento musical creado a principios del siglo pasado que permite reproducir de forma automática melodías que están constituidas a través de la técnica de *piano roll*. Esta técnica consiste en una serie de perforaciones sobre un papel, las cuales representan las notas musicales. La *pianola* posee un mecanismo capaz de detectar estos huecos y reproduce el sonido correspondiente [52]. Esta tecnología ha ido evolucionando y se ha adaptado al soporte digital. La Figura 10 muestra esta representación en formato digital. Sin embargo, un inconveniente importante de esta forma de representación es la falta de información sobre la duración de cada nota [53], a diferencia del formato MIDI, que sí recoge esta información mediante los eventos *key on* y *key off*.

A pesar de ello, su uso es bastante común en muchos proyectos dentro de la literatura sobre la generación de música. Modelos probabilísticos difusos se han utilizado para crear música a partir de *piano rolls* [54]. Por otra parte, se ha estudiado el problema de modelizar secuencias de polifonía usando también esta técnica mediante redes neuronales recurrentes [55].

3.3 Redes neuronales recurrentes

La principal motivación que subyace al estudio de las redes neuronales recurrentes (*RNN*) es su capacidad para predecir el resultado siguiente dada una serie temporal de entrada, la cual se refiere a datos que varían en función del tiempo [56]. Un caso práctico de uso es el campo del mercado bursátil para predecir cómo varía el valor de la acción de una determinada empresa a futuro. De esta manera, se pueden identificar patrones temporales y proyectar tendencias futuras, ofreciendo a los inversores un mapa probabilístico de fluctuaciones. Por otro lado, también son especialmente útiles para tareas creativas como la generación de música, donde la serie temporal es una secuencia de notas musicales. El objetivo es predecir la nota siguiente y así generar propuestas melódicas coherentes que respetan la estructura temporal de la pieza original. Un ejemplo ilustrativo es el proyecto Magenta de Google, donde modelos recurrentes han compuesto obras completas imitando estilos

desde el barroco hasta el jazz moderno, demostrando cómo la memoria temporal de estas redes captura tanto patrones técnicos como expresividad artística [57].

El precursor de las redes recurrentes fue el físico John Hopfield, que desarrolló esta idea en su artículo *Neural networks and physical systems with emergent collective computational abilities*. En este trabajo, implementa un tipo de red neuronal recurrente binaria, denominada red de Hopfield, que al rebasar cierto umbral, devuelve un valor u otro [58].

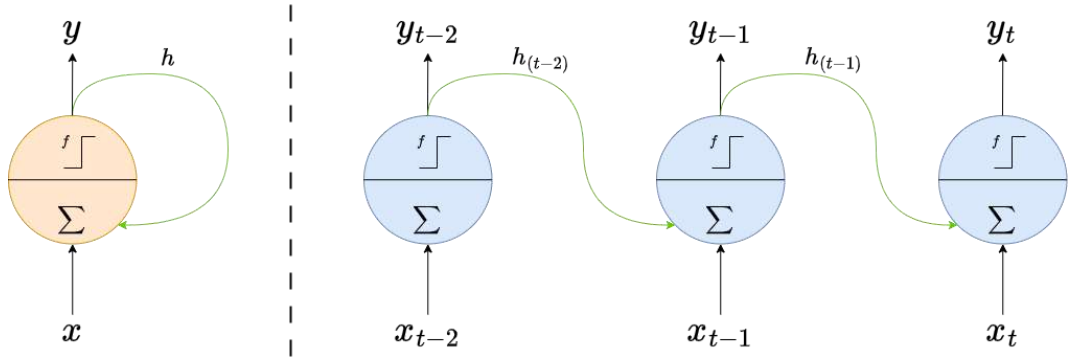


Figura 11. Diagrama del funcionamiento de una neurona recurrente. A la izquierda, la estructura original con retroalimentación (recurrente). A la derecha, esta neurona "desenrollada" a lo largo del tiempo.

La estructura de las redes recurrentes no se organiza en capas, a diferencia de lo que ocurre con las redes de neuronas alimentadas hacia delante, sino que, por el contrario, una neurona se puede conectar con cualquier otra, incluso consigo misma. Esta estructura permite que la información fluya hacia atrás, creando un vínculo con otras neuronas y facilitando el procesamiento a lo largo del tiempo. La Figura 11 muestra el flujo de información de una neurona recurrente a lo largo del tiempo. Gracias a característica la posibilidad de la retroalimentación, las redes recurrentes son capaces de retener y utilizar datos de secuencias previas y la información actual. También es posible definir una capa recurrente, denominada célula o celda de memoria (*memory cell*) que recibe como entrada tanto el vector de entrada actual como el vector de salida en el instante de tiempo anterior. Además, es importante considerar que a estos vectores se les asignan los pesos correspondientes y se les aplica la función de activación no lineal previamente definida para todas las instancias del lote (*mini-batch*) que se está procesando en ese momento. (3.1) indica los cálculos a realizar:

$$\mathrm{Y(t)} = \mathrm{f_{activation}}\big(\mathrm{X(t)W_x} + \mathrm{Y(t-1)W_y} + \mathrm{b}\big), \tag{3.1}$$

El término celda de memoria se debe a que el conjunto de neuronas que la componen actúa de cierta manera como una memoria que recibe la salida anterior y genera la siguiente [59]. Por tanto, debe "recordar" lo que ha pasado en el instante de tiempo anterior. De manera similar, el estado oculto de una celda (*hidden state*) en un momento específico t, representado como $h_{(t)}$, es una función que depende tanto de las entradas en ese instante de tiempo $x_{(t)}$ como del estado de la celda en el instante anterior $h_{(t-1)}$ [60]. Analíticamente, considerando (3.1), se expresa como:

$$h_{(t)} = f\big(h_{(t-1)}, x_t\big) \tag{3.2}$$

Es importante destacar que en las *RNN*, la salida $Y(t)$ de (3.1) es equivalente al estado oculto $h_{(t)}$, ya que ambos representan el resultado producido por la celda en el instante t.

Las celdas de memoria de las *RNN* manifiestan dificultades a la hora de poder recordar información pasada después de haber procesado una secuencia larga de datos. Es decir, este tipo de redes solo tienen en cuenta datos pasados a corto plazo (*short-term memory)*, lo que ocasiona que estas redes neuronales sean efectivas solo en un rango muy limitado de elementos de la secuencia [61]. El origen de esta desventaja radica en el algoritmo de retropropagación del gradiente para el caso específico de *RNN*, llamado algoritmo de propagación hacia atrás en el tiempo (*backpropagation through time*). Este algoritmo requiere "desenrollar" la red a lo largo de múltiples pasos temporales, lo que provoca el mismo problema que en redes de neuronas profundas alimentadas hacia delante con funciones de activación sigmoidales: el desvanecimiento o desaparición de los gradientes (*vanishing gradients problem*) [62]. Cuando se trata una secuencia extensa en el tiempo, la entrenada neta total de las neuronas tiende a ser alta en valor absoluto, lo que provoca que la derivada de la función de activación tienda a cero. Entonces, el gradiente del error es prácticamente nulo, sobre todo en las primeras capas, por lo que los pesos no se actualizan. Este efecto provoca que la red neuronal recurrente no sea capaz de recordar las entradas lejanas al momento actual, sino únicamente las más recientes, lo que se conoce como *short-term memory*.

3.3.1 Long Short-Term Memory

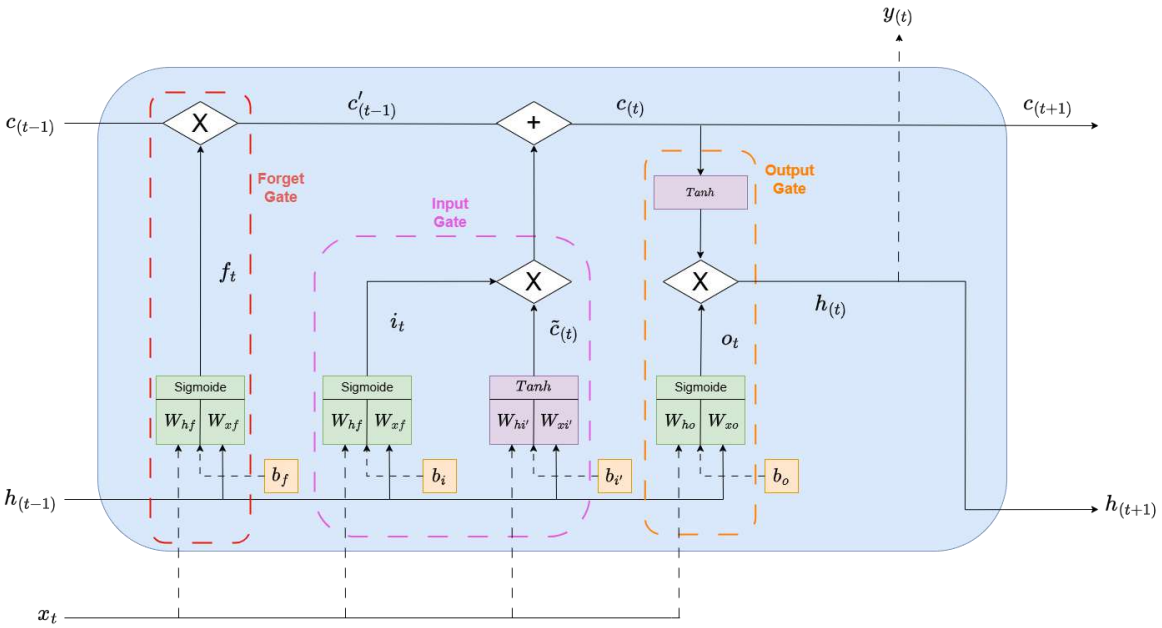


Figura 12. Estructura interna de una celda LSTM.

La memoria larga a corto plazo (*Long Short-Term Memory*, *LSTM*) es un tipo de red neuronal recurrente que surge para resolver el principal problema que manifiestan las *RNN* clásicas: su imposibilidad para recordar información pasada en secuencias de datos extensas, desarrollada por los informáticos Hochreiter y Schmidhuber en 1997 [63]. *LSTM* introduce unidades de memoria que permiten a la red aprender dependencias temporales a largo plazo. En

concreto, divide su estado interno en dos vectores denominados $h_{(t)}$ y $c_{(t)}$, donde el primer vector corresponde a la información almacenada a corto plazo (denominado estado oculto, *hidden state)*, mientras que el segundo corresponde a la memoria de largo plazo (denominado estado de la celda, *cell state*).

La idea es integrar correctamente $c_{(t)}$ en la memoria a corto plazo $h_{(t)}$ con el objetivo de que $c_{(t)}$ influya en la predicción de los nuevos elementos secuenciales. En una celda *LSTM*, $c_{(t)}$ se actualiza a medida que la celda la procesa, y mediante diferentes puertas (gates) se controla qué información añadir, eliminar o conservar en el flujo que determina la salida de la celda. La Figura 12 muestra la estructura de una celda *LSTM* junto a las puertas que la componen. La información $x_{(t)}$ que entra a la celda se procesa a través de tres etapas, cada una correspondiente a una puerta específica de la arquitectura: la puerta de olvido (*forget gate*), la puerta de entrada (*input gate*) y la puerta de salida (*output gate*). Así mismo, el estado anterior, que almacena la información memorizada en pasos previos, se representa como $h_{(t-1)}$ y $c_{(t-1)}$.

En la primera etapa del proceso, la celda *LSTM* evalúa cuánto del estado de memoria a largo plazo $c_{(t-1)}$ debe conservarse en el instante actual t. Para ello, combina las entradas $x_{(t)}$ y $h_{(t-1)}$ mediante una operación lineal que involucra un conjunto de pesos y un sesgo ($\mathrm{W_{xf}}$, $\mathrm{W_{hf}}$, y b_f). Posteriormente, se aplica la función de activación sigmoide σ al resultado de esta operación, obteniendo el valor $f_\mathrm{t} \in [0,1]$, que representa el porcentaje del estado $c_{(t-1)}$ que debe olvidarse o mantenerse. (3.3) indica las operaciones que se realizan:

$$\mathrm{f_t} = \sigma\big(\mathrm{x_{(t)}W_{xf}} + \mathrm{h_{(t-1)}W_{hf}} + \mathrm{b_f}\big), \tag{3.3}$$

de donde finalmente se calcula el producto matricial de f_t con $c_{(t-1)}$, obteniendo la parte actualizada del estado de memoria a largo plazo previo:

$$c'_{(t-1)} = f_t \otimes c_{(t-1)}.$$

En segundo lugar, la celda *LSTM* evalúa qué nueva información proveniente del estado oculto debe añadirse al estado de la celda previo para obtener el correspondiente al instante actual $c_{(\mathrm{t})}$. El proceso es similar al de la etapa anterior, combinando diferentes pesos y un sesgo (W_{xi}, W_{hi} y b_i) con $x_{(t)}$ y $h_{(t-1)}$. Se aplica la función de activación sigmoide al resultado, lo que genera el valor $i_{(t)}$. Este valor indica qué proporción de la nueva información se incorpora al estado de memoria $c'_{(t-1)}$. Así mismo, se calcula un candidato para actualizar el estado de memoria $\tilde{\mathrm{c}}_\mathrm{t}$, aplicando una función de activación tangente hiperbólica *tanh* sobre otra combinación lineal de $x_{(t)}$ y $h_{(t-1)}$ con pesos y un sesgo ($W_{xi'}$, $W_{hi'}$, y b_i). (3.4) y (3.5) muestran estos cálculos:

$$i_t = \sigma\big(x_{(t)}W_{xi} + h_{(t-1)}W_{hi} + b_i\big), \tag{3.4}$$

$$\tilde{c_t} = \tanh\big(x_{(t)}W_{xi'} + h_{(t-1)}W_{hi'} + b_{i'}\big), \tag{3.5}$$

de donde finalmente se determina la contribución efectiva de la nueva información al estado de memoria a largo plazo, que luego se suma a $c'_{(t-1)}$ para completarlo:

$$c_{(t)} = c'_{(t-1)} \oplus (i_t \otimes \tilde{c_t}).$$

En la siguiente etapa del proceso, la celda *LSTM* determina qué parte del estado de memoria a largo plazo $c_{(t)}$ contribuye al estado oculto actual $h_{(t)}$. El proceso

es similar al de la anterior etapa, combinando $x_{(t)}$ y $h_{(t-1)}$ mediante una operación lineal que involucra un conjunto de pesos y un sesgo (W_{xo}, W_{ho}, y b_o). Se aplica la función de activación sigmoide al resultado, generando el valor $o_{(t)}$. Este valor controla qué parte de la información contenida en el estado de memoria $c_{(t)}$ se utiliza para calcular el estado oculto $h_{(t)}$. A continuación, $c_{(t)}$ se transforma aplicándole la función de activación tangente hiperbólica para regular los valores y mantenerlos en un rango adecuado, lo que genera $c_{(t+1)}$, que representa una versión regularizada del estado de memoria que se utiliza en el siguiente instante temporal.

Finalmente, el estado oculto $h_{(t)}$ se calcula como el producto matricial entre $o_{(t)}$ y $c_{(t+1)}$. Este estado oculto $h_{(t)}$ cumple una doble función: por un lado, actúa como la salida actual de la celda *LSTM* hacia la siguiente celda de la red ($y_{(t)}$), y por otro, se utiliza como entrada para el siguiente instante temporal $\left(h_{(t+1)}\right)$, manteniendo así la continuidad en el flujo de información. (3.6) y (3.7) muestran los cálculos a realizar:

$$\mathrm{o}_t = \sigma\left(x_{(t)}W_{xo} + h_{(t-1)}W_{ho} + b_{\mathrm{o}}\right), \tag{3.6}$$

$$\mathrm{y_t} = \mathrm{h_{t+1}} = \mathrm{h_t} = \mathrm{o_t} \otimes \mathrm{c_{t+1}}. \tag{3.7}$$

La estructura y funcionamiento interno de una celda *LSTM* se puede observar en la Figura 12. Asimismo, se ha de tener en cuenta que los diferentes pesos y sesgos utilizados en la estructura se aprenden durante la fase de entrenamiento de la celda.

La arquitectura *LSTM* es esencial en la generación de música automatizada como así lo avalan múltiples artículos que conforman la literatura existente. El científico Michael Conner junto a su equipo ahondaron en el uso de redes *LSTM* en el ámbito musical [64], destacando su capacidad para modelar secuencias y predecir notas musicales. Los autores presentan una red neuronal diseñada para generar música, evaluando su rendimiento en términos de ritmo, armonía y precisión gramatical. Además, el artículo proporciona una visión detallada de la intuición, teoría y aplicación de las *LSTM* en la generación musical, junto con los desafíos enfrentados y posibles mejoras futuras para el modelo.

Por otro lado, el ingeniero Hooman Rafraf propuso un nuevo enfoque en este campo [65]. Su artículo introduce un modelo de inteligencia artificial generativa para la composición musical automatizada utilizando redes *LSTM*. El modelo codifica la información musical basada en intervalos melódicos y armónicos, en lugar de notas absolutas. Los resultados experimentales muestran que las composiciones generadas son musicales y tonales, aunque presentan algunas modulaciones excesivas, atribuibles a la naturaleza de la codificación utilizada.

3.3.2 Gated Recurrent Unit

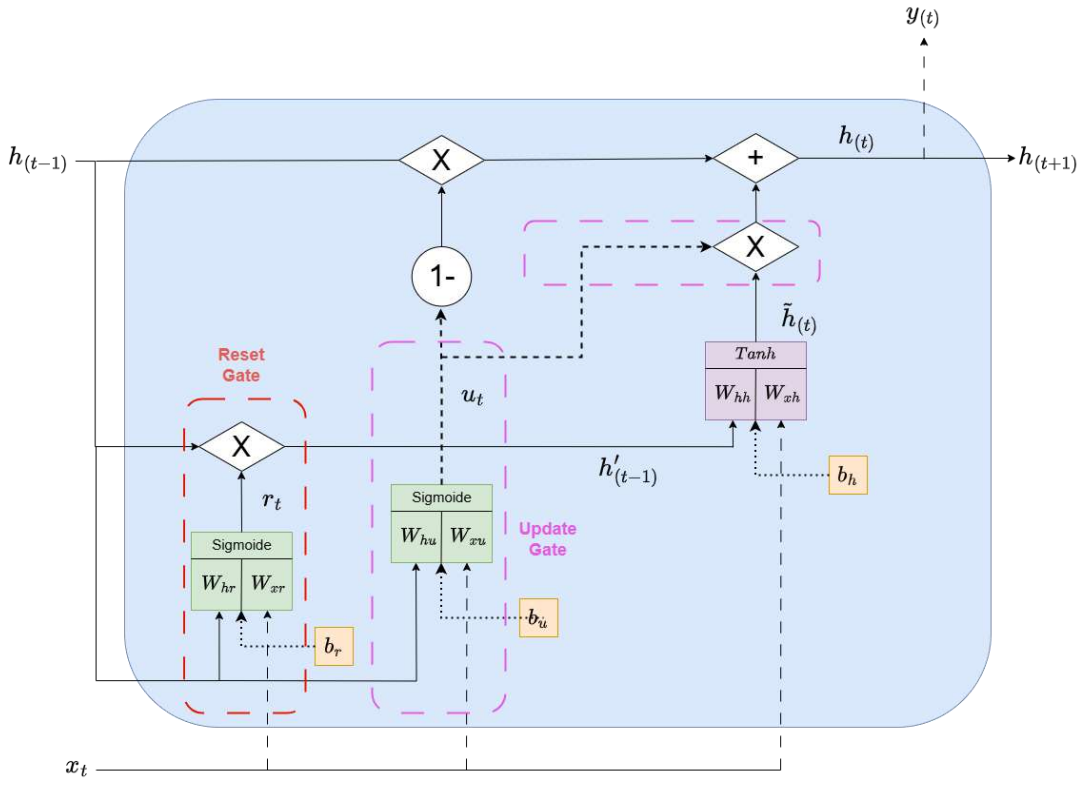


Figura 13. Estructura interna de una celda GRU.

La arquitectura Gated Recurrent Unit (*GRU)* fue introducida por Kyunghyun Cho et al. en 2014 [66]. Este tipo de *RNN* es similar a *LSTM*, capaz de manejar y seleccionar la información a olvidar o recordar a lo largo del tiempo.

La distinción clave entre las celdas *GRU* y las *LSTM* reside en cómo gestionan el estado de memoria. En las *LSTM*, el estado de memoria se mantiene separado del estado oculto y se ajusta mediante tres tipos de puertas: la de olvido, la de entrada y la de salida. En contraste, las *GRU* sustituyen el estado de memoria por un vector de activación candidato $\tilde{h}_t$, que se actualiza a través de dos mecanismos: la puerta de reinicio (*reset gate*), encargada de determinar cuánto del estado oculto previo $h_{(t-1)}$ debe descartarse, y la puerta de actualización (*update gate*), que decide qué cantidad de $\tilde{h}_t$ debe incorporarse al nuevo estado oculto $h_{(t)}$ [67].

El funcionamiento y estructura interna de la celda *GRU* se muestra en la Figura 13. Se puede descomponer en 3 etapas. En primer lugar, se calculan el valor resultante de la puerta de reinicio r_t y la puerta de actualización u_t. Para ello, se combinan las entradas $x_{(t)}$ y $h_{(t-1)}$ mediante una operación lineal que involucra un conjunto de pesos y un sesgo por cada puerta a calcular (W_{xr}, W_{hr}, b_r, W_{xu}, W_{xu}, b_u). Posteriormente, se aplica la función de activación sigmoide σ al resultado, lo que genera los valores r_t y u_t. (3.8) y (3.9) muestran estas operaciones:

$$r_t = \sigma\left(x_{(t)}W_{xr} + h_{(t-1)}W_{hr} + b_r\right), \tag{3.8}$$

$$u_t = \sigma\left(x_{(t)}W_{xu} + h_{(t-1)}W_{hu} + b_u\right). \tag{3.9}$$

En la segunda etapa, $\widetilde{h_t}$ se genera utilizando la entrada actual $x_{(t)}$ y una versión ajustada del estado oculto previo denotada como $h^{'}_{(t-1)}$. Este ajuste es fruto de la acción de la puerta de reinicio, que "reinicia" el estado oculto previo al modificarlo mediante un producto matricial. (3.10) y (3.11) muestran este cálculo:

$$h^{'}_{(t-1)} = r_t \otimes h_{(t-1)}, \tag{3.10}$$

$$\widetilde{h_t} = tanh\left(x_{(t)}W_{xh} + h^{'}_{(t-1)}W_{hh} + b_h\right), \tag{3.11}$$

donde W_{xh}, W_{hh} y b_h son dos matrices de pesos y un sesgo, respectivamente, que se ajustan durante el entrenamiento.

En la última etapa, el nuevo estado oculto h_t se obtiene combinando el estado oculto previo $h_{(t-1)}$ y el vector de activación candidato $\widetilde{h_t}$, ponderados por la puerta de actualización u_t. Al igual que ocurre en *LSTM*, h_t actúa como la salida actual de la celda *LSTM* hacia la siguiente celda de la red ($y_{(t)}$), y por otro, se utiliza como entrada para el siguiente instante temporal $\left(h_{(t+1)}\right)$.

(3.12) describe los cálculos a realizar:

$$y_t = h_{(t+1)} = h_t = \left\{(1 - u_t) \otimes h_{(t-1)}\right\} \oplus \left(u_t \otimes \widetilde{h_t}\right) \tag{3.12}$$

La arquitectura *GRU* también ostenta un papel notorio en la literatura acerca de la generación de música, como en el estudio realizado por A. A. Gunawan et al. [68]. En este artículo se utilizan archivos MIDI como entrada para el entrenamiento de modelos usando celdas *GRU* y *LSTM* con el objetivo de crear patrones musicales característicos. Además, se destaca la eficacia y el desempeño del modelo *GRU* con dos capas apiladas para generar música agradable y atractiva mediante una evaluación tanto objetiva como subjetiva.

En el artículo titulado *Music Generation using RNN-LSTM with GRU* [69], se aborda el uso de *RNN* en la creación automática de música, enfocándose específicamente en las arquitecturas *LSTM* y *GRU*. El estudio tiene como objetivo comparar el rendimiento de los modelos *LSTM* y *GRU* en la generación de composiciones musicales que sean armónicas y agradables al oído. Los resultados muestran que, si bien tanto las *LSTM* como las *GRU* son capaces de producir secuencias musicales convincentes, las *GRU* destacan por su mayor eficiencia computacional y tiempos de entrenamiento más cortos. Esto indica que las *GRU* constituyen una alternativa eficiente a las *LSTM* para tareas de generación musical.

3.4 Modelos neuronales generativos

Los modelos generativos se distinguen por su capacidad para producir datos nuevos, denominados sintéticos que, aunque no forman parte del conjunto original, conservan las relaciones y patrones subyacentes presentes en los datos de entrenamiento. Para lograr esto, el modelo aprende la función de distribución del conjunto de datos, lo que le permite generar contenido que no es una mera copia, sino una interpretación coherente y novedosa basada en la estructura aprendida. De esta manera, los modelos generativos suelen utilizarse en tareas que involucran a la creatividad humana [70].

3.4.1 Red Generativa Adversaria

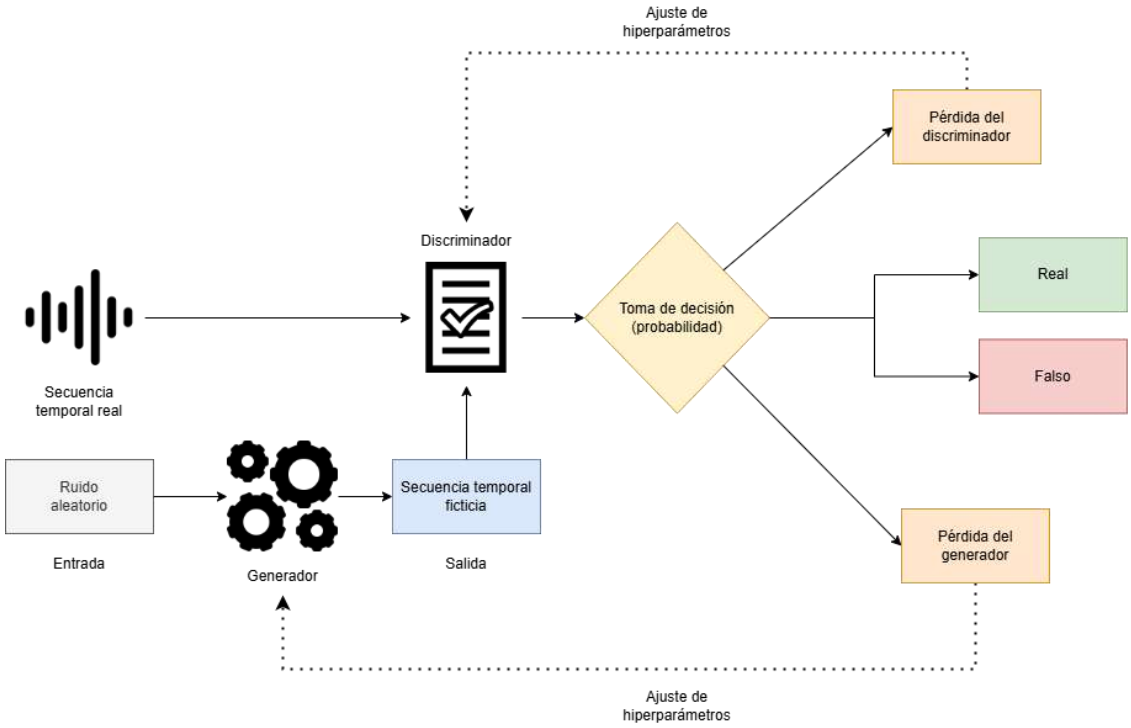


Figura 14. Diagrama que muestra la arquitectura de una GAN con secuencias temporales.

Las redes generativas adversarias (GAN) fueron propuestas inicialmente en 2014 por Goodfellow *et al.* [71]. Como muestra la Figura 14, su arquitectura aplicada a series temporales integra dos redes neuronales con funciones opuestas. La primera, llamada generador, procesa valores aleatorios (normalmente con distribución uniforme o gaussiana) para crear datos sintéticos que imiten la estructura estadística del conjunto de entrenamiento. Durante este proceso, el modelo ajusta sus parámetros, transformando progresivamente el ruido inicial en muestras que replican los patrones de los datos reales. La segunda red, conocida como discriminador, opera como un verificador. Su objetivo es distinguir entre las muestras auténticas del entrenamiento y las generadas artificialmente. El éxito del sistema radica en que el generador mejore hasta producir datos tan convincentes que el discriminador no pueda diferenciarlos de los reales, momento en que se considera completado el entrenamiento. Ambas redes emplean funciones de pérdida independientes para optimizar sus parámetros. Sin embargo, el generador basa sus ajustes en la retroalimentación indirecta proporcionada por los errores de clasificación del discriminador, estableciendo así un ciclo de mejora mutua durante el aprendizaje.

El campo de aplicaciones de esta arquitectura es bastante extenso, destacando sobre todo en áreas relacionadas con el tratamiento de imágenes. Por ejemplo, es recalcable su uso en la visión por computador para tareas como la segmentación [72], la mejora de la resolución en imágenes [73] o la creación de imágenes provenientes de entradas en formato textual [74].

No obstante, hay también numerosas contribuciones que utilizan esta arquitectura en la composición de música. El modelo *MuseGAN* se presentó por Dong *et al.* [75]. Este equipo desarrolló tres modelos basados en *GAN*s (modelo de *jamming*, de composición e híbrido) con el objetivo de conseguir producir música proveniente de cinco instrumentos musicales: bajo, batería, guitarra, piano e instrumentos de cuerdas. El conjunto de datos de entrenamiento de la red constaba de canciones de estilo rock.

Otro ejemplo es el modelo *GANSynth* implementado por Jessen Engel *et al.* [76]. Los autores propusieron este modelo para demostrar que la arquitectura GAN obtiene mejores resultados a la hora de generar música de una manera coherente. El modelo se basa en una *GAN* progresiva, en la cual a través de convoluciones se muestrea de forma incremental la fuente de audio hasta obtener el sonido completo. El conjunto de datos utilizado para el entrenamiento del modelo fue *NSynth*, que incluye 300.000 notas distintas generadas por 1.000 instrumentos diferentes, cada uno con sus propios timbres y tonos. Aunque el procesamiento de este conjunto es especialmente complejo debido a su diversidad, se encuentra muy bien estructurado, ya que tiene etiquetas con información detallada sobre las cualidades acústicas de las melodías.

Entre las ventajas de este tipo de arquitectura destaca la versatilidad del generador. El generador puede aceptar diversas fuentes de ruido, como distribuciones multivariadas o distribuciones no estacionarias, lo que le permite adaptarse a diferentes escenarios y facilitar el aprendizaje de distribuciones complejas. El modelo *BigGAN* es una arquitectura que maneja distribuciones multivariadas sobre datos de alta dimensión, como imágenes de alta resolución [77]. Por otro lado, *TimeGAN* combina redes generativas adversarias con componentes de modelado temporal para capturar dependencias no estacionarias en series temporales [78]. Además, no existe limitación alguna en el tamaño de la entrada del generador, lo que le concede una gran flexibilidad para modelar estructuras de datos de distinta dimensionalidad y ajustarse a diversas aplicaciones sin necesidad de modificar su arquitectura básica. Sin embargo, sus principales desventajas son: la inestabilidad del proceso de entrenamiento del modelo cuando el generador y el discriminador no logran converger y la poca variedad en los datos sintéticos que se generan [77].

3.4.2 Transformers

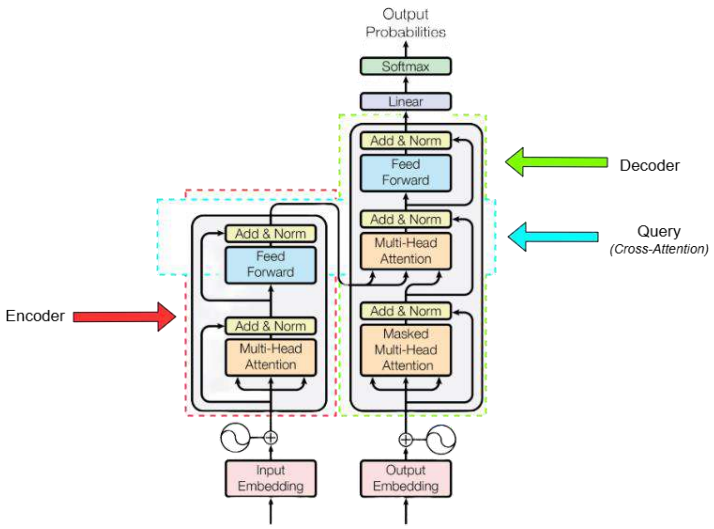


Figura 15. Arquitectura del modelo Transfomer [79].

El concepto de *Transformer* dentro del área de la inteligencia artificial generativa fue propuesto por un equipo de científicos que trabajaban para Google en 2017 [79]. Su arquitectura se muestra en la Figura 15, donde se aprecian dos componentes conectados: el codificador (*encoder*) y el decodificador (*decoder*).

El codificador tiene como objetivo principal procesar secuencias de entrada (una oración en español) para generar representaciones numéricas detalladas de cada palabra. Estas representaciones no solo conservan el significado léxico individual de los términos (por ejemplo, "perro" como animal doméstico), sino también las relaciones contextuales que mantienen con el resto de la secuencia. Para lograr esto, cada capa del codificador integra dos componentes. El primero es el mecanismo de atención multicabezal (*multi-head attention)*, que funciona identificando conexiones contextuales entre palabras. Por ejemplo, en una frase como *"El perro y el gato juegan; ellos siempre se divierten"*, este mecanismo detecta que el pronombre "ellos" se refiere a "perro y gato", incluso si están separados por varias palabras. El segundo componente es una red neuronal *alimentada hacia delante*, que ajusta y refina estas representaciones mediante transformaciones no lineales. Esta red permite capturar patrones complejos o abstractos que los mecanismos de atención no identifican directamente, como matices de ironía o ambigüedades sutiles. A diferencia de las redes recurrentes tradicionales (*RNN*), que procesan las palabras secuencialmente, el codificador opera en paralelo. Esta capacidad de procesamiento simultáneo permite que palabras distantes en la secuencia, como "ellos" y "perro y gato", interactúen directamente sin depender de pasos intermedios. Por ejemplo, en una frase extensa como *"A pesar de vivir en casas diferentes, el perro de la esquina y el gato del parque juegan juntos todas las tardes; ellos son inseparables"*, el codificador vincula "ellos" con "perro" y "gato" de manera inmediata, gracias a su paralelismo.

El decodificador genera secuencias de salida de manera autorregresiva, es decir, cada nueva palabra se genera en función de las anteriores, permitiendo mantener la coherencia y estructura del texto. Su diseño incorpora dos mecanismos de atención especializados. El primero es la atención multicabezal *enmascarada* (masked *multi-head attention*). Este mecanismo

aplica una máscara para restringir el acceso a posiciones futuras en la secuencia de salida, garantizando que la predicción de cada palabra dependa exclusivamente de las posiciones anteriores. Este enfoque evita que el modelo utilice información que aún no se ha generado, un requisito clave en tareas como traducción o generación de texto. El segundo mecanismo es la atención cruzada (*cross-attention*), que conecta el decodificador con las representaciones contextuales producidas por el codificador mediante un proceso de consulta (*query*) sobre sus salidas (véase *query* en Fig.15). Esto permite alinear elementos de la secuencia de entrada con la de salida. Por ejemplo, al traducir *"ellos"* al inglés (*"they"*), *cross-attention* identifica que este pronombre debe vincularse semánticamente con *"perro"* y *"gato"* en la entrada, preservando la coherencia referencial en el idioma objetivo. Acto seguido, se vuelve a ejecutar una red de neuronas *alimentada hacia delante*. Finalmente, la secuencia procesada recibe una transformación lineal que proyecta las representaciones de alta dimensión al espacio del vocabulario de salida. Finalmente se emplea la función *Softmax* para generar una distribución de probabilidad sobre todas las palabras posibles. Por ejemplo, en la traducción de *"ellos"*, *Softmax* asignaría una probabilidad alta a *"they"* en inglés, mientras suprimiría opciones incongruentes como *"it"* o *"she"*. Este proceso se repite iterativamente, generando cada palabra de la secuencia de salida en función del contexto acumulado, hasta completar la predicción.

La cooperación coordinada entre codificador y decodificador es esencial. Mientras el codificador sintetiza el contexto global de la entrada, el decodificador utiliza este conocimiento para generar una salida precisa y contextualizada. Para que ambos componentes trabajen de forma eficiente, es necesario convertir primero el texto en representaciones numéricas (*embeddings*). Estas codificaciones permiten organizar las palabras en un espacio donde términos con significados similares se agrupan, facilitando que el modelo pueda descubrir relaciones entre ellas. [80]. Asimismo, esta interdependencia entre ambos se potencia mediante la *multi-head attention*, un mecanismo que divide la atención en múltiples subprocesos paralelos (*cabezas*). Cada cabeza captura patrones distintos: una podría identificar relaciones gramaticales (como concordancia de género), otra referencias pronominales, y otra conexiones semánticas de largo alcance. Al concatenar estos resultados, el modelo integra perspectivas diversas, superando limitaciones de enfoques unificados. Analíticamente, esta operación se puede expresar como:

$$\mathrm{MultiHead(Q,K,V)} = \mathrm{Concat(head_1, \dots, head_h)W^O},$$

$$\mathrm{head}_i = \mathrm{Attention}\left(\mathrm{QW}_i^{\mathrm{Q}}, \mathrm{KW}_i^{\mathrm{K}}, \mathrm{VW}_i^{\mathrm{V}}\right),$$

donde $\mathrm{W_i^Q}$, $\mathrm{W}_i^{\mathrm{K}}$ y $\mathrm{W}_i^{\mathrm{V}}$ son las matrices de proyección para capturar los distintos patrones en un espacio dimensional reducido de dimensión $\mathrm{d_q}$, d_k y d_v sobre el espacio del modelo d_{model} en cada una de las *cabezas* h.

El núcleo matemático de estos mecanismos radica en la autoatención (*self-attention)*, definida por la operación:

$$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}\left(\frac{QK^{\mathrm{T}}}{\sqrt{d_k}}\right)V,$$

donde Q, K y V son proyecciones lineales de las representaciones númericas del texto (*embeddings)* de entrada. En la oración *"El perro y el gato se llevan bien, creo que ellos serán buenos amigos."*, la palabra *"ellos"* genera un vector de consulta (Q) que interactúa con los vectores clave (K) de todas las palabras

anteriores. El producto escalar $QK^{\top}$, modulado por el factor$\sqrt{d_k}$, cuantifica la compatibilidad semántica, y la función *Softmax* convierte estos valores en pesos de atención normalizados [79]. Finalmente, los vectores de valor (V) de *"perro"* y *"gato"* se combinan proporcionalmente a estos pesos, generando una representación contextual precisa para *"ellos"*.

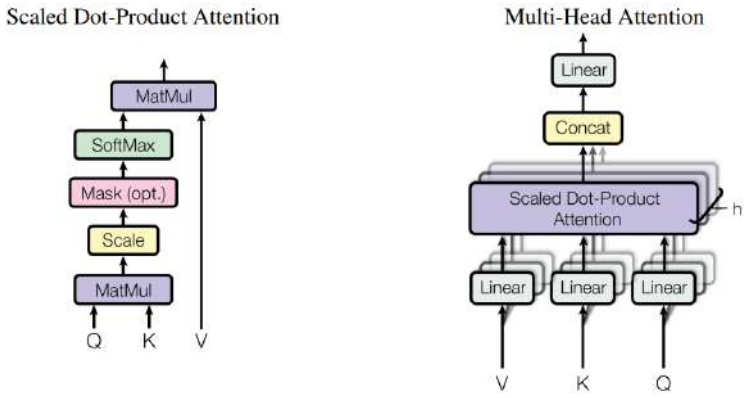


Figura 16. Relación entre self-attention y multi-head attention [79].

La Figura 16 ilustra gráficamente la interrelación entre el mecanismo de *autoatención* y la atención multi-cabeza (*multi-head attention)*. En esta representación, cada cabeza de atención ejecuta un cálculo independiente de *autoatención*, generando representaciones especializadas. Posteriormente, los resultados de estas cabezas se concatenan y se proyectan mediante una capa lineal, integrando las perspectivas diversas en un espacio de representación unificado. Este proceso potencia la capacidad del modelo para capturar relaciones contextuales multifacéticas, desde patrones gramaticales hasta conexiones semánticas de largo alcance, tal como se describe en el marco teórico.

La aplicación más conocida de los *Transformers* son los grandes modelos de lenguaje (*LLM*) como *Claude 3.5*, *GPT-4* o *DeepSeek-V3* [81]. Estos *LLM* tienen una gran popularidad donde, a través de *chats,* los usuarios introducen las diversas entradas textuales y son capaces de generar textos comprensibles a una velocidad aceptable. Estas tecnologías se han convertido en esenciales hoy en día en el área de la inteligencia artificial. No obstante, su uso no se limita a la generación de texto. En los últimos años se han extendido a otras aplicaciones como la visión por computador o la generación de música [82].

El modelo *MuseNet* se basa en una *GAN* en la cual el generador incorpora tanto una *RNN* como el *Transformer* GPT-2. El discriminador, por su parte, está compuesto por una red neuronal multicapa de cinco niveles (64-32-16-8-8). El objetivo principal de este modelo es generar secuencias musicales afines en respuesta a entradas de audio, las cuales provienen del *Lakh MIDI Dataset*, un conjunto de datos que incluye 174.154 archivos MIDI de *piano rolls* [83]. Los datos obtenidos revelan que la implementación de GPT-2 en el sistema generador deriva en secuencias de audio con mayor estructuración interna frente a las producidas por modelos basados en *RNN*. Este hallazgo refuerza la hipótesis de que la arquitectura *Transformer*, específicamente en su variante GPT-2, desempeña un rol clave en la optimización de dos aspectos críticos: la coherencia temporal durante el desarrollo de las piezas y la consistencia melódica entre sus componentes.

3.5 Redes Neuronales Convolucionales

Las redes neuronales convolucionales (*CNN*), propuestas inicialmente por Yann LeCun en su trabajo *Convolutional Networks for Images, Speech and Time Series*, surgen como una solución innovadora para la clasificación de dígitos manuscritos en el conjunto MNIST [84]. Este problema, aparentemente simple, implica desafíos significativos debido a las diferencias en la forma y la disposición de los trazos al escribir a mano. LeCun introdujo la operación de convolución como mecanismo central, inspirándose en la organización jerárquica del sistema visual biológico, lo que permite a las *CNN* extraer características locales y progresivamente abstractas [85]. En la Figura 17 se puede observar el proceso que realiza la red neuronal convolucional para la tarea de clasificación de dígitos manuscritos.

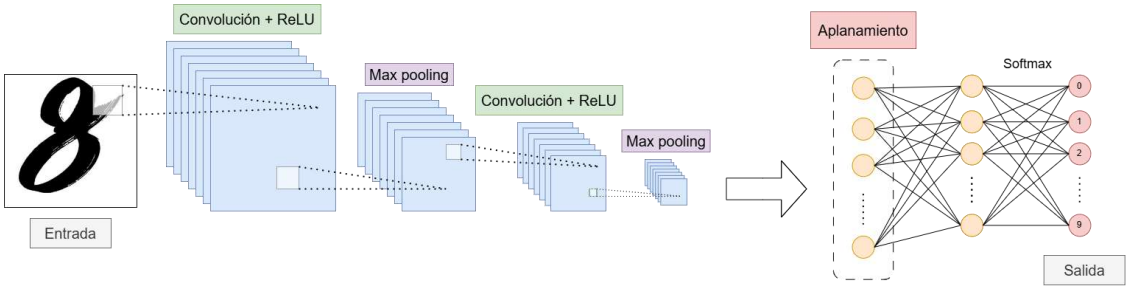


Figura 17. Arquitectura de una red neuronal convolucional.

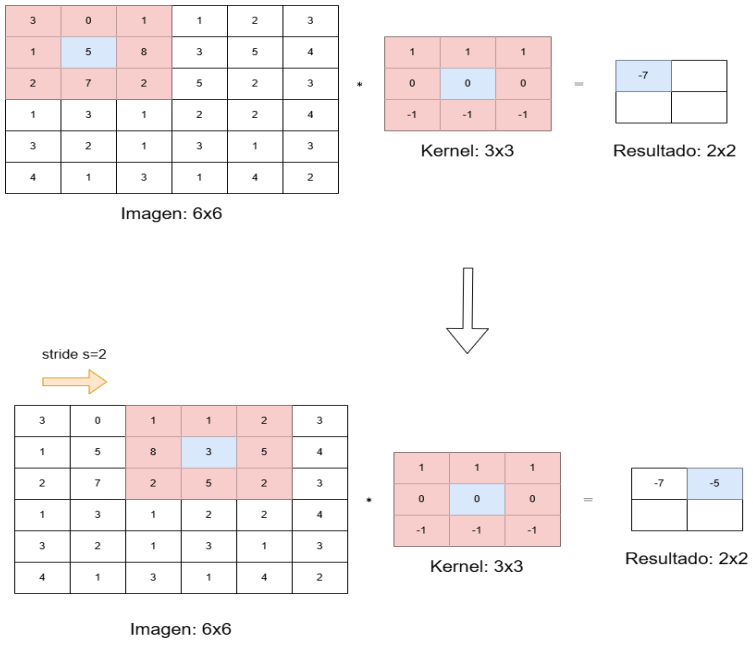
La operación de convolución en *CNNs* se define matemáticamente para señales discretas en dos dimensiones (como imágenes). Dada una entrada I (matriz de $\mathrm{W} \times \mathrm{H}$) y un kernel (matriz cuadrada de dimensión de k), la convolución se calcula como:

$$(I * K)(i,j) = \sum_{m=0}^{k-1} \sum_{n=0}^{k-1} I(i+m, j+n) \cdot K(m,n) + \mathrm{b},$$

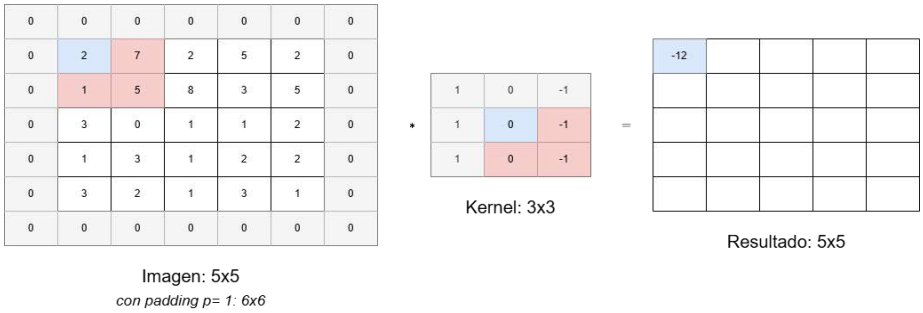
donde b es un término de sesgo (*bias*), m es el desplazamiento vertical dentro del *kernel* (filas) y n es el desplazamiento horizontal dentro del *kernel* (columnas) [86]. El *kernel*, que actúa como filtro para extraer características, se desplaza sistemáticamente sobre la entrada I : horizontalmente en intervalos de s posiciones (*stride*) y, al llegar al límite derecho, desciende s posiciones verticalmente para reiniciar el recorrido horizontal. Por ejemplo, con $\mathrm{s} = 2$, el kernel avanza dos posiciones en cada paso horizontal y dos al cambiar de fila, reduciendo la superposición entre regiones y el tamaño de salida. Un *kernel* diseñado para detectar bordes horizontales (Filtro de Prewitt) puede tener valores como:

$$\begin{matrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{matrix}$$

enfatizando las transiciones verticales de intensidad; es decir, los bordes y detalles que aparecen en la imagen en esa dirección [87].



a. Dinámica de la operación de convolución sobre una imagen con stride s = 2



b. Operación de convolución sobre una imagen con *padding* p = 1 y stride s = 1

Figura 18. Comparación del resultado de aplicar la operación de convolución sobre una imagen con y sin padding.

La Figura 18a muestra un kernel 3×3 aplicado sobre una imagen 6×6 con *stride* s = 2. En cada paso, el kernel se desplaza y solo se aplica en posiciones donde encaja completamente dentro de la imagen de entrada. Además, el centro de la matriz del filtro debe ubicarse siempre en la coordenada del punto de la imagen donde se aplica la convolución. Como resultado, la convolución reduce inherentemente el tamaño de la imagen procesada. Para preservar las dimensiones se utiliza el *padding*, extendiendo I con un margen de ceros de ancho p. La dimensión de salida O se calcula como:

$$O = \left\lfloor \frac{W+2p-k}{s} \right\rfloor + 1.$$

Si $p = \frac{k-1}{2}$ (*padding* "válido"), la salida mantiene las dimensiones de I. Por ejemplo, en la Figura 18b se aprecia una imagen 5×5 con kernel 3×3, *stride* $s = 1$ y *padding* $p = 1$, la cual produce una salida 5×5. Esta estrategia es fundamental en arquitecturas de redes neuronales profundas, donde la

aplicación secuencial de capas convolucionales sin *padding* provocaría una reducción progresiva de la resolución espacial. La reducción degrada los mapas de características, que son representaciones intermedias de la entrada generadas en cada capa para capturar patrones como bordes, texturas o formas; comprometiendo así su utilidad en etapas posteriores de la red [88].

Tras la convolución, el resultado se combina con una función de activación no lineal, típicamente *ReLU*. Asimismo, también es habitual aplicar una capa de *pooling* después de la anterior con el objetivo de resumir las regiones de los mapas de características. Una de las más utilizadas es *max pooling*. El *max pooling* consiste en retener el valor máximo en ventanas cuadradas de dimensión k_p [89]. Por ejemplo, con $k_p = 2$ y $stride = 2$, una entrada 4×4 se reduce a 2×2. Esta reducción a la mitad ocurre porque el kernel divide la entrada en bloques no superpuestos, descartando datos pero preservando rasgos dominantes (véase Fig. 17). Además, introduce invarianza a pequeñas traslaciones, útil en tareas como reconocimiento de objetos en distintas posiciones.

En la capa final, los mapas de características se aplanan y se proyectan en un espacio de clases mediante una capa densa. La función *Softmax* normaliza estos *logits* en probabilidades:

$$P(y = i) = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}},$$

donde $P(y = i)$ es la probabilidad de que la entrada pertenezca a la clase i, z_i y z_j representan los *logits* (salidas no normalizadas de la capa densa); C es el número total de clases y el denominador $\sum_{j=1}^{C} e^{z_j}$ asegura que las probabilidades sumen 1, normalizando los valores. En MNIST, esto permite asignar, por ejemplo, un 90% de probabilidad al dígito "8" si los bordes curvos y se detecta que se cruzan en la mitad formando dos círculos.

3.5.1 WaveNet

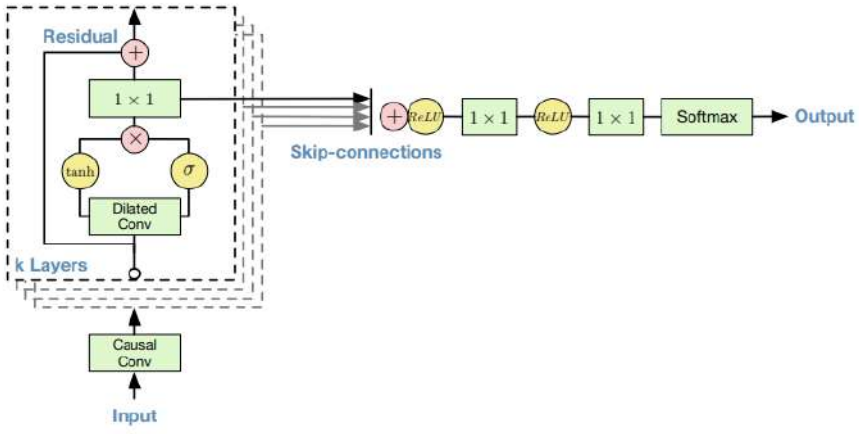


Figura 19. Arquitectura de WaveNet [90].

El artículo *WaveNet: A Generative Model for Raw Audio*, desarrollado por Aäron van den Oord y su equipo [90], propone una arquitectura innovadora para la síntesis de audio mediante redes neuronales profundas. Este modelo, de naturaleza autorregresiva, predice cada muestra de audio sucesiva basándose

exclusivamente en las muestras anteriores, lo que lo hace ideal para manejar secuencias temporales como el habla o la música. Además de la generación de audio en bruto, los autores destacan aplicaciones prácticas como la conversión de texto a voz (*text-to*-speech, *TTS*) y el reconocimiento automático de habla, áreas donde el modelo demuestra avances significativos [90].

La estructura de *WaveNet*, ilustrada en la Figura 19, se fundamenta en capas convolucionales con dos características distintivas: la convolución causal y la convolución dilatada causal. La primera garantiza que las predicciones en un momento dado no dependan de información futura, preservando la secuencialidad temporal. Por ejemplo, al generar una nota musical, el modelo solo considera las notas anteriores, evitando "mirar hacia adelante". La segunda, la convolución dilatada, introduce un factor de dilatación que expande el rango temporal procesado en cada capa. Este factor determina el espaciado entre las muestras de entrada consideradas en base 2^n, siendo n el número de capa desde $n = 0$. En la primera capa (dilatación 1), se procesan muestras consecutivas; en la siguiente (dilatación 2), se saltan una muestra intermedia; en la tercera (dilatación 4), se omiten tres, y así sucesivamente. Esta progresión geométrica, como se muestra en la Figura 20, permite al modelo capturar dependencias tanto locales como de largo alcance, similar a cómo un músico anticipa patrones armónicos a lo largo de una pieza. El campo receptivo hace referencia al intervalo temporal que influye en cada predicción y se amplía exponencialmente con cada capa, permitiendo a *WaveNet* integrar un contexto histórico extenso. Por ejemplo, para generar una sílaba en un discurso, el modelo no solo considera los fonemas inmediatos, sino también la entonación y el ritmo acumulados. Esta capacidad resulta crucial para producir audio coherente y natural.

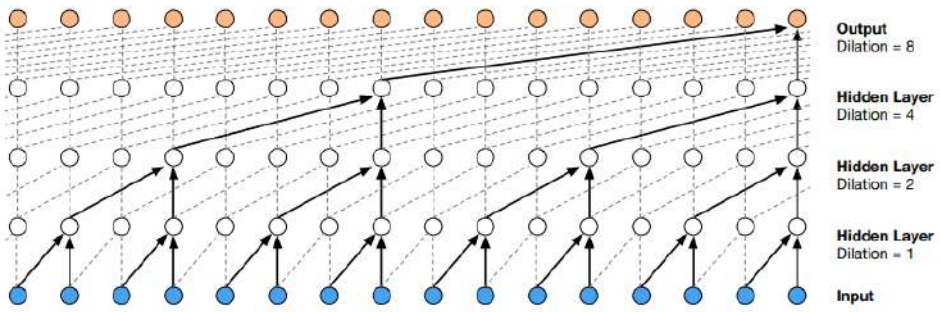


Figura 20. Ejemplo de convolución dilatada causal usada en una capa del modelo WaveNet [90].

La estructura interna del modelo comienza aplicando una convolución causal estándar (equivalente a una dilatada con factor 1) a la entrada de audio. Posteriormente, se apilan múltiples capas ocultas, cada una con un factor de dilatación creciente. En cada capa, se aplica una función de activación dual: una combinación de tangente hiperbólica tanh (para modelar relaciones no lineales) y sigmoide σ (para regular la magnitud de las salidas). Este proceso genera dos tipos de señales: los residuos, que alimentan la siguiente capa, y las conexiones saltadas (*skip connections*), que capturan información intermedia para acelerar la convergencia del entrenamiento. Estas últimas actúan como atajos que evitan la pérdida de detalles críticos en capas profundas, similar a cómo un director de orquesta coordina a los diferentes instrumentos que forman secciones individuales sin perder la visión global.

Tras procesar todas las capas, las conexiones saltadas se suman y se pasan por dos funciones de activación *ReLU*, que introducen no linealidad y estabilidad numérica. Finalmente, la salida se transforma mediante una función *Softmax* para asignar probabilidades a cada posible valor de la siguiente muestra de audio, asegurando que la predicción refleje la distribución real de los datos.

Por último, en cuanto a los resultados obtenidos por el modelo, los experimentos demuestran que *WaveNet* produce música con estructura armónica y calidad acústica notable, especialmente cuando se limita a un solo instrumento, como el piano. Sin embargo, su rendimiento decae ante entornos complejos, como composiciones con múltiples géneros o instrumentos. Aunque el campo receptivo amplio ayuda a capturar patrones, la falta de restricciones explícitas sobre la estructura musical, como una progresión de acordes o un esquema rítmico, dificulta mantener coherencia en piezas largas [91]. Esto sugiere que, si bien el modelo es poderoso para modelar secuencias locales, requiere mecanismos adicionales para integrar conocimiento de alto nivel, como reglas compositivas más complejas o jerarquías temporales que modelen la polifonía entre distintos instrumentos [92].

4 Planteamiento del problema

El objetivo de este trabajo de fin de grado consiste en implementar y comparar diferentes modelos neuronales para generar, de forma sintética y a partir de fragmentos de audio, melodías originales de piano formadas por secuencias de 200 notas musicales. Se espera que las composiciones generadas sean coherentes, melódicamente estructuradas y agradables al oído.

Las arquitecturas neuronales a estudiar se agrupan de acuerdo con dos enfoques o paradigmas. El primero son las redes neuronales recurrentes, específicamente *LSTM, GRU* y una combinación de ambas. Para analizar su impacto en la generación musical, se exploran distintas configuraciones de ambas arquitecturas. Se evalúa el rendimiento de *LSTM* y *GRU* de forma independiente, probando tanto con una única capa de celdas como con múltiples capas de celdas de memoria en cada modelo. Además, se estudia la posibilidad de combinar ambas arquitecturas dentro de una misma red neuronal para determinar si su integración mejora los resultados. El segundo grupo se enfoca en el modelo autorregresivo de *WaveNet*, una red neuronal de tipo convolucional que emplea convoluciones dilatadas para capturar dependencias de largo alcance en secuencias temporales. Esta arquitectura, originalmente diseñada para síntesis de voz, resulta adecuada para la generación musical debido a su capacidad para modelar relaciones jerárquicas en las secuencias de audio.

Para el entrenamiento de los modelos neuronales a estudiar se emplea en conjunto de datos *Maestro Dataset* de Google Magenta, un repositorio público con 200 horas de grabaciones en formato MIDI de piezas clásicas para piano con un total de 1276 audios. Este conjunto de datos incluye, asimismo, metadatos como el compositor, la obra y detalles técnicos de cada interpretación. En el siguiente enlace se puede acceder al conjunto de datos *Maestro*:

https://magenta.tensorflow.org/datasets/maestro

Para evaluar los resultados de los modelos neuronales implementados de forma objetiva, se prioriza la función de pérdida (*loss*). La pérdida cuantifica cómo de bien el modelo predice la siguiente nota en una secuencia, comparando sus estimaciones con las notas reales del conjunto de datos de entrenamiento. No obstante, al generar audio, se consideran métricas adicionales para evaluar su estructura y calidad. Una de ellas es la polifonía, que mide la frecuencia con la que suenan dos o más notas simultáneamente, incluso con ligeros desfases temporales. También se analiza la consistencia escalar, que representa el porcentaje de notas alineadas con escalas estándar (mayores y menores), permitiendo identificar la escala más representativa en cada pieza. Asimismo, se toma en cuenta el rango tonal, que indica la distancia en semitonos entre la nota más grave y la más aguda, lo que resulta útil para detectar melodías monótonas o excesivamente caóticas.

La evaluación subjetiva se basa en una escucha activa en la que se analizan diversos aspectos de la composición generada. Entre las cualidades evaluadas se incluyen la fluidez y coherencia melódica, el ritmo, la armonía entre las notas y la progresión musical. Además, se considera el grado en que la melodía resulta agradable, asegurando que las piezas generadas también sean estéticamente satisfactorias para el oyente.

Así mismo, las melodías generadas por los modelos implementados se comparan con otros ya existentes como *MuseNet, C-RNN-GAN* y *GANSynth*. MuseNet, basado en transformadores, tiene la capacidad de generar piezas polifónicas mediante un mecanismo de atención a largo plazo. Por su parte, *C-RNN-GAN* combina redes recurrentes (*RNN*) con redes generativas adversarias (*GAN*) para imitar estilos clásicos. *GANSynth* emplea *GAN*s para sintetizar audio directamente en el dominio frecuencial, priorizando la calidad sonora.

El último objetivo de este trabajo es analizar el impacto sociocultural de la generación automática de música. Se considera su potencial para democratizar la creación artística, los desafíos éticos relacionados con la autoría de obras generadas por IA y su posible influencia en la evolución de los géneros musicales a largo plazo. También se abordan implicaciones prácticas en sectores como el entretenimiento y la educación, donde esta tecnología podría reducir barreras técnicas para compositores emergentes.

5 Solución propuesta y resultados

En este trabajo se han implementado seis modelos neuronales para la generación de composiciones musicales a piano sintéticas, basados en dos enfoques arquitectónicos principales: redes neuronales recurrentes (*RNN*), que engloba a las arquitecturas *LSTM* y *GRU*, y el modelo *WaveNet*. Cada tipo de modelo se ha diseñado y ajustado de manera individual para optimizar su rendimiento en dicha tarea, lo que ha requerido configuraciones específicas en su arquitectura, calibración de hiperparámetros y una preparación de datos diferenciada. Para su identificación, se establece la siguiente nomenclatura técnica:

- Modelo *LSTM*: integra una única capa de celdas *LSTM*.
- Modelo *GRU*: incorpora una única capa de celdas *GRU*.
- Modelo multi-*LSTM*: consta de dos capas secuenciales de celdas *LSTM*.
- Modelo multi-*GRU*: emplea dos capas secuenciales de celdas *GRU*.
- Modelo *LSTM-GRU*: combina dos capas de celdas *LSTM* seguidas de dos capas de unidades *GRU*, dispuestas en secuencia.
- Modelo *Wavenet*: basado en capas de convoluciones dilatadas, diseñado para capturar dependencias temporales a largo plazo.

El lenguaje de programación escogido para la implementación es *Python*, utilizando las bibliotecas *Keras* y *TensorFlow*. Así mismo, se ha empleado una tarjeta gráfica NVIDIA GTX 1050.

El presente apartado detalla la metodología seguida en cada una de las seis arquitecturas implementadas. En primer lugar, se especifica el preprocesado en común realizado sobre el conjunto de datos seleccionado para todos los modelos neuronales. Así mismo, para cada uno de ellos, se describe el preprocesado concreto realizado. Acto seguido, se detalla la configuración de hiperparámetros del modelo neuronal y la implementación de su estructura. Posteriormente, se incluyen los resultados obtenidos en cada uno de los experimentos, seleccionando el audio más representativo del conjunto total de audios generados sintéticamente, junto a un enlace para escuchar dicho fragmento de audio y acompañado de la partitura compuesta a partir de la melodía creada. Los modelos neuronales implementados generan predicciones secuenciales para construir una nueva melodía, de tal manera que, en cada predicción, la primera nota de la secuencia se descarta y la nueva nota generada se añade en la última posición. La Figura 21 ilustra este proceso.

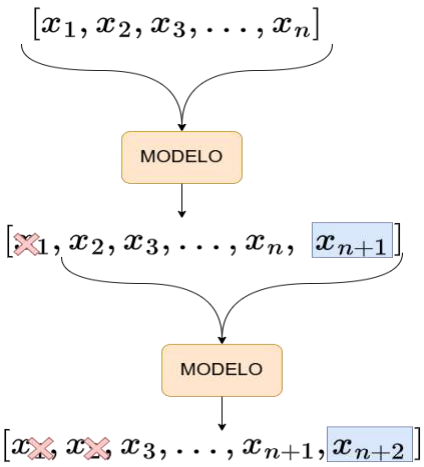


Figura 21. Ejemplo de generación de una melodía sintética.

Para cada modelo neuronal implementado, se realiza una valoración cualitativa de su capacidad generativa a partir de las piezas de audio a piano generadas sintéticamente, que se complementa con evaluaciones objetivas . Por último, se realiza una comparativa con modelos generativos de referencia actuales como *MuseNet*, *C-RNN-GAN* y *GANsynth* con el fin de contextualizar las contribuciones de este estudio en el ámbito de la generación musical automatizada.

5.1 Preprocesado del conjunto de datos

El conjunto de datos seleccionado para el entrenamiento de las redes neuronales es *Maestro Dataset* de Google Magenta, el cual se puede consultar en el siguiente enlace:

maestro_dataset

Este conjunto contiene 1276 audios, que se han distribuido en los siguientes subconjuntos: 75 % para entrenamiento (962 audios), 11 % para validación (137 audios) y 14 % para las pruebas finales (177 audios). A pesar de disponer de un gran número de audios en formato MIDI, las limitaciones en los recursos computacionales exigieron trabajar con un conjunto de datos reducido: 20 audios para entrenamiento, 10 para validación y 10 reservados como conjunto de pruebas finales. Para la generación de melodías sintéticas, se produjeron inicialmente 15 audios, de los cuales se seleccionó el más representativo de este último grupo, garantizando que el modelo no se hubiera expuesto previamente a dichas entradas durante su entrenamiento.

En cuanto a la extracción de las notas musicales de los archivos MIDI, ha sido necesario implementar una función que devuelve el tono de cada una de las notas y determina si se trata de un acorde formado por varias notas. De esta manera, se obtienen 111542 notas musicales que conforman el conjunto de entrenamiento; mientras que 46792 notas forman el conjunto de validación, al igual que el de pruebas finales. Se ha implementado también una función para reproducir el audio en formato MIDI. Su interfaz se puede observar en la Figura 22 junto con la llamada a la función *play_audio*, que toma como parámetro un archivo MIDI, en este caso, *pm*. La tasa de muestreo utilizada para reproducir los audios generados es de 16000 Hz.

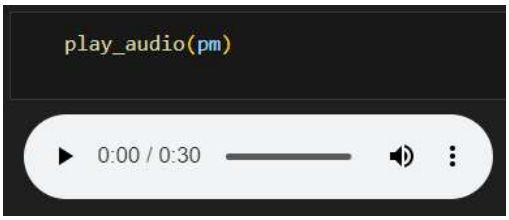


Figura 22. Reproductor de audio en formato MIDI.

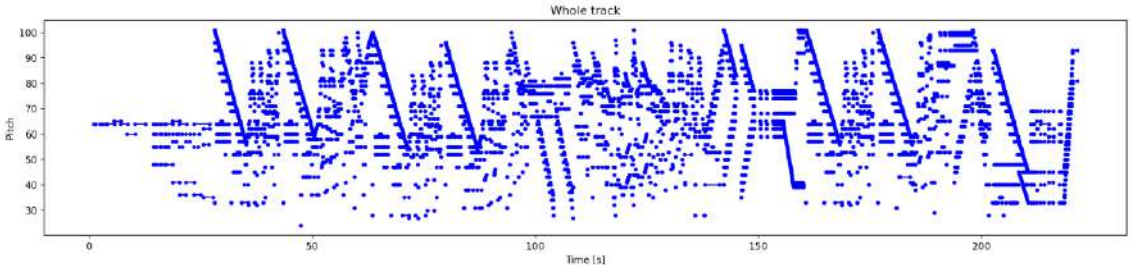
5.2 Preprocesado de datos para los modelos LSTM y GRU

El primer paso en la implementación de redes neuronales basadas en LSTM y GRU, en todas sus variantes, consiste en un preprocesado específico de los datos para adaptarlos al formato requerido por estos modelos neuronales, asegurando un entrenamiento eficiente para la generación musical. La generación de audio requiere separar de una melodía su tono (*pitch*), su duración (*duration*) y el intervalo entre notas consecutivas (*step*). Por tanto, parte del procesado del archivo MIDI consiste en extraer las notas con sus tres características, que se almacenan en una tabla (*dataframe*) junto al momento en que cada nota empieza y deja de tocarse. La Figura 23 muestra esta tabla para las cinco primeras notas extraídas de un audio.

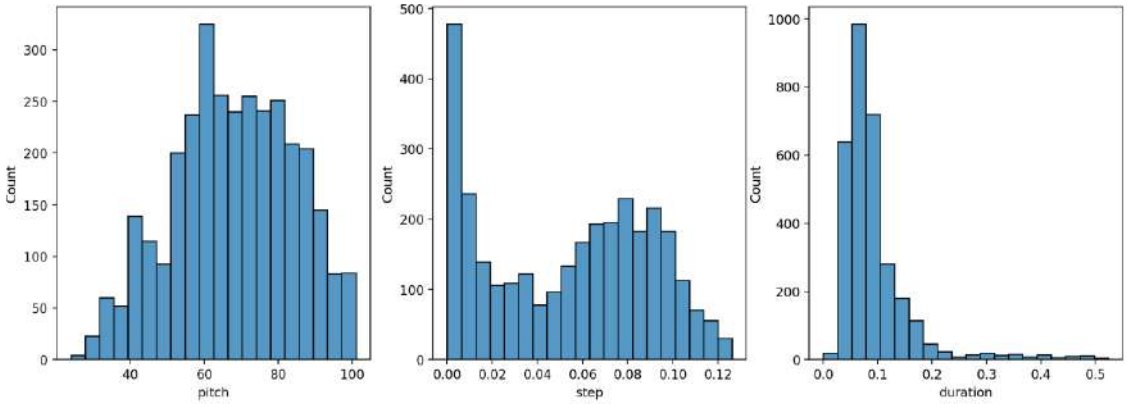
	pitch	start	end	step	duration
0	64	0.983073	1.451823	0.000000	0.468750
1	64	2.488281	2.933594	1.505208	0.445312
2	64	3.789062	3.944010	1.300781	0.154948
3	64	4.065104	5.695312	0.276042	1.630208
4	65	5.553385	7.134115	1.488281	1.580729

Figura 23. Tabla para almacenar el tono, intervalo (step) y duración de cada una de las notas en los audios.

Esta información es útil para visualizar un *piano roll* y tener más claro la distribución de las notas en los audios usados en el entrenamiento. La Figura 24 muestra un *piano roll* de un ejemplo de audio (a.) junto a la distribución del tono, duración e intervalos de las notas que lo conforman (b.).



c. *Piano roll* de un audio completo del conjunto de datos.



d. Distribución del tono, duración e intervalos de un audio completo del conjunto de datos.

Figura 24. Piano roll y distribución del tono, duración e intervalos de un audio completo del conjunto de datos.

En el siguiente enlace se puede escuchar un fragmento de un ejemplo de audio perteneciente al conjunto de datos:

example_maestro_dataset

Los modelos neuronales basados en LSTM y GRU se entrenan a partir de secuencias de notas musicales. Se establece un tamaño de entrada de 63 notas musicales, siendo la siguiente usada como salida deseada durante el entrenamiento. El modelo debe aprender a predecir la siguiente nota en la secuencia. La mayoría de las melodías clásicas a piano tienen un compás de 4/4, por lo que establecer una dimensión de 63 notas de entrada ayuda al modelo a aprender mejor el patrón rítmico. Se sigue la misma metodología para el conjunto de validación. Puesto que la entrada al modelo consiste en 63 notas musicales, compuestas cada una de ellas por su tono, duración e intervalo, se tiene que la dimensión de cada entrada a la red de neuronas es de (63,3). Cabe destacar que los modelos neuronales LSTM y GRU trabajan con secuencias, por lo que la entrada de estos modelos es una secuencia de tamaño (63,3). Como se quiere obtener un archivo MIDI en la salida del modelo neuronal para poder reproducirlo posteriormente, se usa el valor 128 como tamaño del vocabulario para escalar los tonos de las notas de los audios en ese rango, es decir, hay 128 posibles tonos de salida para una nota. De esta manera, la conversión a formato MIDI se puede realizar más fácilmente y cada una de los 63 notas tiene codificado el valor del tono como un número entre el 0 y el 1. La duración y el intervalo permanecen inmutables sin ninguna conversión.

5.3 Implementación del modelo LSTM

El modelo neuronal basado en LSTM recibe como entrada una secuencia de 63 notas, donde cada una incluye información acerca del tono, duración e intervalo. El tono viene normalizado según el formato MIDI de 128 valores entre 0 y 1. Por tanto, la entrada del modelo tiene dimensión (63,3). Acto seguido, la secuencia pasa a la única capa intermedia formada por 128 celdas *LSTM*. El resultado es un vector de dimensión 128 que se conecta a una capa densa de salida formada por 128 neuronas, que representan los 128 valores posibles del número de nota MIDI (0 a 127) que define el tono musical. La salida neta de esta capa son los *logits*, valores que aún no han sido procesados por la función de activación. El

modelo realiza la predicción del tono utilizando la función *Softmax*, que transforma estos *logits* en un vector de probabilidades. Esta salida se compara con una codificación *one-hot* del tono real, que es un número entero que se corresponde con el rango de nota MIDI (0 a 127). De esta manera, cada neurona corresponde a una nota específica dentro de ese rango, y la activación de una neurona (mediante *Softmax*) indica la probabilidad de que esa nota sea la correcta. Esta codificación permite generar directamente un archivo MIDI compatible, ya que el formato utiliza este mismo rango numérico para definir las notas. En cuanto a la predicción de la duración y el intervalo, la salida de la capa intermedia de 128 celdas LSTM se conecta también con dos neuronas, que devuelve el resultado como un valor real. Las tres salidas: valor de tono, duración e intervalo definen la nota musical generada como salida del modelo. La Figura 25 muestra la estructura neuronal del modelo.

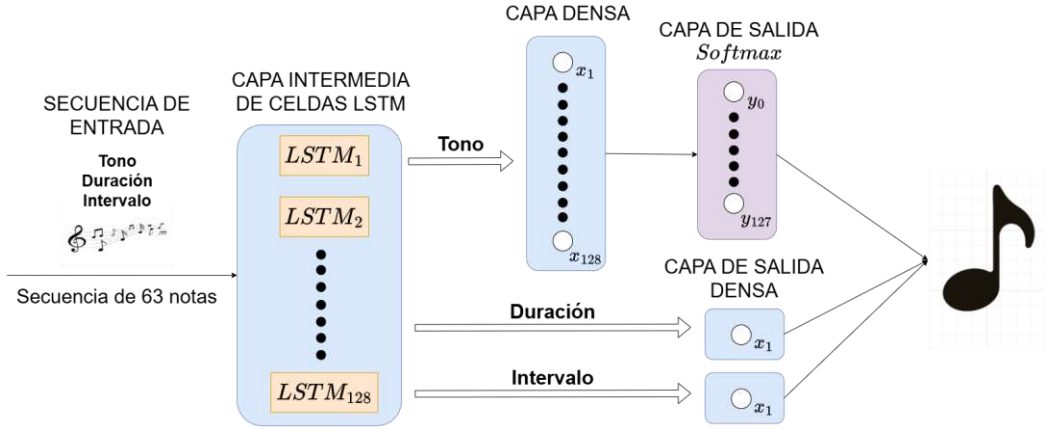


Figura 25. Estructura neuronal del modelo LSTM.

Se utiliza una función de error diferente para el tono y otra tanto para la duración como para el intervalo. En el caso del tono, se trata de un problema de clasificación, por lo que se utiliza la función de pérdida (*loss*) *sparse categorical crossentropy*, cuya fórmula es la siguiente:

$$L_{CE} = -\sum_{i=1}^{N} t_i \log(p_i), \tag{5.1}$$

que compara las diferencias entre la etiqueta del tono real (t_i), con la probabilidad *Softmax* de la clase *i-ésima* (p_i) para todos los valores de tono N, que en este caso es 128. De esta forma, el resultado indica lo bien que se ha realizado la predicción respecto al valor real del tono, siendo el valor cero, el óptimo.

En caso de las salidas correspondientes a la duración e intervalo, se ha diseñado una función de pérdida específica, basada en el error cuadrático medio (MSE), que incentiva al modelo a no devolver valores negativos, dado que no son posibles para la duración y el intervalo:

$$L_D = \left[\left(\widehat{Y_D} - Y_D\right)^2 + 10 \cdot \max\left(-\widehat{Y_D}, 0\right)\right]. \tag{5.2}$$

$$L_I = \left[\left(\widehat{Y_I} - Y_I\right)^2 + 10 \cdot \max\left(-\widehat{Y_I}, 0\right)\right].$$

Se considera la variable m como m = D, I. El primer término $\left(\widehat{Y_m} - Y_m\right)^2$ es el error cuadrático medio (MSE). El segundo término, $10 \cdot \max\left(-\widehat{Y_m}, 0\right)$, actúa como penalización para predicciones negativas: si $-\widehat{Y_m} < 0$, se suma un término al error proporcional a $10 \cdot |\widehat{y_m}|$. De esta forma, la red neuronal aprende a no

devolver valores negativos para la duración y el intervalo a la vez que se compara con su valor deseado o real.

La función de error total engloba al tono, duración e intervalo, siendo la suma de las dos funciones de pérdida anteriores. Por tanto:

$$\mathcal{L}(\mathrm{W}) = L_{\mathrm{CE}} + L_{\mathrm{D}} + L_{\mathrm{I}}\,. \tag{5.3}$$

En las pruebas realizadas durante el entrenamiento, se observa que la pérdida para el tono (*pitch*) es mucho mayor que la correspondiente para la duración y el intervalo, por lo que el error total del modelo se encuentra dominado por el tono. Para solventarlo, se ha empleado el argumento *loss_weights* que ofrece *Keras* en la compilación del modelo. Este parámetro pondera las pérdidas de las diferentes salidas por su valor. Otorgando un valor de loss weights de 0,05 al tono es posible equilibrar la influencia de los valores de las tres funciones de pérdida en el total. De esta manera, la función de pérdida total ponderada final es la siguiente:

$$\mathcal{L}(W) = 0.05 \cdot L_{\mathrm{CE}} + L_{\mathrm{D}} + L_{\mathrm{I}}\,. \tag{5.4}$$

La Figura 26 muestra el antes y el después en el cómputo de la función de pérdida del modelo.

```
{'loss': 5.596830368041992,
 'duration_loss': 0.1844051033258438,
 'pitch_loss': 4.8476176261901855,
 'step_loss': 0.5648069977760315}
```

a. *Pérdida total del modelo sin la suma ponderada de las pérdidas individuales.*

```
{'loss': 0.9915928244590759,
 'duration_loss': 0.1844051033258438,
 'pitch_loss': 4.8476176261901855,
 'step_loss': 0.5648069977760315}
```

b. *Pérdida total del modelo con la suma ponderada de las pérdidas individuales.*

Figura 26. Comparación de la función de pérdida total sin y con suma ponderada de las pérdidas individuales.

En cuanto a los hiperparámetros utilizados en este modelo, se establece un tamaño de lote (*mini-batch*) de 1024 para el entrenamiento del modelo, después de varias pruebas que mostraron que este valor ofrecía los mejores resultados. La tasa de aprendizaje se establece en 0,001. Para mitigar el riesgo de sobreajuste, se aplica la técnica de regularización *dropout* con un valor de 0,2 en la capa intermedia de celdas LSTM y también se incorpora *early stopping* como método de detención temprana. El optimizador utilizado es *RMSprop*.

El proceso de entrenamiento de la red neuronal se compone de 300 épocas en las que el modelo aprende las características musicales de la secuencia de entrada con el objetivo de predecir y generar la siguiente nota musical.

El último paso consiste en generar una secuencia de 200 notas utilizando el modelo. Cabe destacar que, en cada predicción, la nota situada en la primera posición de la secuencia se elimina y se añade, en la última posición, la nueva nota generada como se muestra en la Figura 21. Como paso adicional en la generación de notas, se emplea un parámetro *temperatura (temperature)* que proporciona *Keras* para regular la variedad en la predicción o generación del tono. Se aplica dividiendo los *logits* del vector de salida correspondiente al tono (antes de aplicar *Softmax*) por dicho valor. Un valor bajo concentra la probabilidad del vector de salida para el tono en los valores más predecibles. Un valor alto suaviza la distribución, permitiendo variaciones más creativas. Se ha seleccionado un valor alto de este hiperparámetro (*temperature* = 5) para aportar mayor variedad a la composición musical, lo que permite enriquecer la creatividad del resultado.

5.3.1 Resultados del modelo LSTM

Las notas generadas sintéticamente (tono, duración e intervalo) se almacenan en una tabla (*dataframe)* para su posterior análisis. La Figura 27 muestra un ejemplo de una tabla compuesta por 10 notas.

	pitch	step	duration	start	end
0	103	0.135987	0.061905	0.135987	0.197892
1	94	0.000000	0.000000	0.135987	0.135987
2	96	0.096955	0.000000	0.232942	0.232942
3	99	0.161293	0.000000	0.394235	0.394235
4	83	0.157780	0.000000	0.552015	0.552015
5	85	0.179745	0.000000	0.731759	0.731759
6	97	0.242953	0.000000	0.974712	0.974712
7	118	0.243007	0.000000	1.217719	1.217719
8	72	0.243668	0.000000	1.461387	1.461387
9	96	0.297314	0.000000	1.758701	1.758701

Figura 27. Tabla compuesta por 10 notas generadas sintéticamente mediante el modelo LSTM.

Como se puede observar, la columna *duration* tiene la mayoría de sus valores a 0. Esto se debe a que la figura musical que representa la duración de la nota generada es tan pequeña que no se capta fácilmente, pudiendo tratarse de la figura musical fusa, la treintaidosava parte.

El audio completo generado se puede escuchar en el siguiente enlace:

music_lstm_raw

El archivo de audio generado contiene notas distribuidas en octavas extremas (demasiado graves o agudas), lo que dificulta su identificación auditiva. Para convertirlo en una partitura, se redondean las duraciones irregulares a los valores de las figuras musicales convencionales. Si una nota se registra con duración cero, se le asigna automáticamente una semicorchea (1/8). Se realiza también un ajuste en las alturas de las notas generadas modificando el tono, que puede tomar valores entre 0 y 127. Para ello, el límite superior se fija en el

valor MIDI 108, bajando el tono de cualquier nota que lo supere hasta este valor. El límite inferior se establece en MIDI 21, elevando las notas inferiores a este mínimo. Esto asegura que todas las notas caigan dentro del rango de un piano estándar compuesto por 88 teclas (La_0 a Do_8). La principal motivación es permitir la visualización del audio generado en una partitura para piano. Sin embargo, se emplean 128 neuronas en la capa de salida para cubrir todo el formato MIDI sin restricciones, asegurando así una representación más flexible y completa. De esta manera, resulta más fácil ajustar posteriormente el rango de las notas al del piano estándar, permitiendo una adaptación más precisa a este instrumento. Como último paso, se desplazan las notas a una octava central (Do_4-Si_4) para evitar sonidos que dificulten la escucha del audio generado sintéticamente por el modelo neuronal. Posteriormente, se genera un nuevo archivo MIDI como el siguiente, generado a partir del anterior (*music_lstm_raw*):

music_lstm_tuned

Finalmente, se genera y visualiza la partitura del audio generado y ajustado para comprender mejor las notas generadas. La Figura 28 muestra la partitura del audio anterior *music_lstm_tuned*. Se puede observar el cambio notorio en las figuras musicales utilizadas al principio y en la segunda mitad del audio generado. En el inicio, son figuras musicales con menor duración (fusa), mientras que en el resto de la partitura domina el valor de corchea junto a la semicorchea.

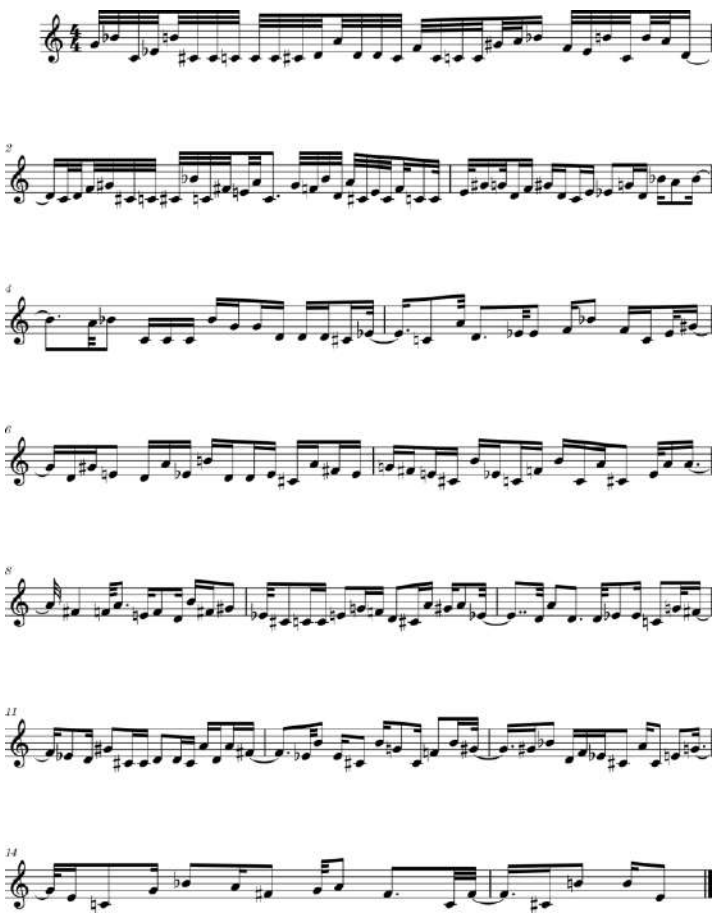
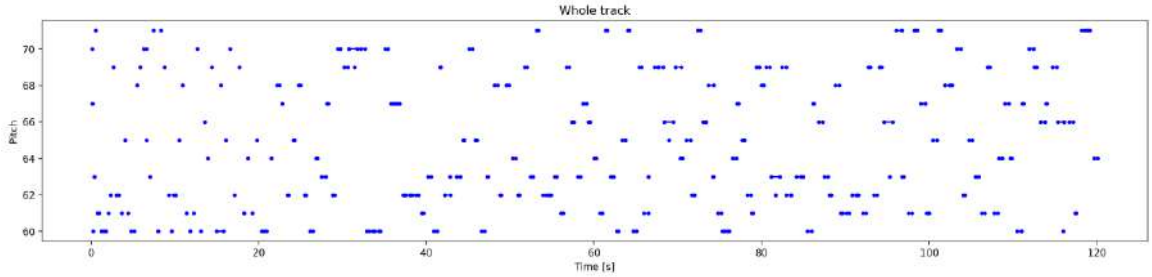
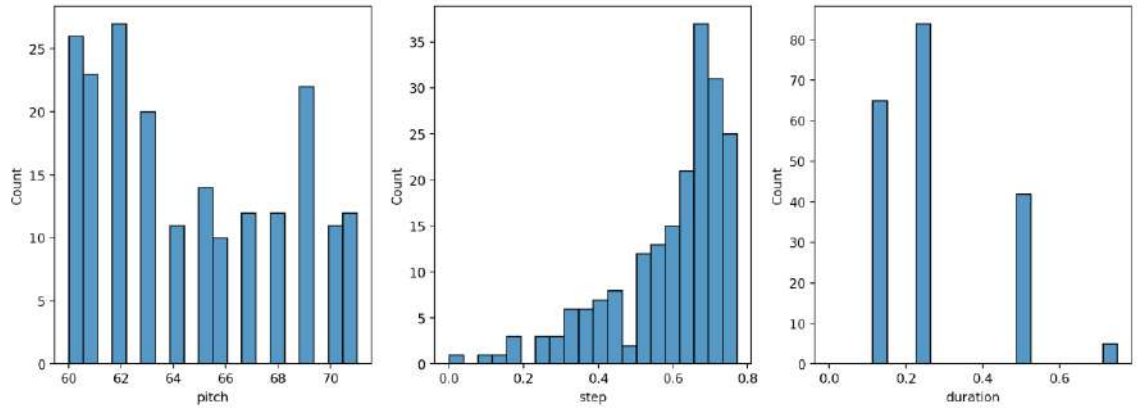


Figura 28. Partitura del audio generado mediante el modelo LSTM y posteriormente ajustado

Con el objetivo de visualizar de manera más efectiva la disposición de las notas generadas y cómo varían las variables de tono, duración e intervalo se muestra en la Figura 29 un *piano roll* (a.) del audio ajustado (*music_lstm_tuned*) junto con una gráfica de la distribución de los valores de cada una de estas tres variables (b.). Se puede observar que la distancia entre notas consecutivas (*step*) suele ser alta y que la duración media de las notas generadas se sitúa en 0.3, sugiriendo que es una pieza musical con un ritmo rápido.



a. *Piano roll del audio generado.*



b. *Distribución de las variables tono, duración e intervalo de las notas generadas.*

Figura 29. Piano roll y gráficas con la distribución de las variables tono, duración e intervalo de las notas generadas mediante el modelo LSTM.

A continuación, se estudian los resultados obtenidos con el modelo *LSTM* desde tres puntos de vista.

En primer lugar, se examina el proceso de entrenamiento mediante la evolución de la función de pérdida, reflejando la capacidad del modelo para aprender patrones musicales. La Figura 30 muestra la evolución durante 300 épocas de la pérdida total para el conjunto de entrenamiento, calculada como la suma ponderada de las pérdidas individuales del *sparse categorical crossentropy* para el tono y la función de pérdida personalizada basada en el MSE tanto para la duración como para el intervalo (5.4).

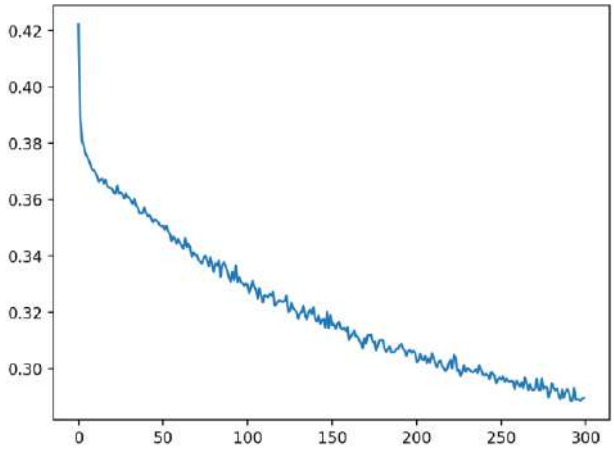


Figura 30. Gráfica de la evolución durante las 300 épocas de la función de pérdida total del conjunto de entrenamiento en el modelo LSTM.

Como puede apreciarse, el entrenamiento se detiene en la época 300, con una pérdida en el conjunto de entrenamiento de aproximadamente 0,30. A pesar de la tendencia decreciente en dicha pérdida, esto puede ser indicativo de sobreajuste, lo cual afecta negativamente la capacidad de generalización del modelo.

En segundo lugar, se evalúan cinco métricas cuantitativas que miden la calidad del audio generado y ajustado (*music_lstm_tuned*), seleccionado como el más representativo de un total de 15 audios generados sintéticamente por su equilibrio entre creatividad y respeto a los patrones entrenados. Estas métricas son: polifonía, escala que mejor se adecua, consistencia, subsecuencias musicales únicas y rango tonal. La Figura 31 muestra los resultados obtenidos.

```
Polyphony Score: 1.18 %
Best Matching Scale: Abstract major
Scale Consistency: 18.31 %
Repetitions of Short Subsequences: 29
Tone Span: 11 semitones
```

Figura 31. Resultados de las métricas de evaluación objetiva sobre el audio generado más representativo mediante el modelo LSTM.

La primera métrica indica la puntuación en polifonía que alcanza esta pieza musical generada. El porcentaje bajo de 1,18 % sugiere que durante el audio no hay apenas acordes o, al menos, dos notas simultáneas que suenen a la vez. La segunda métrica indica el tipo de escala que mejor se adecua al audio. El resultado de *abstract major* indica que es una pieza musical con matices dinámicos que evoca sentimientos alegres. La siguiente métrica indica el porcentaje del audio generado que se ajusta perfectamente a la escala anterior. El valor obtenido de 18.31 % sugiere que, aunque algunos pasajes respetan su estructura, la mayoría no lo hace. La cuarta métrica revela las diferentes subsecuencias o frases musicales únicas que tiene el audio generado. Por último, el rango tonal o la diferencia entre la nota más grave y aguda de la pieza musical creada es de 11 semitonos. Estas dos últimas métricas dan una idea de si el audio generado tiende a ser monótono o si por el contrario es más caótico. El resultado obtenido indica que se trata de una pieza poco repetitiva. Aunque este análisis se centra en uno (el más representativo por su equilibro entre creatividad y respeto a los patrones entrenados)de los 15 audios generados

sintéticamente por el modelo LSTM, el resto ofrece valores similares de polifonía y repeticiones. En general, todos los audios generados a partir conjunto de pruebas finales reflejan una coherencia estructural comparable, aunque con fluctuaciones en la complejidad armónica y la longitud de las frases musicales.

En tercer lugar, se realiza un análisis y evaluación subjetivos basados en la crítica musical a partir de la escucha activa del audio representativo generado sintéticamente y ajustado (*music_lstm_tuned*), integrando perspectivas técnicas y artísticas. En este caso, se puede apreciar que el principio de la pieza musical tiene un ritmo rápido, produciendo un sentimiento de agitación. Las notas musicales parecen estar en una armonía aceptable, sonando de forma agradable pero no siempre. Esto es debido a que, si bien hay progresiones de notas correctas, siempre ocurre que una nota se desvía ligeramente de esa progresión, causando ese malestar musical. Todo esto cambia en la segunda mitad del audio, donde se juega con variaciones de secuencias de notas parecidas, lo que crea un sonido agradable. No obstante, la monotonía y la duración constante de las notas crea un ambiente conformista y sin mucho más que ofrecer, nada vivaz. En general, la pieza musical creada es aceptable y agradable al oído, pero hacen falta más matices y cambios de ritmo para atraer más. Esta evaluación es generalizable al resto de audios generados sintéticamente a partir del conjunto de pruebas finales. Si bien la mayoría de ellos comparte rasgos como la falta de matices rítmicos y transiciones abruptas en secciones iniciales, se observa variabilidad en la intensidad de estos efectos.

5.4 Implementación del modelo GRU

El modelo neuronal basado en GRU recibe como entrada una secuencia de 63 notas, cada una con información acerca del tono, duración e intervalo. Por tanto, la entrada del modelo tiene dimensión (63,3). Hay una capa intermedia formada por 128 celdas GRU. El resultado es un vector de dimensión 128. Esta capa de células GRU se conecta a una capa densa de salida con función *Softmax* formada por 128 neuronas para representar los 128 valores posibles del número de nota que define el tono musical. Esta salida se compara con la codificación *one-hot* del tono real. En cuanto a la duración y el intervalo, la salida de la capa de celdas GRU, de dimensión 128, se conecta con una única neurona de salida que predice el respectivo resultado como un valor real. La combinación del valor de tono, duración e intervalo definen la salida del modelo como una nota musical. La Figura 32 muestra la estructura neuronal del modelo.

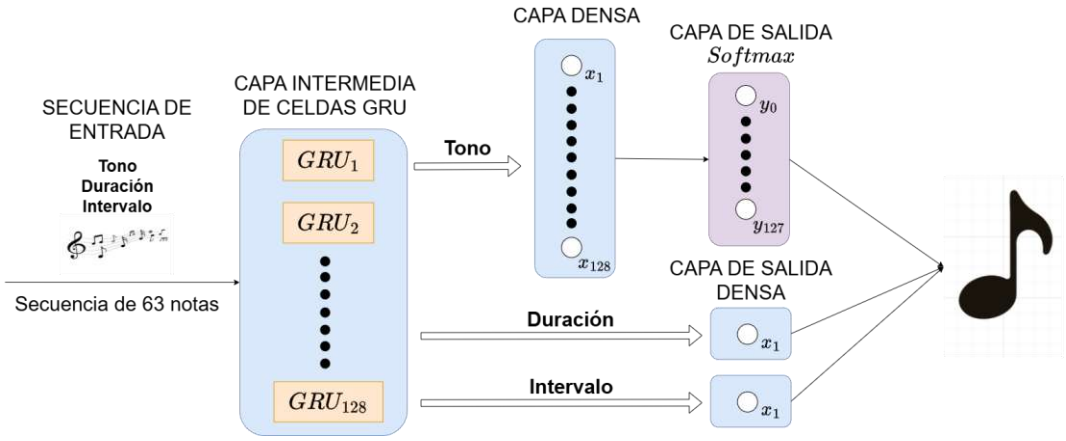


Figura 32. Estructura neuronal del modelo GRU.

En cuanto a las funciones de error del modelo, se vuelve a utilizar una diferente para el tono y otra tanto para la duración y el intervalo. En el caso del tono, se trata de un problema de clasificación, por lo que se utiliza la función *sparse categorical crossentropy* (5.1). Para las dos siguientes variables de salida (duración e intervalo), se ha diseñado una función de pérdida personalizada basada en el error cuadrático medio (MSE) que incentiva al modelo a no devolver valores negativos para la duración y el intervalo (5.2).

Al ejecutar el modelo se encuentra el mismo problema anterior relacionado con la pérdida total del modelo. La pérdida asociada al tono (*pitch*) es mayor en comparación con la de la duración y el intervalo, lo que provoca que el error total del modelo esté dominado principalmente por el tono. Para abordar esta situación, se ha optado por la misma solución utilizada anteriormente: ajustar el argumento *loss_weights* que proporciona *Keras* en la compilación del modelo. Por tanto, la función de pérdida total corresponde con (5.4).

En cuanto a los hiperparámetros utilizados en este modelo, se establece un tamaño de lote (*mini-batch*) de 512 para el entrenamiento del modelo, después de varias pruebas que mostraron que este valor ofrece los mejores resultados. La tasa de aprendizaje se establece en 0,001. Para mitigar el riesgo de sobreajuste, se aplica la técnica de regularización *dropout* con un valor de 0,15 en la capa intermedia de celdas GRU. Además, se utiliza el *early stopping* como técnica de detención temprana para evitar el sobreajuste. El optimizador utilizado es *RMSprop*.

El proceso de entrenamiento de la red neuronal se desarrolla durante 300 épocas, con el objetivo de que el modelo identifique patrones en secuencias musicales para predecir la siguiente nota. Una vez completado el entrenamiento, se genera una secuencia de 200 notas mediante el mismo método iterativo aplicado anteriormente mostrado en la Figura 21.

Un elemento relevante es el parámetro *temperature* (valor 5), previamente implementado, que regula la diversidad en las predicciones al ajustar la distribución de probabilidad de los *logits* asociados a las notas. Esta configuración busca equilibrar coherencia musical con innovación, priorizando resultados exploratorios sin restringir la estructura aprendida.

5.4.1 Resultados del modelo GRU

Las notas generadas sintéticamente (tono, duración e intervalo) se almacenan en una tabla (*dataframe)* para analizarlas después. En la Figura 33 se puede observar un ejemplo de una tabla generada, compuesta por 10 notas.

	pitch	step	duration	start	end
0	72	0.114841	0.493346	0.114841	0.608186
1	79	1.723860	0.000000	1.838701	1.838701
2	84	1.529921	0.000000	3.368622	3.368622
3	72	1.454016	0.000000	4.822637	4.822637
4	101	1.425439	0.000000	6.248076	6.248076
5	50	1.412220	0.000000	7.660297	7.660297
6	95	1.454551	0.000000	9.114848	9.114848
7	101	1.382421	0.000000	10.497269	10.497269
8	84	1.371388	0.000000	11.868657	11.868657
9	73	1.393999	0.000000	13.262655	13.262655

Figura 33. Tabla compuesta por 10 notas generadas sintéticamente mediante el modelo GRU.

Se puede apreciar que vuelve a ocurrir que la columna de *duration* tiene la mayoría de sus valores en 0 debido a la figura musical de pequeño valor que no es capaz de captar el modelo.

El audio completo generado se puede escuchar en el siguiente enlace:

music_gru_raw

El archivo de audio presenta el mismo problema que en el modelo LSTM al estar las notas distribuidas en octavas extremas, con sonidos demasiado graves o agudos que dificultan su identificación auditiva. Para abordar esta situación, se aplica el mismo preprocesamiento realizado anteriormente para ajustar el audio. Tras completar este proceso, se genera el siguiente archivo MIDI a partir del anterior (*music_gru_raw*):

music_gru_tuned

Por último, se presenta la partitura correspondiente al audio generado y ajustado, con el objetivo de analizar más claramente las notas producidas. La Figura 34 ilustra la partitura del archivo de audio *music_gru_tuned*. En esta partitura se aprecia que la figura dominante en toda la pieza musical es la fusa, con lo que posee un ritmo rápido y ligero. Un aspecto interesante es la ligadura ubicada al final del tercer pentagrama, que otorga a esa nota más duración que las adyacentes.

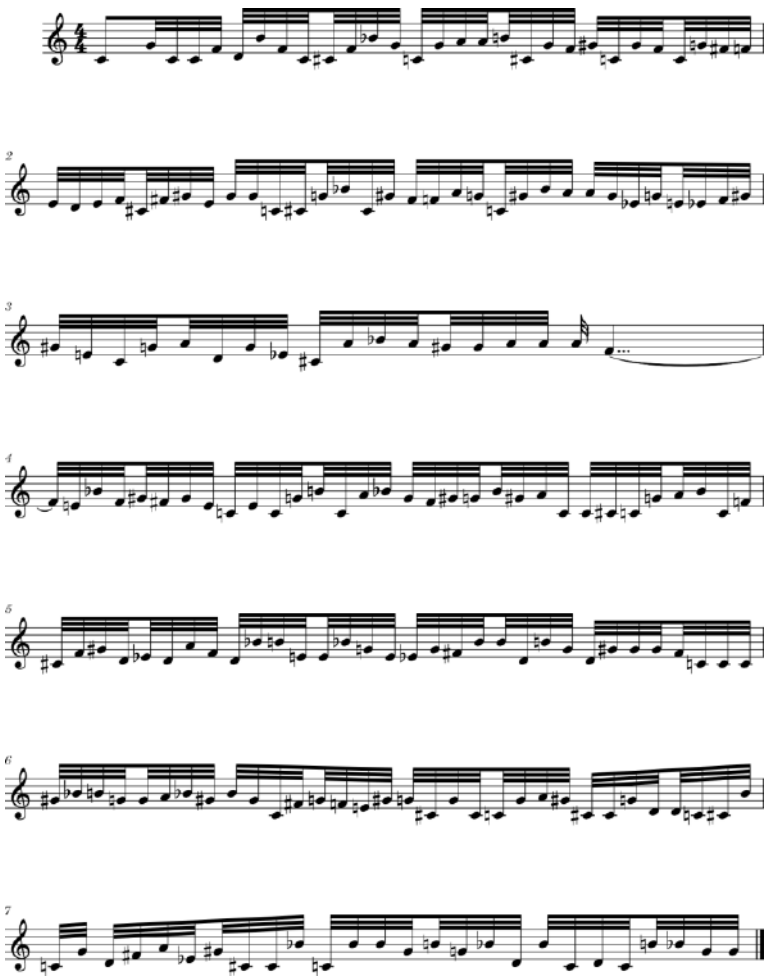
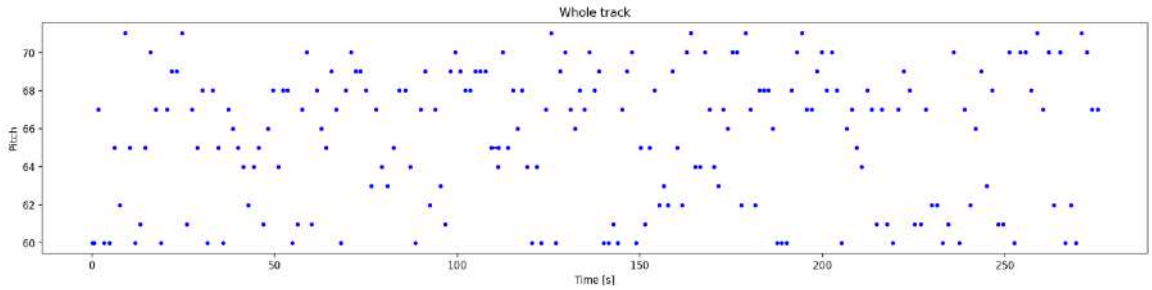
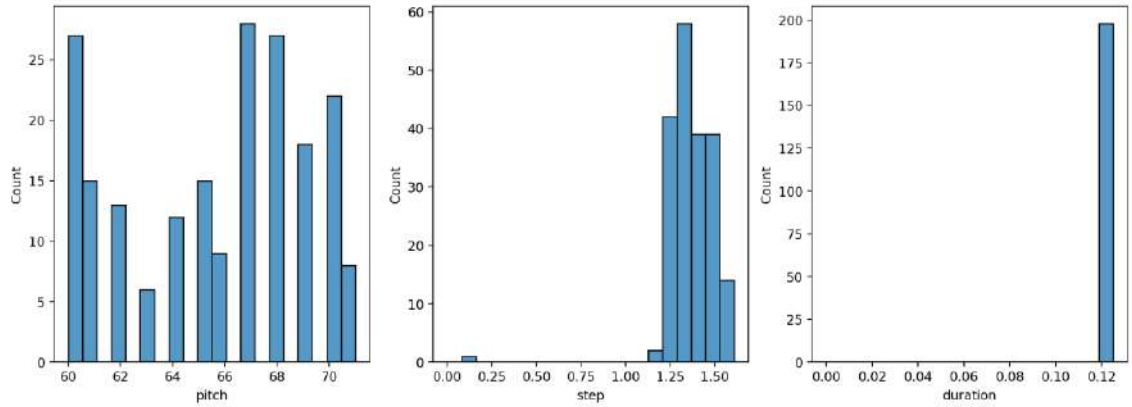


Figura 34. Partitura del audio generado mediante el modelo GRU y posteriormente ajustado para centralizar las notas en una octava media

Para facilitar la interpretación visual de la organización espacial de las notas generadas y sus variaciones en tono, duración e intervalo, la Figura 35 muestra un *piano roll* (a.) del audio procesado (*music_gru_tuned*) junto con gráficos de distribución de dichas variables (b.). El *piano roll* muestra una variación uniforme en las notas generadas, como así muestra la gráfica de distribución del tono de las notas. Se puede apreciar que el intervalo (*step*) es bastante mayor, alcanzado de media 1.3, mientras que el valor de duración se concentra exclusivamente en el valor 0.12, lo que indica un ritmo en su mayoría rápido formado por figuras musicales de duración corta.



a. *Piano roll del audio generado.*



b. *Distribución de las variables tono, duración e intervalo de las notas generadas.*

Figura 35. Piano roll y gráficas con la distribución de las variables tono, duración e intervalo de las notas generadas mediante el modelo GRU.

A continuación, se estudian los resultados obtenidos con el modelo *GRU* desde los mismos tres puntos de vista que en el caso del modelo LSTM: proceso de entrenamiento, métricas cuantitativas y crítica musical.

En primer lugar, se examina el proceso de entrenamiento mediante la evolución de la función de pérdida durante 300 épocas. La Figura 36, similar a la figura 30, muestra esta evolución.

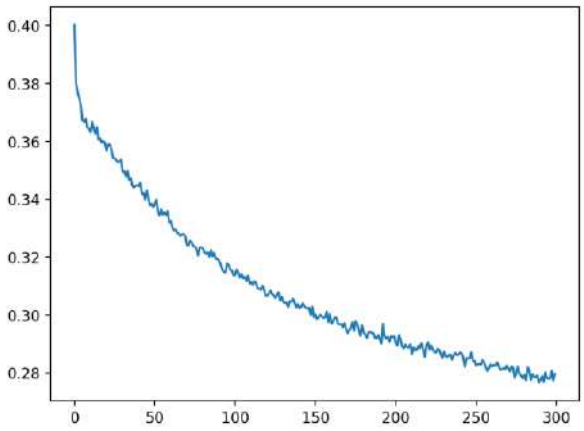


Figura 36. Gráfica de la evolución durante las 300 épocas de la función de pérdida total del conjunto de entrenamiento en el modelo GRU.

La interrupción del entrenamiento en la época 300, con una métrica de pérdida estabilizada alrededor de 0,28 para los datos de entrenamiento, responde a una estrategia preventiva. La decisión de finalizar anticipadamente busca mitigar riesgos de adaptación excesiva a patrones específicos del conjunto de entrenamiento, preservando así la flexibilidad predictiva del modelo frente a nuevas entradas.

En cuanto a las cinco métricas cuantitativas para medir la calidad del audio generado y ajustado (*music_gru_tuned*), la Figura 37 muestra los resultados obtenidos de uno de los 15 audios generados sintéticamente, seleccionado como el más representativo.

```
Polyphony Score: 98.12 %
Best Matching Scale: Abstract major
Scale Consistency: 58.12 %
Repetitions of Short Subsequences: 189
Tone Span: 11 semitones
```

Figura 37. Resultados de las métricas de evaluación objetiva sobre el audio generado más representativo mediante el modelo GRU.

El porcentaje alto de 98,12 % en polifonía de la pieza musical generada sintéticamente indica que durante el audio existen múltiples ocasiones donde al menos dos notas simultáneas suenan a la vez. En cuanto al tipo de escala que mejor se adecua al audio, el resultado de *abstract major* indica que es una pieza musical con matices alegres relacionados con emociones felices. A partir de los resultados obtenidos en consistencia, subsecuencias musicales únicas y rango tonal, se trata de una pieza con más repeticiones y que se amuebla más a los esquemas musicales seguidos sin tratar de improvisar. Los 14 audios restantes presentan características similares a las reflejadas en este análisis. Mantienen altos niveles de polifonía, se ajustan a escalas con matices alegres y muestran una estructura tonal coherente con patrones musicales predefinidos. Asimismo, exhiben una tendencia a repetir frases musicales en lugar de incorporar improvisaciones significativas, lo que refuerza la consistencia estilística observada en la pieza analizada.

En cuanto a la crítica musical a partir de la escucha activa del audio representativo generado sintéticamente y ajustado (*music_gru_tuned*), se identifica una estructuración coherente en su fase inicial, donde la selección de

sonidos afines contribuye a una buena base melódica. La progresión rítmica inicia con una cadencia moderada, evolucionando posteriormente hacia una transición gradual que desemboca en un tempo *andante*, caracterizado por su fluidez y continuidad expresiva. A nivel compositivo, el trabajo presenta recurrencias temáticas con variaciones sutiles en las frases musicales, patrón que, pese a su cohesión conceptual, podría interpretarse como predecible en ciertos segmentos. Sin embargo, destaca la incorporación estratégica de articulaciones en *staccato*, técnica que acorta la duración de la nota de forma repentina y agrega energía en pasajes específicos. Como valoración global, la ejecución demuestra una correcta coherencia estructural. No obstante, se sugiere la integración de elementos de ruptura melódica en momentos clave, lo que permite enriquecer la narrativa musical y mitigar la sensación de repetición, potenciando así la dimensión expresiva de la obra. De manera similar, los audios restantes mantienen una estructuración coherente en sus secciones iniciales, con una base melódica bien definida gracias a la selección de sonidos afines. La progresión rítmica sigue un patrón comparable, iniciando con una cadencia moderada y evolucionando hacia un tempo andante, lo que aporta fluidez y continuidad expresiva. En términos compositivos, también presentan recurrencias temáticas con variaciones sutiles, lo que refuerza la cohesión pero puede generar a la vez cierta previsibilidad.

5.5 Implementación del modelo multi-LSTM

El modelo neuronal *multi-LSTM* utiliza una secuencia de 63 notas musicales como entrada, donde cada nota incluye tres parámetros: tono, duración e intervalo (dimensión de entrada: 63, 3). La arquitectura, similar a modelos previamente implementados, está compuesta por dos capas LSTM secuenciales, cada una con 64 unidades. La primera capa procesa la entrada, generando un vector de dimensión 64, que es inviado a la segunda capa LSTM. La salida de esta segunda capa se divide en dos rutas. Por un lado, se conecta a una capa densa de 128 neuronas correspondiente al rango estándar de valores MIDI (0-127). Los *logits* generados por esta capa son transformados en probabilidades mediante una función *Softmax*, permitiendo identificar el tono más probable, que se compara con el tono real codificado mediante *one-hot*. Por otro lado, la misma salida de dimensión 64 se utiliza como entrada para dos neuronas más, encargadas de predecir los valores reales correspondientes a la duración y el intervalo respectivamente. La Figura 38 detalla esta estructura, destacando la interconexión entre capas y el flujo de datos desde la entrada hasta la generación de la nota musical a la salida.

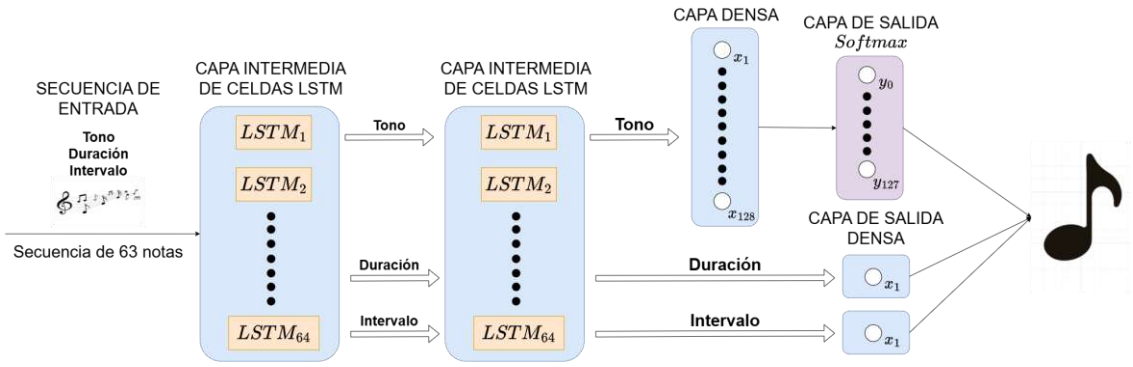


Figura 38. Estructura neuronal del modelo multi-LSTM.

El modelo emplea la función de pérdida *sparse categorical crossentropy* (5.1) para el tono. En cuanto a la duración y el intervalo, se utiliza la función de pérdida personalizada (5.2).

El modelo presenta el mismo inconveniente observado en las arquitecturas anteriores, relacionado con la función de pérdida total. En este sentido, la función de error utilizada corresponde a la expresada en (5.4).

La configuración de hiperparámetros seleccionada para el modelo incluye un tamaño de lote (*mini-batch*) de 256, determinado tras varias pruebas que demostraron estabilidad en el aprendizaje. La tasa de aprendizaje se fija en 0,001. Para contrarrestar el sobreajuste, se implementa la técnica de regularización *dropout* con un valor de 0,15 en las capas de celdas LSTM. Adicionalmente, se incorpora un mecanismo de parada anticipada (*early stopping*). El optimizador elegido es *RMSprop*.

El proceso de entrenamiento de la red neuronal se desarrolla durante 300 épocas, donde el modelo aprende estructuras musicales para predecir la siguiente nota. Al finalizar el entrenamiento, se genera una secuencia de 200 notas a través del método iterativo mostrado en la Figura 21.

Un componente clave es el hiperparámetro *temperature* (con valor 5), , cuya función es modular la diversidad de las predicciones mediante el ajuste de la distribución de probabilidad de los *logits* correspondientes al, tono de las notas.

5.5.1 Resultados del modelo multi-LSTM

Las notas generadas sintéticamente (tono, duración e intervalo) se almacenan en una tabla (*dataframe)* para su posterior análisis. En la Figura 39 se puede observar un ejemplo de una tabla generada, compuesta por 10 notas.

	pitch	step	duration	start	end
0	72	0.089888	0.099973	0.089888	0.189861
1	79	0.139295	0.532379	0.229183	0.761562
2	47	0.240774	0.503808	0.469957	0.973765
3	95	0.351977	0.415273	0.821934	1.237207
4	101	0.396270	0.340692	1.218204	1.558895
5	50	0.377300	0.307181	1.595504	1.902685
6	95	0.326323	0.310573	1.921827	2.232400
7	101	0.310672	0.260952	2.232498	2.493450
8	84	0.297699	0.278510	2.530197	2.808707
9	73	0.263193	0.288226	2.793390	3.081615

Figura 39. Tabla compuesta por 10 notas generadas sintéticamente mediante el modelo multi-LSTM.

Se puede observar que la columna de *duration* contiene valores no nulos. El modelo está compuesto por varias capas de celdas *LSTM*, lo que puede haber permitido una captura más precisa de esa variable en las notas generadas, reflejándose en los valores obtenidos.

El audio completo generado se puede escuchar en el siguiente enlace:

music_multi_lstm_raw

El archivo de audio presenta el mismo problema al estar las notas distribuidas en octavas extremas, con sonidos demasiado graves o agudos que dificultan su identificación auditiva. Para abordar esta situación, se aplica el mismo preprocesamiento realizado en los anteriores modelos para ajustar el audio. Tras completar este proceso, se genera el siguiente archivo MIDI a partir del anterior (*music_multi_lstm_rau*):

music_multi_lstm_tuned

Finalmente, se muestra la partitura del audio previamente generado y ajustado, lo cual permite una mejor comprensión de las notas obtenidas. La Figura 40 presenta la partitura correspondiente al archivo *music_multi_lstm_tuned*. En esta partitura se puede observar que la figura dominante es la semicorchea, por lo que posee un ritmo ágil. Este ritmo se mantiene constante a lo largo de la pieza.

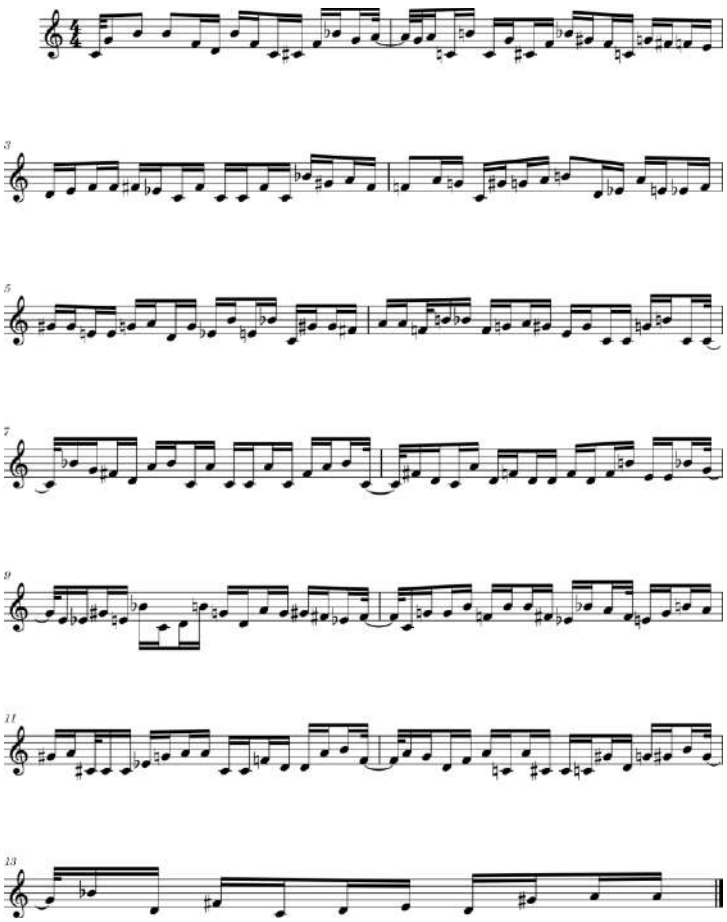
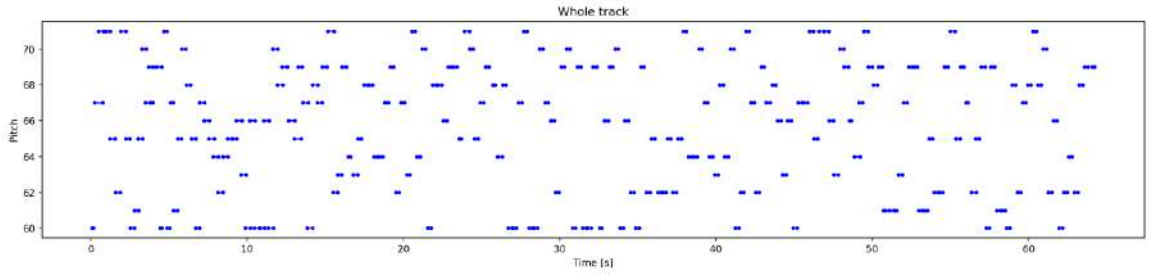
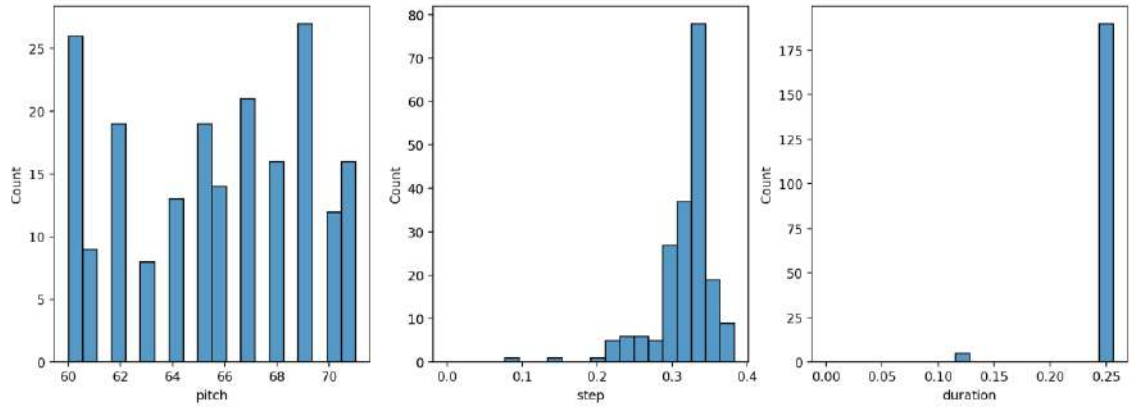


Figura 40. Partitura del audio generado mediante el modelo multi-LSTM y posteriormente ajustado para centralizar las notas en una octava media.

Con el fin de mostrar de manera más clara cómo se organizan las notas generadas y cómo varían las variables de tono, duración e intervalo, la Figura 41 presenta un piano roll del audio ajustado (*music_multi_lstm_tuned*) en (a.), acompañado de una gráfica que representa la distribución de los valores de estas tres variables en (b.). En este *piano roll*, las notas muestran una variación uniforme, lo que también se refleja en la gráfica de distribución del tono. Se observa que el intervalo (*step*) se sitúa alrededor del 0.35, mientras que el valor de duración en 0.25. Esto sugiere un ritmo predominantemente rápido, compuesto por figuras musicales de corta duración.



a. *Piano roll del audio generado.*



b. *Distribución de las variables tono, duración e intervalo de las notas generadas.*

Figura 41. Piano roll y gráficas con la distribución de las variables tono, duración e intervalo de las notas generadas mediante el modelo multi-LSTM.

A continuación, se estudian los resultados obtenidos con el modelo *multi-LSTM* desde los mismos tres puntos de vista que en los modelos anteriores: proceso de entrenamiento, métricas cuantitativas musicales y valoración subjetiva mediante escucha activa.

En primer lugar, se examina el proceso de entrenamiento mediante la evolución de la función de pérdida. La Figura 42, parecida a la Figura 36, muestra la evolución durante 300 épocas de la pérdida total para el conjunto de entrenamiento.

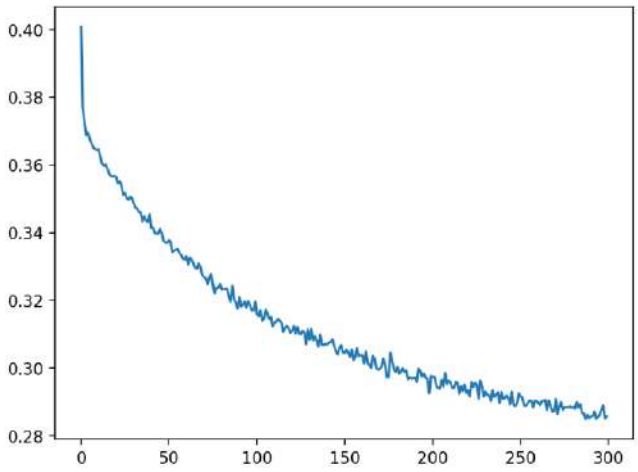


Figura 42. Gráfica de la evolución durante las 300 épocas de la función de pérdida total del conjunto de entrenamiento en el modelo multi-LSTM.

El entrenamiento se interrumpió de manera preventiva en la época 300, donde se obtiene un valor aproximado para la pérdida de 0,28 sobre los datos de entrenamiento. Esta decisión anticipada busca evitar el sobreajuste y conservar la capacidad generalizadora del modelo frente a datos no utilizados en el entrenamiento del modelo.

Respecto a las cinco métricas cuantitativas para evaluar la calidad del audio generado y ajustado (*music_multi_lstm_tuned*), la Figura 43 muestra los resultados obtenidos del audio más representativo entre los 15 generados sintéticamente.

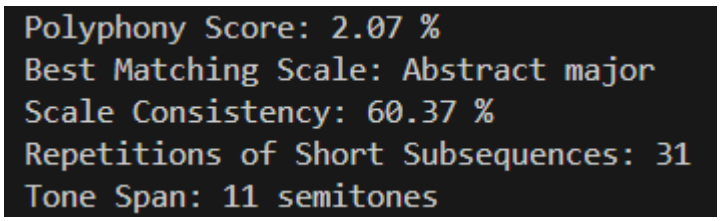


Figura 43. Resultados de las métricas de evaluación objetiva sobre el audio generado más representativo mediante el modelo multi-LSTM.

La primera cuantifica el índice de polifonía, registrando un 2,07 %, lo que evidencia que la presencia simultánea de dos o más notas es escasa a lo largo de la pieza. La segunda identifica la tipología de escala predominante, clasificada como *abstracta mayor*, asociada a tonalidades vinculadas con expresiones emocionales positivas. La tercera evalúa la coherencia armónica con la escala identificada, determinando el porcentaje de alineación estructural. Adicionalmente, se examina la diversidad compositiva mediante dos indicadores: la densidad de frases musicales únicas y el rango tonal. Los resultados reflejan una predominancia de motivos no repetitivos y una amplitud tonal amplia, sugiriendo que la pieza prioriza la innovación melódica sobre la recurrencia a esquemas predefinidos. Esta tendencia se manifiesta en estructuras innovadoras que evitan patrones cíclicos, potenciando la originalidad en la progresión armónica. Paralelamente, los audios restantes generados presentan variaciones en estos parámetros, aunque mantienen una línea coherente con estos resultados. Por ejemplo, en algunos casos, el porcentaje de polifonía aumenta ligeramente reflejando momentos de mayor superposición melódica sin comprometer la claridad armónica. Respecto a la estructura, la mayoría de las piezas muestran un rango tonal amplio, con transiciones fluidas entre intervalos que evitan la monotonía. Adicionalmente, el análisis de frases únicas confirma una tendencia generalizada a minimizar las repeticiones, favoreciendo evoluciones temáticas que exploran variaciones rítmicas y dinámicas. Esto sugiere que el modelo prioriza la innovación dentro de un marco reconocible, equilibrando originalidad y cohesión.

En relación con la evaluación subjetiva, basada en la crítica musical y en la escucha activa del audio representativo generado sintéticamente y ajustado (*music_multi_lstm_tuned*), se aprecia que esta pieza se caracteriza, desde su inicio, por un ritmo alegre y rápido, respaldado por una base armónica sólida, que aporta dinamismo y cohesión al conjunto. El planteamiento inicial introduce una frase melódica creativa en sus primeras repeticiones, lo que genera un ambiente divertido y estimulante. El pulso, de carácter ameno y placentero al oído, mantiene una energía constante. Sin embargo, ciertos deslices armónicos interrumpen brevemente la fluidez en algunas transiciones específicas. Sobre el desarrollo compositivo, si bien la frase inicial se explora con variaciones rítmicas y extensiones ingeniosas en su primera mitad, el

discurso posterior se limita a repeticiones melódicas con cambios mínimos, carentes de innovación. Este patrón genera cierta monotonía hacia la sección central, donde no se perciben elementos de contraste. Pese a ello, el carácter general alegre y la estructura accesible aseguran una escucha satisfactoria. Como sugerencia, se propone profundizar en el juego rítmico-melódico de la frase principal para evitar la sensación de estancamiento. Además, corregir los desajustes armónicos puntuales pueden enriquecer la propuesta, equilibrando su vitalidad inicial con una evolución más impredecible y matizada. En cuanto a la valoración de los audios generados restantes, se observa una tendencia similar: ritmos y bases armónicas bien definidas que sostienen la energía de las piezas desde los primeros compases. La mayoría inicia con buenas ideas melódicas, caracterizadas por giros originales y un pulso contagioso que invita al movimiento. No obstante, como en el caso anterior, varios de ellos sufren de irregularidades en el desarrollo temático. Por ejemplo, en algunas composiciones, las variaciones rítmicas iniciales dan paso a repeticiones literales de la frase principal sin adiciones significativas, limitando la narrativa musical a un ciclo predecible.

5.6 Implementación del modelo multi-GRU

El modelo neuronal *multi-GRU* acepta como entrada secuencias de 63 notas, cada una representada mediante tres características principales: tono, duración e intervalo, por lo que la entrada tiene dimensión (63, 3). La primera capa del modelo está compuesta por 64 unidades GRU que procesan la entrada para extraer representaciones intermedias de las tres características. El resultado es un vector de dimensión 64 . Estos vectores se procesan posteriormente mediante una segunda capa GRU, también compuesta por 64 celdas, que genera una representación final de dimensión 64 para el tono, la duración y el intervalo. En el caso del tono, este vector se conecta a una capa densa de salida con 128 neuronas, correspondientes a los posibles valores MIDI (0–127). Esta capa genera los *logits*, que luego se transforman en probabilidades utilizando la función *Softmax*. La predicción resultante se compara con la codificación *one-hot* del tono real, permitiendo interpretar la salida del modelo como una distribución de probabilidad sobre las posibles notas MIDI y facilitando su conversión directa a este formato. En cuanto a la duración y el intervalo, la salida de la segunda capa GRU (dimensión 64) se conecta también con dos neuronas, encargadas, cada una, de predecir sus respectivos valores como valores reales. Por lo tanto, la salida final del modelo, compuesta por tono, duración e intervalo, define una nota musical completa. La Figura 44 ilustra la arquitectura del modelo.

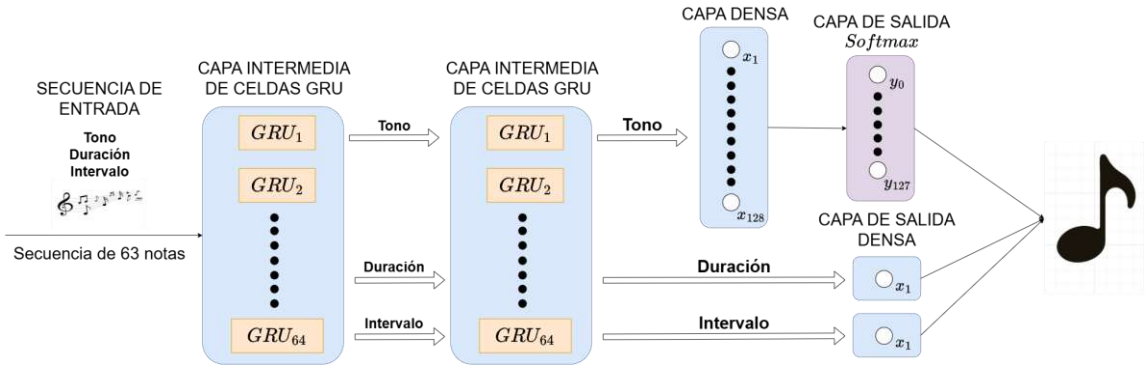


Figura 44. Estructura neuronal del modelo multi-GRU.

En cuanto a las funciones de error del modelo, se emplea nuevamente una función distinta para cada tipo de salida. Para el tono, al tratarse de un problema de clasificación, se utiliza la función *sparse categorical crossentropy* (5.1), mientras que para duración e intervalo se aplica una función de pérdida personalizada basada en el error cuadrático medio (MSE), diseñada para penalizar predicciones negativas (5.2).

Durante la ejecución, se detecta el mismo problema identificado en modelos anteriores: la pérdida asociada al tono predomina sobre las demás, afectando el cálculo de la pérdida total. Para compensar este desequilibrio, se recurre nuevamente al ajuste del parámetro *loss_weights* proporcionado por *Keras*, definiendo así la función de pérdida total (5.4).

Respecto a los hiperparámetros, se mantiene un tamaño de lote (*mini-batch*) de 512, seleccionado tras varias pruebas por su rendimiento óptimo. La tasa de aprendizaje se fija en 0,001. Para prevenir el sobreajuste, se aplica la técnica de *dropout* con una tasa del 0,15 en las capas intermedias de celdas GRU, y se utiliza *early stopping* para una detención anticipada del entrenamiento. El optimizador empleado es RMSprop.

El entrenamiento del modelo se lleva a cabo durante 300 épocas, en las que la red aprende a modelar patrones musicales a partir de las secuencias de entrada, con el objetivo de generar la siguiente nota. Finalizado el proceso, se produce una secuencia de 200 notas siguiendo el mismo mecanismo iterativo descrito anteriormente y representado en la Figura 21.

Un componente clave en esta etapa es el uso del parámetro *temperature* (valor 5), ya implementado en modelos previos, que ajusta la dispersión de la distribución de probabilidad sobre los *logits* del tono. Esta configuración permite generar resultados más variados, favoreciendo la creatividad musical sin comprometer la coherencia aprendida.

5.6.1 Resultados del modelo multi-GRU

Las notas generadas sintéticamente (tono, duración e intervalo) se almacenan en una tabla (*dataframe)* para su posterior análisis. En la Figura 45 se puede observar un ejemplo de una tabla generada, compuesta por 10 notas.

	pitch	step	duration	start	end
0	72	0.969773	1.775822	0.969773	2.745595
1	79	0.983061	3.821976	1.952834	5.774810
2	47	0.943655	4.201111	2.896488	7.097600
3	95	1.035078	2.534145	3.931566	6.465711
4	12	0.968192	2.640521	4.899758	7.540279
5	50	0.937734	1.703041	5.837492	7.540532
6	95	0.647540	2.448287	6.485032	8.933318
7	101	0.567858	2.383989	7.052889	9.436879
8	84	0.745638	4.064823	7.798527	11.863350
9	73	0.827402	3.440315	8.625929	12.066243

Figura 45. Tabla compuesta por 10 notas generadas sintéticamente mediante el modelo multi-GRU.

Como se puede notar, en este caso la columna *duration* presenta valores distintos de nulo. El modelo está formado por múltiples capas de celdas GRU, lo que puede haber favorecido una captura más precisa de esta variable en las notas generadas, reflejándose en los valores obtenidos

El audio completo generado se puede escuchar en el siguiente enlace:

music_multi_gru_raw

El archivo de audio presenta el mismo problema al estar las notas distribuidas en octavas extremas, con sonidos demasiado graves o agudos que dificultan su identificación auditiva. Para abordar esta situación, se aplica el mismo preprocesamiento realizado en los modelos previos para ajustar el audio. Tras completar este proceso, se genera el siguiente archivo MIDI a partir del anterior (*music_multi_gru_raw*):

music_multi_gru_tuned

Finalmente, se visualiza la partitura del audio generado y ajustado para comprender mejor las notas generadas. La Figura 46 muestra un extracto de la partitura del audio anterior *music_multi_gru_tuned*. El extracto de la partitura generada se distingue por el uso predominante de figuras musicales de mayor duración, como blancas y negras, junto con ligaduras distribuidas a lo largo de toda la obra. Como resultado, el ritmo es más pausado y uniforme.

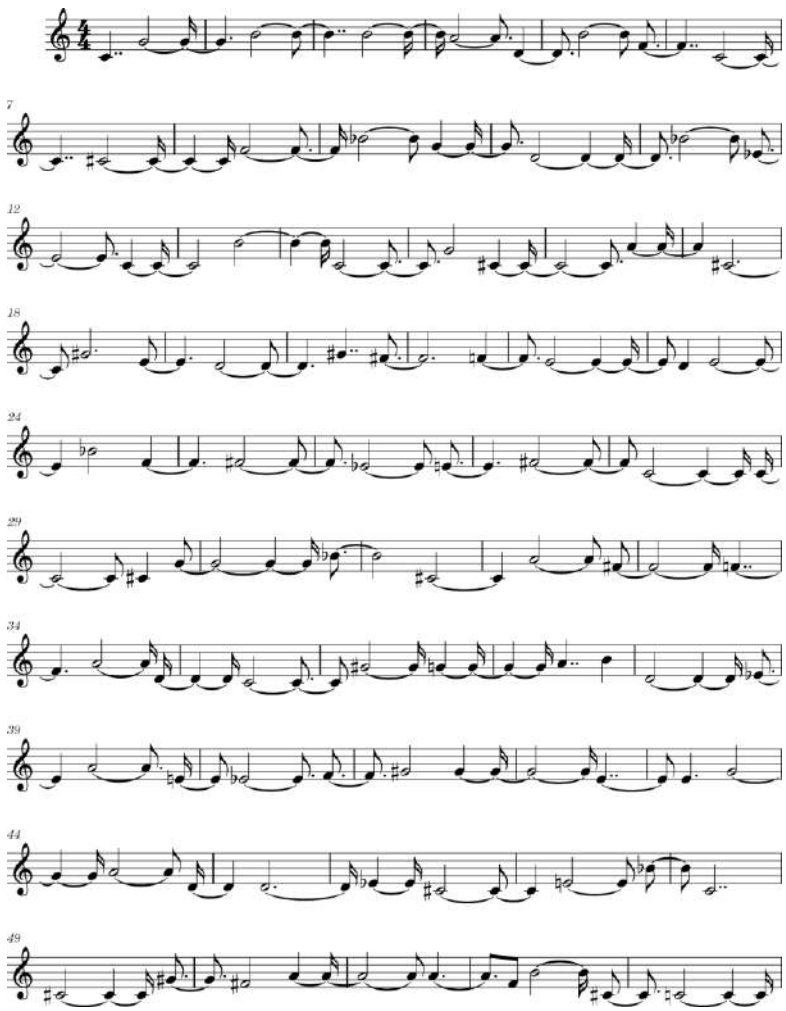
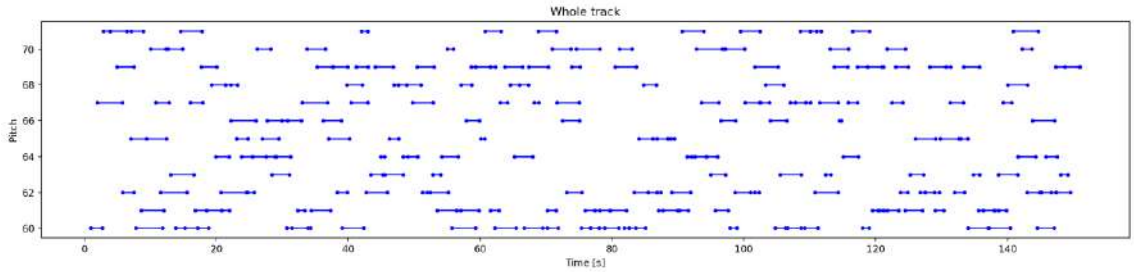
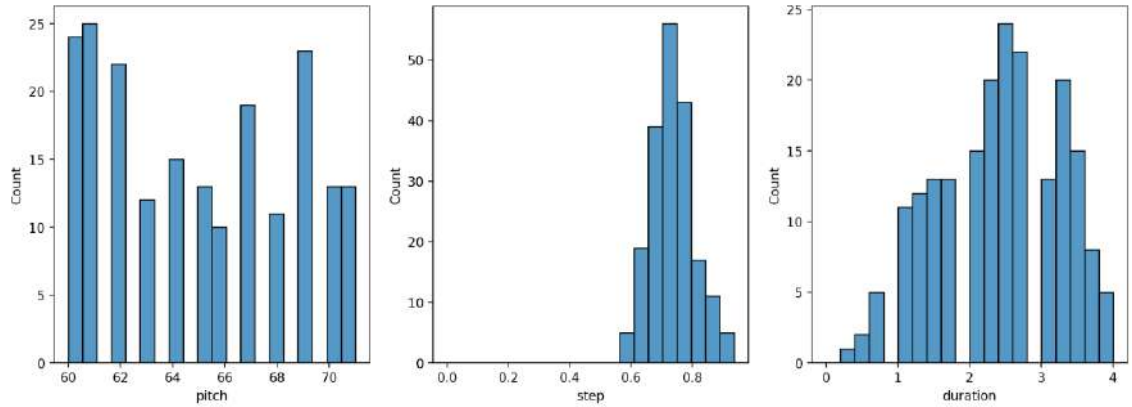


Figura 46. Extracto de la partitura del audio generado mediante el modelo multi-GRU y posteriormente ajustado para centralizar las notas en una octava media.

Para visualizar de forma más clara la organización de las notas generadas y la variabilidad presente en las características de tono, duración e intervalo, en la Figura 47 se presenta un *piano roll* (a.) correspondiente al audio ajustado (*music_multi_gru_tuned*), acompañado de una gráfica (b.) que muestra la distribución de los valores para cada una de estas tres variables. En este *piano roll*, las notas presentan una distribución monótona, lo que concuerda con la partitura generada. Además, en la gráfica de *duration* se observan valores predominantemente altos, en su mayoría superiores a 1. Esto indica que la pieza posee un carácter sereno con una ejecución más sostenida y prolongada.



a. *Piano roll del audio generado.*



b. *Distribución de las variables tono, duración e intervalo de las notas generadas.*

Figura 47. Piano roll y gráficas con la distribución de las variables tono, duración e intervalo de las notas generadas mediante el modelo multi-GRU.

A continuación, se analizan los resultados obtenidos con el modelo *multi-GRU*, siguiendo los mismos criterios utilizados en los modelos anteriores: evaluación del proceso de entrenamiento, análisis de métricas cuantitativas y valoración desde una perspectiva musical.

En primer lugar, se examina el proceso de entrenamiento mediante la evolución de la función de pérdida, reflejando la capacidad del modelo para aprender patrones musicales. La Figura 48 muestra la evolución durante 300 épocas de la pérdida total para el conjunto de entrenamiento.

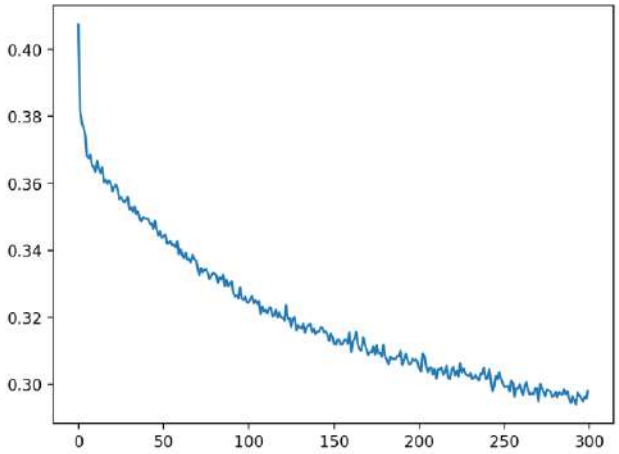


Figura 48. Gráfica de la evolución durante las 300 épocas de la función de pérdida total del conjunto de entrenamiento en el modelo multi-GRU.

El entrenamiento se detiene en la época 300, momento en el que la pérdida sobre el conjunto de entrenamiento se encuentra alrededor de 0,30. Se decide finalizar el proceso en este punto como medida preventiva, con el objetivo de evitar un posible sobreajuste y preservar así la capacidad del modelo para generalizar adecuadamente frente a nuevos datos.

En cuanto a la evaluación objetiva de la calidad musical, el análisis se enfoca en la comparación de 15 composiciones sintéticas generadas, eligiendo *music_multi_gru_tuned* como muestra representativa. Los resultados cuantitativos obtenidos de este análisis se resumen en la Figura 49.

```
Polyphony Score: 0.05 %
Best Matching Scale: Abstract major
Scale Consistency: 47.52 %
Repetitions of Short Subsequences: 24
Tone Span: 11 semitones
```

Figura 49. Resultados de las métricas de evaluación objetiva sobre el audio generado más representativo mediante el modelo multi-GRU.

El análisis se fundamenta en cinco indicadores clave. El primero evalúa el grado de superposición de notas simultáneas (polifonía), registrando un 0,05 %, lo que sugiere una mínima concurrencia de eventos sonoros. Este bajo porcentaje implica que las notas tienden a prolongarse en el tiempo, limitando las superposiciones. El segundo indicador determina la escala predominante, identificada como *abstract major*, una tonalidad asociada a estados emocionales positivos y vitalidad sonora. El tercero analiza la coherencia armónica con dicha escala, cuantificando el porcentaje de alineación. Adicionalmente, se miden la densidad de motivos melódicos únicos y la amplitud de variación tonal. Los resultados destacan una escasa repetición temática y una diversificación en las secuencias, evidenciando una preferencia por la exploración de patrones innovadores frente a la adherencia rígida a estructuras musicales predispuestas. En comparación con las otras composiciones generadas por el modelo, los resultados reflejan tendencias análogas, aunque con discrepancias marginales en los valores métricos específicos. En cuanto a los demás audios generados, los resultados obtenidos son muy similares, con algunas variaciones en las métricas. En general, la polifonía sigue siendo baja, indicando una

predominancia de notas largas, y la escala predominante continúa siendo *abstract major*, reflejando un carácter animado en la mayoría de las composiciones. Sin embargo, se observan ligeras diferencias en la diversidad de secuencias musicales y en el rango tonal, lo que sugiere que algunas piezas presentan una estructura melódica más variada o exploran patrones ligeramente distintos.

Con respecto a la evaluación subjetiva, se lleva a cabo un análisis basado en la crítica musical a partir de la escucha activa del audio representativo generado y ajustado sintéticamente (*music_multi_gru_tuned*). La pieza presenta un ritmo constante marcado por notas de gran duración, lo que atribuye una sensación de estabilidad pero también de rigidez en su desarrollo. Desde el inicio, la ausencia de ligaduras entre notas de distinto tono genera una articulación entrecortada y poco fluida, afectando la naturalidad del discurso musical. Aunque la melodía no cae en la repetición evidente, el ritmo muestra una uniformidad excesiva, lo que deriva en una escucha monótona y poco atractiva. En el plano armónico, la relación entre las notas carece de una cohesión clara, dando la impresión de una sucesión de tonos sin una conexión orgánica. Esta falta de dirección armónica contribuye a la sensación de aleatoriedad, debilitando la estructura general de la pieza. Desde un punto de vista técnico, la utilización de varias capas de celdas GRU dificulta el aprendizaje de las relaciones melódicas y armónicas subyacentes en los audios de piano. Como resultado, si bien la composición no llega a ser discordante, sí se percibe estancada en una linealidad carente de matices expresivos o de una evolución significativa. En relación con los demás audios generados del conjunto de pruebas finales, los resultados muestran un comportamiento similar, con algunas variaciones sutiles. En líneas generales, las piezas suelen mantener un ritmo constante basado en notas prolongadas, lo que les da solidez pero también cierta rigidez en su construcción. Como se observa, la forma de enlazar los sonidos sigue presentando saltos entre tonos distintos, lo que interrumpe la fluidez del conjunto. Este enfoque, si bien aporta claridad, termina dando una sensación de fragmentación en varios pasajes. Sobre el ritmo, aunque predomina un patrón repetitivo en la mayoría de los casos, vale la pena mencionar que algunas piezas logran introducir variaciones sutiles. Estos pequeños ajustes en la duración de las notas añaden matices que rompen levemente la uniformidad, dando la impresión de cambios dinámicos sin alterar la esencia base. Por otro lado, en lo que respecta a la armonía, la conexión entre las notas se mantiene como un aspecto a mejorar. Si bien es cierto que en ciertos fragmentos las transiciones suenan más vitales, persiste una tendencia a que los sonidos funcionen como elementos aislados en lugar de integrarse en un diálogo coherente con el resto de la melodía.

5.7 Implementación del modelo LSTM-GRU

El modelo neuronal *LSTM-GRU* recibe como entrada una secuencia de 63 notas, cada una caracterizada por su tono, duración e intervalo. La entrada tiene, por tanto, dimensión (63, 3). La secuencia es procesada inicialmente por una capa LSTM intermedia compuesta por 64 celdas, que actúa sobre las tres características para generar un vector de dimensión 64 para tono, duración e intervalo. Esta salida se transmite a una segunda capa LSTM, también de 64 celdas. A continuación, se aplican dos capas adicionales de 64 celdas GRU. La salida de la segunda capa GRU se conecta a una capa densa de salida con 128 neuronas, que representan cada uno de los posibles valores MIDI (0 a 127). Esta capa genera los *logits*, los cuales se transforman en una distribución de

probabilidad mediante la función *Softmax*. Para las predicciones de duración e intervalo, la salida de la segunda capa GRU se conecta también con dos neuronas, respectivamente. En conjunto, las predicciones de tono, duración e intervalo constituyen la nota musical generada por el modelo. La estructura del modelo se ilustra en la Figura 50.

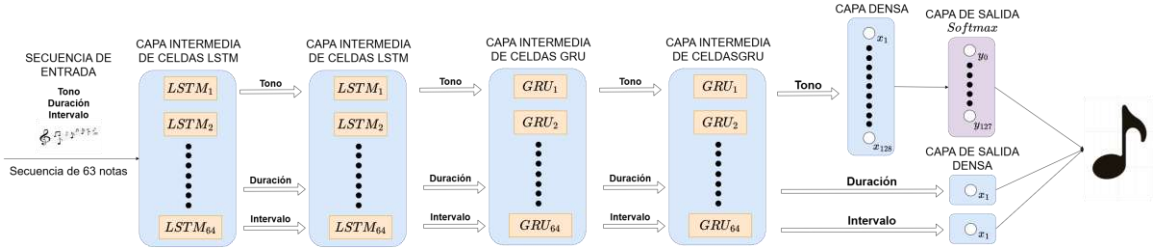


Figura 50. Estructura neuronal del modelo LSTM-GRU.

El modelo emplea la función de pérdida *sparse categorical crossentropy* (5.1) para el tono. Para la duración y el intervalo, se utiliza la función de pérdida personalizada (5.2).

Como en modelos anteriores, se presenta un desequilibrio en la contribución de cada salida a la pérdida total, dominada por el error del tono. Para corregirlo, se ajustan los pesos de las funciones de pérdida mediante el parámetro *loss_weights* de Keras, resultando en la función total descrita en (5.4).

En cuanto a los hiperparámetros, se utiliza un tamaño de lote (*mini-batch*) de 1024 y una tasa de aprendizaje de 0,005, seleccionados tras varias pruebas. Para reducir el riesgo de sobreajuste, se aplica *dropout* con una tasa de 0,2 en todas las capas intermedias, junto con *early stopping* como criterio de parada anticipada. El optimizador elegido es RMSprop.

El entrenamiento se lleva a cabo durante 300 épocas, permitiendo al modelo aprender patrones musicales y generar nuevas notas. Posteriormente, se genera una secuencia de 200 notas mediante un proceso iterativo que sustituye la nota inicial por la nueva predicción tal y como se muestra en la Figura 21.

Para controlar la diversidad en la generación musical, se emplea el parámetro *temperature* (valor 5), que ajusta la distribución de los *logits* del tono antes de aplicar *Softmax*. Este valor elevado favorece la creatividad, permitiendo composiciones más variadas.

5.7.1 Resultados del modelo LSTM-GRU

Las notas generadas sintéticamente (tono, duración e intervalo) se almacenan en una tabla (*dataframe)* para analizarlas posteriormente. En la Figura 51 se puede observar un ejemplo de una tabla generada, compuesta por 10 notas.

	pitch	step	duration	start	end
0	72	0.177544	0.280500	0.177544	0.458044
1	79	0.257249	0.113724	0.434793	0.548517
2	47	0.093101	0.399130	0.527894	0.927024
3	95	0.352705	0.462813	0.880599	1.343412
4	101	0.362239	0.531512	1.242838	1.774350
5	50	0.425877	0.534412	1.668715	2.203127
6	95	0.391012	0.611818	2.059727	2.671545
7	101	0.377739	0.338384	2.437466	2.775850
8	91	0.321025	0.563679	2.758490	3.322169
9	73	0.433469	0.653646	3.191959	3.845605

Figura 51. Tabla compuesta por 10 notas generadas sintéticamente mediante el modelo LSTM-GRU.

La columna *duration* muestra valores distintos de nulo. La arquitectura del modelo, compuesta por varias capas de celdas *LSTM* y *GRU*, ha facilitado una mejor captura de esta variable en las notas generadas, lo que se refleja en los valores obtenidos.

El audio completo generado se puede escuchar en el siguiente enlace:

music_lstm_gru_raw

El archivo de audio presenta el mismo problema al estar las notas distribuidas en octavas extremas, con sonidos demasiado graves o agudos que dificultan su identificación auditiva. Para abordar esta situación, se aplica el mismo preprocesamiento realizado en los anteriores modelos para ajustar el audio. Tras completar este proceso, se genera el siguiente archivo MIDI a partir del anterior (*music_lstm_gru_raw*):

music_lstm_gru_tuned

Finalmente, se muestra la partitura del audio generado y ajustado para comprender mejor las notas generadas. La Figura 52 presenta la partitura del audio anterior *music_lstm_gru_tuned*. La partitura generada destaca por el predominio de semicorcheas y fusas, lo que le otorga un ritmo ágil y ligero.

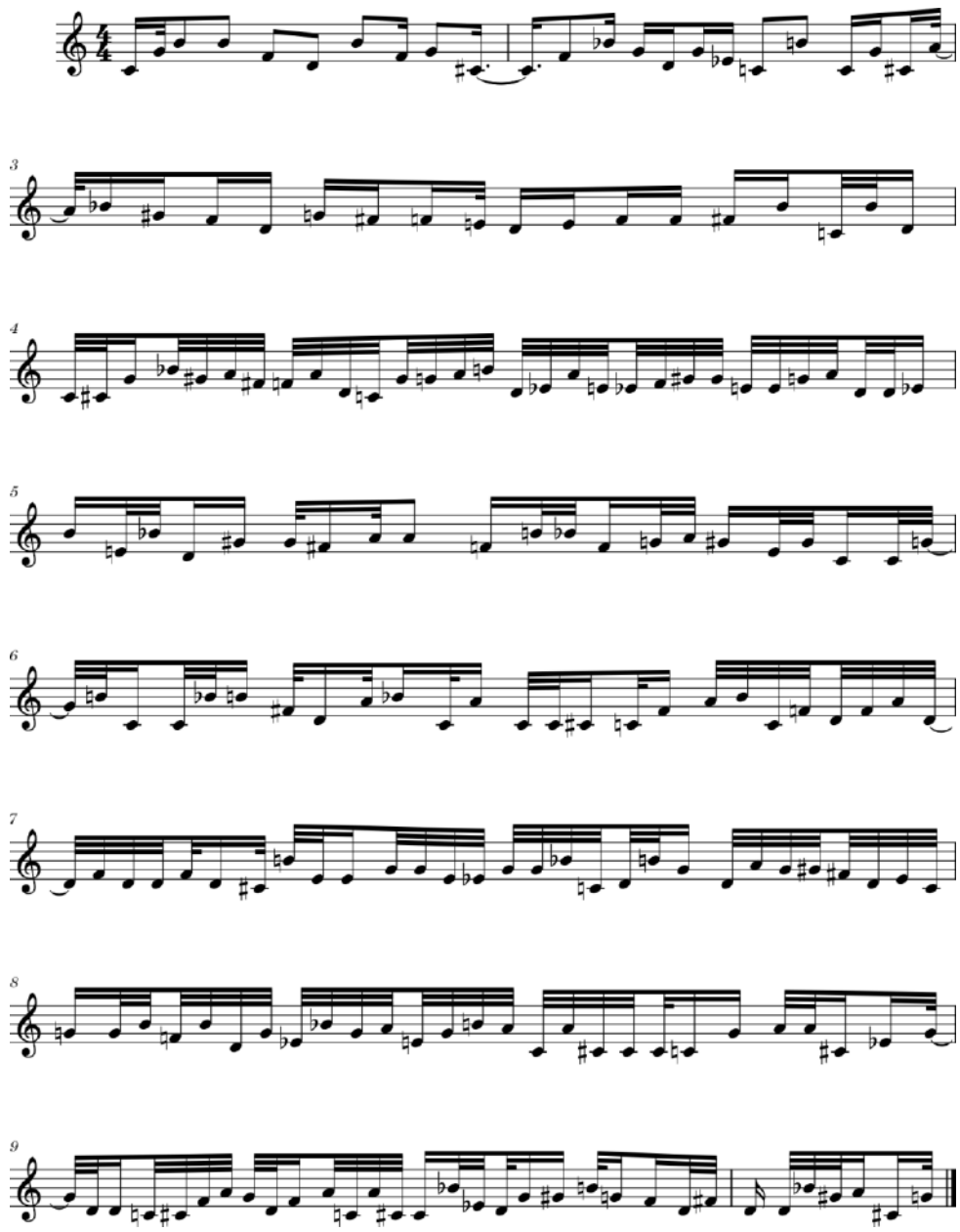
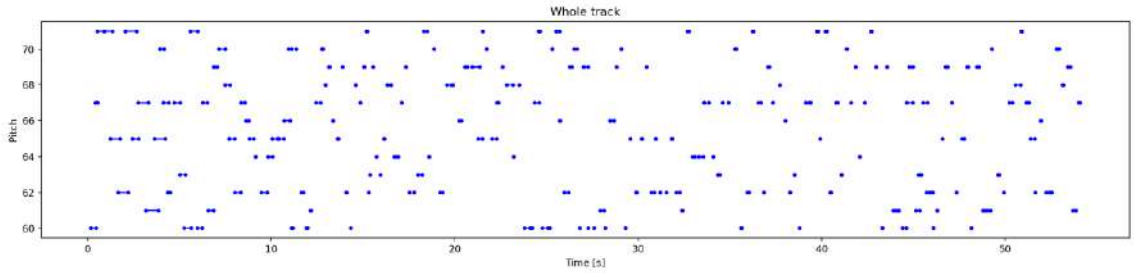
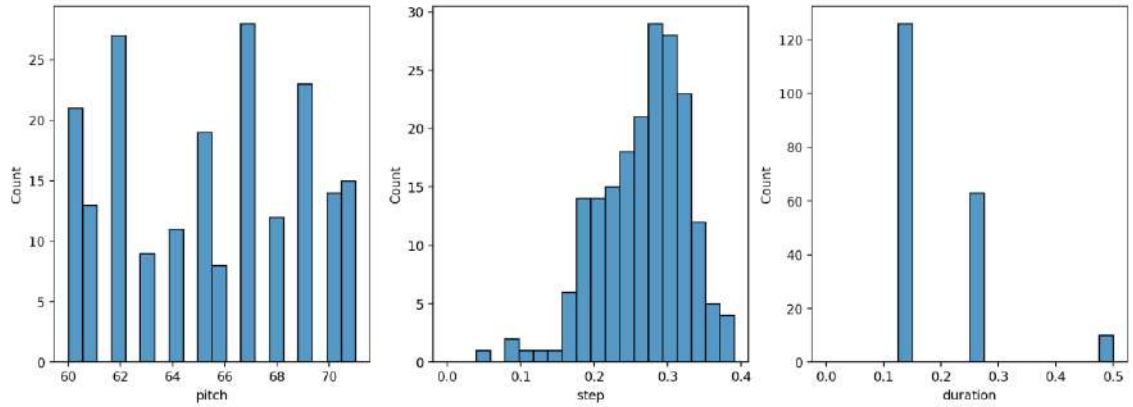


Figura 52. Partitura del audio generado mediante el modelo LSTM-GRU y posteriormente ajustado para centralizar las notas en una octava media.

Para visualizar de forma más clara la disposición de las notas generadas y la variabilidad en el tono, la duración y el intervalo, la Figura 53 presenta un *piano roll* del audio ajustado (*music_lstm_gru_tuned*) (a.), acompañado de una gráfica que muestra la distribución de los valores de estas tres características (b.). En este *piano roll*, las notas presentan una distribución uniforme, lo que concuerda con la partitura generada. Además, en la gráfica de *duration* se observan valores cortos, alrededor del 0,3, al igual que en la de *step*, lo que indica que los intervalos entre notas son breves. Esto sugiere que la pieza se compone de figuras musicales de menor duración y, por tanto, tiene un ritmo más veloz.



a. *Piano roll del audio generado.*



b. *Distribución de las variables tono, duración e intervalo de las notas generadas.*

Figura 53. Piano roll y gráficas con la distribución de las variables tono, duración e intervalo de las notas generadas mediante el modelo LSTM-GRU.

A continuación, se analizan los resultados del modelo *LSTM-GRU* considerando los mismos tres enfoques aplicados en los modelos anteriores: evolución del entrenamiento, métricas musicales cuantitativas y evaluación subjetiva a través de la escucha activa.

En primer lugar, se examina el proceso de entrenamiento mediante la evolución de la función de pérdida. La Figura 54 muestra la evolución durante 150 épocas de la pérdida total para el conjunto de entrenamiento.

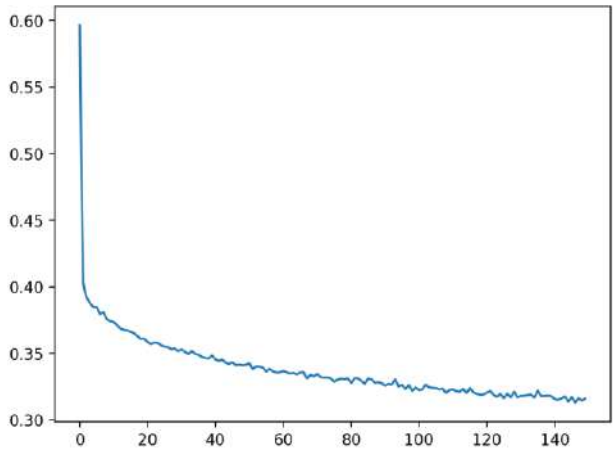


Figura 54. Gráfica de la evolución durante las 150 épocas de la función de pérdida total del conjunto de entrenamiento en el modelo LSTM-GRU.

El entrenamiento se interrumpió de forma preventiva en la época 150, obteniendo un valor de pérdida en torno a 0,30 sobre los datos de entrenamiento. Aunque no se evidencian signos claros de sobreajuste, esta parada temprana se realiza con el fin de preservar la capacidad del modelo para generalizar frente a nuevas entradas.

En relación con a las cinco métricas cuantitativas para evaluar la calidad del audio generado y ajustado (*music_lstm_gru_tuned*), la Figura 55 muestra los resultados obtenidos del audio más representativo entre los 15 generados sintéticamente.

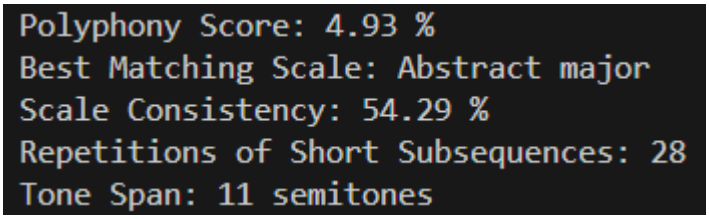


Figura 55. Resultados de las métricas de evaluación objetiva sobre el audio generado mediante el modelo LSTM-GRU.

La primera métrica refleja el grado de polifonía presente en la pieza generada con un valor del 4,93 %, lo que indica una escasa presencia de notas simultáneas. En cuanto a la escala más adecuada al contenido musical, se identificó una tonalidad cercana a la *abstract major*, asociada comúnmente a emociones positivas y un carácter enérgico. La tercera métrica evalúa la proporción del audio que se ajusta efectivamente a esta escala. Por su parte, la siguiente medida analiza la variedad de secuencias musicales distintas dentro de la pieza, y la última se enfoca en la amplitud tonal explorada. En conjunto, estos resultados indican que la composición evita la repetición excesiva y demuestra una notable diversidad melódica, optando por una exploración creativa en lugar de mantener estructuras musicales rígidas. En cuanto a los demás audios generados, los resultados guardan similitudes notables con el análisis anterior, aunque con ciertas diferencias. Por ejemplo, la polifonía se incrementa ligeramente en la mayoría de los casos, manteniendo ese carácter donde las notas destacan de forma individual. Al igual que en la pieza descrita, *abstract major* sigue siendo la escala predominante, lo que refuerza la consistencia en la creación de atmósferas vibrantes y emocionalmente positivas. No obstante, en dos audios se detectan incursiones puntuales en escalas modales, introduciendo un contraste sutil sin alterar el tono general. Respecto a la diversidad estructural, todas las composiciones evitan caer en repeticiones literales, priorizando secuencias melódicas renovadas en cada sección. Este enfoque, combinado con una distribución equilibrada de las frases musicales, genera una sensación de evolución constante, incluso cuando el esquema rítmico base se mantiene estable.

En cuanto a la evaluación subjetiva musical a partir de la escucha activa del audio representativo generado sintéticamente y ajustado (*music_lstm_gru_tuned*), la pieza presenta un ritmo dinámico y alegre desde el inicio, con una velocidad que transmite ligereza y un aire lúdico. Esta constancia rítmica aporta fluidez y mantiene el interés sin generar tensión o rigidez. La melodía inicial construye frases coherentes, con una armonía bien estructurada que resulta agradable al oído y transmite dinamismo. Sin embargo, hacia la mitad de la pieza, el desarrollo armónico se estanca en una progresión única, limitándose a variaciones menores en las notas sin evolucionar la base tonal. Esto genera cierta monotonía, ya que la repetición del mismo patrón

melódico modificado ligeramente reduce el impacto emocional inicial y crea un efecto predecible. En el plano técnico, el modelo LSTM-GRU demuestra capacidad para capturar patrones rítmicos y melódicos básicos del audio de entrenamiento, logrando una pieza breve, ágil y estructuralmente sólida en su primera mitad. No obstante, la limitación en la complejidad armónica en fases avanzadas sugiere dificultades del modelo para manejar desarrollos prolongados. A pesar de ello, el resultado global mantiene una agradable calidad auditiva. Con respecto a los demás audios generados del conjunto de pruebas finales, los resultados reflejan patrones análogos al caso analizado. La mayoría comparte una energía rítmica vibrante desde los primeros compases, lo que refuerza la sensación de dinamismo. Las melodías iniciales suelen presentar frases bien estructuradas, con armonías claras y progresiones que capturan rápidamente la atención del oyente. Sin embargo, también replican la limitación en el desarrollo armónico. Hacia la segunda mitad de estas piezas, las variaciones melódicas se reducen significativamente, priorizando repeticiones con ajustes mínimos en lugar de explorar nuevas direcciones tonales. Por último, un detalle recurrente es la tendencia a mantener una base rítmica sólida pero estática, que actúa como columna vertebral mientras la capa melódica intenta introducir novedades.

5.8 Preprocesado de datos para el modelo Wavenet

La arquitectura de *Wavenet* requiere un preprocesamiento de datos adaptado a sus particularidades técnicas. La primera etapa implica reunir 62176 eventos musicales (notas y acordes) extraídos de 30 archivos de audio incluidos en el *Maestro Dataset*. Al tratarse de archivos MIDI, cada evento registra ciertas características: la altura tonal (*pitch*), la duración, la intensidad de ejecución (*velocity*) y su ubicación exacta dentro de la melodía (*offset*). Este formato también permite codificar acordes en forma de triadas, manteniendo los mismos parámetros mencionados.

Para reducir la carga computacional y evitar sobrecargar el modelo, se priorizan los eventos más recurrentes. Concretamente, se identifican los 201 eventos (notas o acordes) que aparecen al menos 50 veces en el conjunto total. Tras este filtrado, se conservan 54893 eventos de los 62176 eventos originales, descartándose los eventos menos representativos.

Posteriormente, se construyen secuencias de entrenamiento utilizando una ventana deslizante. Cada secuencia de entrada (X) está compuesta por 32 eventos consecutivos, y la etiqueta de salida (Y) corresponde al evento inmediatamente siguiente en la secuencia (posición 33). A partir de las 54893 notas filtradas, este procedimiento produce 53933 notas totales creando 1686 secuencias entrada-salida (X-Y). Cada evento se asigna a un número entero entre 0 y 200, lo que transforma el problema en un problema de clasificación multiclase entre las 201 categorías posibles.

Los 1686 pares de datos se dividen en tres grupos: el 70 % (1180 secuencias) se destina a entrenamiento, el 20 % (337 secuencias) a validación y el 10 % restante (169 secuencias) a pruebas finales. Esta separación garantiza que el modelo se evalúe con secuencias totalmente desconocidas, asegurando que no haya memorizado patrones durante el entrenamiento. Finalmente, el conjunto de pruebas finales se utiliza para generar 15 melodías sintéticas, seleccionando la que mejor refleja la coherencia armónica y rítmica aprendida.

5.9 Implementación del modelo Wavenet

El modelo *WaveNet* implementado se presenta en la Figura 56. Su arquitectura se compone de una capa inicial de *embedding* que procesa una secuencia de entrada de 32 notas musicales, donde cada una se codifica originalmente mediante un valor entero en el rango de 0 a 200. Esta capa tiene como objetivo convertir estos índices discretos en vectores continuos de dimensionalidad (32, 100). Dicha conversión facilita que el sistema identifique conexiones semánticas y estructurales entre las notas dentro de un espacio multidimensional, evitando el tratamiento de cada elemento como una entidad independiente y desvinculada. La elección de una dimensión de *embedding* de 100 se definió tras análisis experimentales propios llevados a cabo durante la fase de optimización del modelo, donde se contrastaron múltiples tamaños de representación. En estos ensayos, configuraciones reducidas como 32 o 64 evidenciaron limitaciones en la captura de matices musicales, mientras que opciones más amplias (128 o superiores) añadieron carga computacional excesiva sin aportar ganancias notables en la generación de audios.

Las siguientes cuatro capas del modelo son convoluciones dilatadas causales, con factores de dilatación crecientes (1, 2, 4 y 8). Este esquema escalonado permite al modelo capturar dependencias a diferentes escalas temporales dentro de la secuencia musical, extendiendo progresivamente su campo receptivo a relaciones más largas entre las notas. Cada capa utiliza filtros de tipo *Conv1D* (Convolución unidimensional), una operación que aplica filtros deslizantes a lo largo de la dimensión temporal de la secuencia. En este contexto, los filtros extraen características locales, como patrones melódicos o rítmicos, detectando relaciones entre eventos musicales contiguos. La densidad de los filtros *Conv1D* se incrementa en cada capa (64, 128, 256 y 512 filtros respectivamente), permitiendo que el modelo capture patrones cada vez más complejos. A cada operación de convolución le sigue la aplicación de una función de activación *ReLU*.

Posteriormente, tras cada operación convolucional, se introduce una capa de *MaxPool1D*, que realiza una reducción en la dimensión temporal. Esta operación selecciona el valor máximo dentro de pequeñas ventanas consecutivas, conservando así las características más destacadas y descartando información redundante. El efecto combinado de la convolución y el *max pooling* permite construir representaciones jerárquicas, en las que cada capa acumula progresivamente información de contextos temporales más amplios.

Después de la última convolución, se aplica una operación de *GlobalMaxPool1D*, que sintetiza toda la información temporal restante en un solo vector. Esta operación selecciona el valor máximo para cada filtro en toda la dimensión temporal, reteniendo las características más fuertes detectadas en cualquier punto de la secuencia, lo cual genera un resumen compacto y robusto de la entrada. El vector resultante alimenta una capa densa de 512 neuronas con activación *ReLU*.

Finalmente, el modelo predice la siguiente nota mediante una capa densa de 201 neuronas con función de activación *Softmax*. Esta capa genera un vector de 201 probabilidades, cada una correspondiente a una de las posibles notas predichas. Cabe destacar que, en este modelo, tono, duración e intervalo no se tratan como atributos independientes, sino que cada evento musical (nota o acorde) se codifica como una única unidad indivisible.

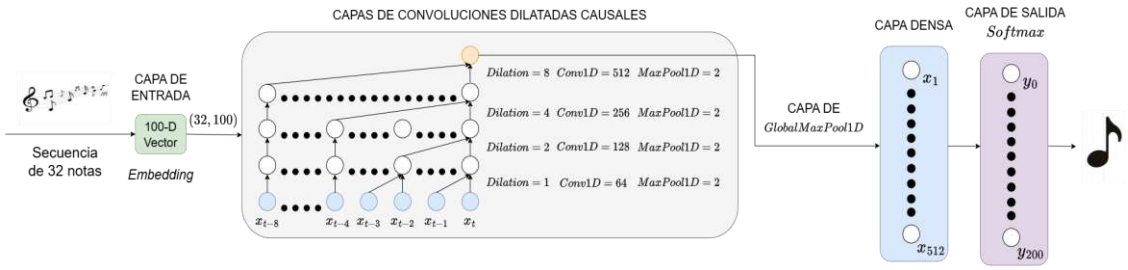


Figura 56. Estructura interna del modelo Wavenet implementado.

Se emplea la función de pérdida categórica *sparse categorical crossentropy* (5.1). Esta función es particularmente adecuada cuando el problema se plantea como una clasificación multiclase y las etiquetas verdaderas están codificadas como enteros. En este caso, la tarea del modelo consiste en predecir la nota musical siguiente, eligiendo entre un conjunto de 201 posibles notas discretas, cada una representada por un número entero entre 0 y 200.

Respecto a los hiperparámetros utilizados en este modelo, se establece un tamaño de lote (*mini-batch*) de 512 para el entrenamiento del modelo, después de varias pruebas que mostraron que este valor ofrece los mejores resultados. Se aplica una tasa de aprendizaje inicial de 0,001, que decrece de manera exponencial a lo largo del entrenamiento utilizando el esquema *ExponentialDecay* de *Keras*. Con este método, la tasa de aprendizaje disminuye según la fórmula:

$$\alpha_{\text{nueva}} = \alpha_{\text{inicial}} \times e^{-\text{decay\_rate} \times \text{step}} \tag{5.5}$$

donde α_{inicial} es la tasa inicial (0,001), *decay_rate* (0,97) es el coeficiente que controla la velocidad de decrecimiento, y *step* (100000) representa el número máximo de pasos en el entrenamiento en los que se aplica. Esta estrategia permite que el modelo realice ajustes grandes en las fases iniciales del entrenamiento y refinamientos progresivamente más pequeños conforme avanza, facilitando así una convergencia más estable hacia mínimos más precisos.

Para evitar el sobreajuste, el modelo implementa tres mecanismos de regularización simultáneos en todas sus capas: *dropout* (0,3), *batch normalization*, y *kernel regularizer L2* (0,0001). Adicionalmente, se emplea *early stopping* como método de parada temprana del entrenamiento. El optimizador escogido es *RMSprop*.

El entrenamiento de la red neuronal se desarrolla a lo largo de 500 épocas, durante las cuales el modelo aprende las características musicales de la secuencia de entrada para predecir y generar la siguiente nota.

Finalmente, el modelo genera una secuencia de 200 notas musicales. Para ello, se selecciona una nota aleatoria perteneciente al conjunto de pruebas finales como punto de partida y, a partir de ella, se construye una secuencia inicial de 32 notas. En cada predicción, se descarta la primera nota de la secuencia y se añade la nueva nota generada en la última posición, como se puede observar en el proceso iterativo mostrado en la Figura 21.

La generación de las 200 notas que conforman las melodías sintéticas se realiza utilizando tres enfoques distintos: seleccionar siempre la nota con mayor probabilidad en la capa de salida *Softmax*, utilizar un método de selección estocástico y suavizar el vector de probabilidades de la capa de salida.

5.9.1 Resultados del modelo Wavenet

Los resultados obtenidos con el modelo *Wavenet* se analizan considerando los tres enfoques utilizados en los modelos anteriores: evolución del entrenamiento, métricas musicales cuantitativas y evaluación subjetiva a través de la escucha activa.

En primer lugar, comúnmente para las tres formas de generación de las 200 notas sintéticas, se examina el proceso de entrenamiento del modelo mediante la evolución de la función de pérdida y la métrica acierto (*accuracy*) para los conjuntos de entrenamiento y validación. La Figura 57 muestra los resultados.

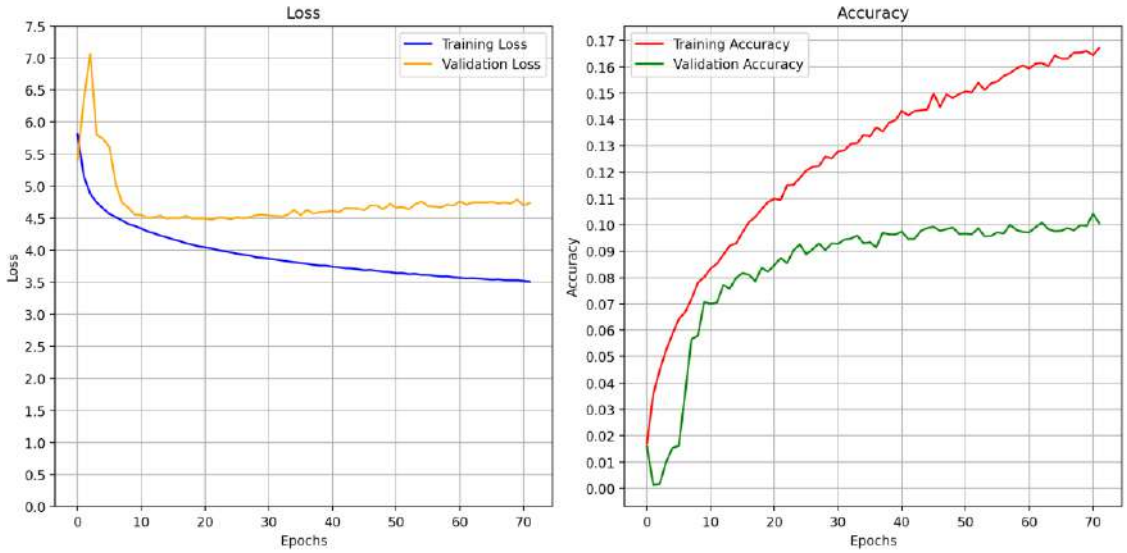


Figura 57. Gráficas de la evolución durante las 72 épocas de la función de pérdida total del conjunto de entrenamiento (izquierda) y validación junto con el accuracy (derecha) en el modelo Wavenet.

El entrenamiento se detiene en la época 72 para prevenir el posible sobreajuste que puede desarrollarse durante las siguientes épocas, ya que la pérdida del conjunto de entrenamiento (en azul) continúa decreciendo mientras la correspondiente al conjunto de validación (en verde) crece. Los valores de *accuracy* obtenidos se sitúan en torno al 17 % para el conjunto de entrenamiento y al 10 % para el de validación.

En el caso de seleccionar siempre la nota con mayor probabilidad en la capa de salida *Softmax*, el audio completo más representativo, de entre el total de 15 generados, se puede escuchar en el siguiente enlace:

music_wavenet_max

La Figura 58 muestra la partitura del audio anterior. Como se puede apreciar, el modelo predice la misma secuencia de notas durante gran parte del audio, con ligeras variaciones al inicio. Además, toda la composición se construye exclusivamente a partir de acordes, lo que define su estructura y carácter sonoro.

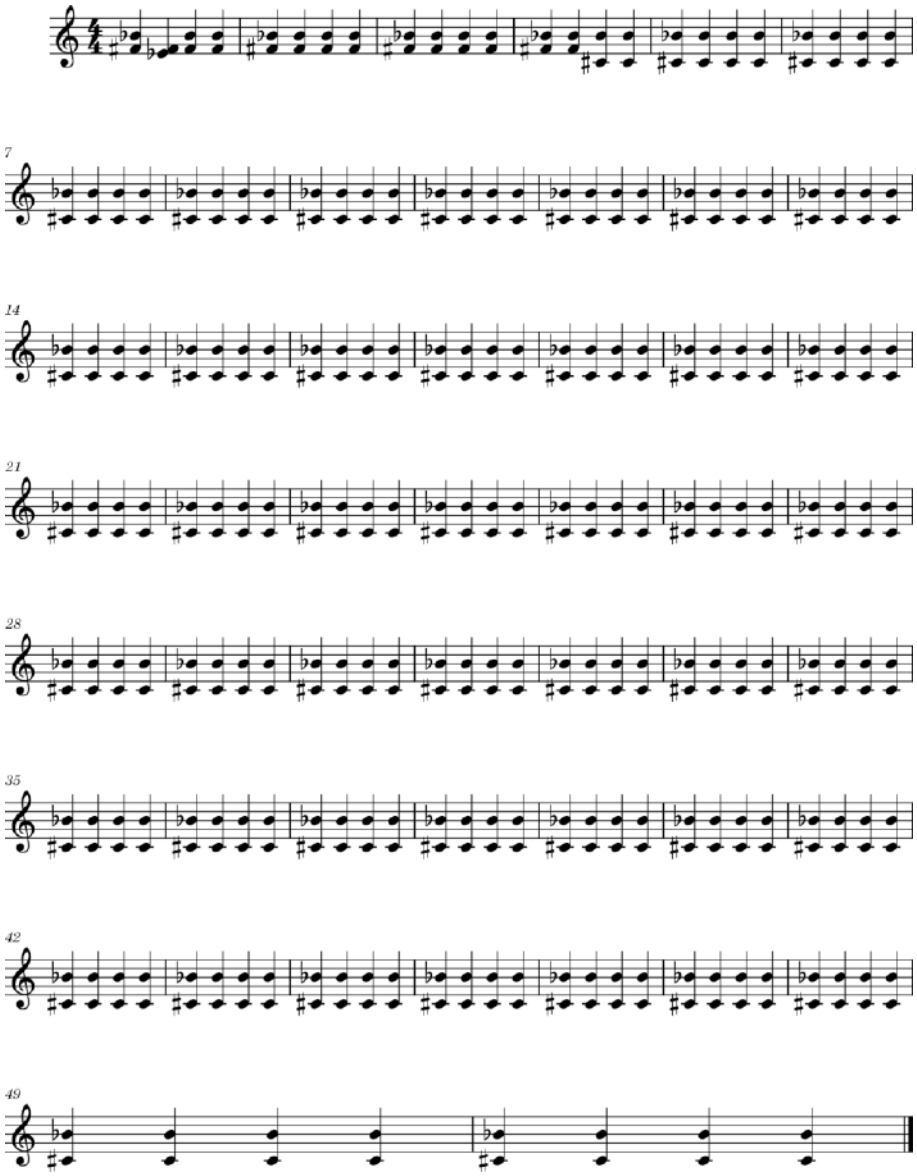


Figura 58. Partitura del audio generado más representativo mediante el modelo Wavenet en el caso de mayor probabilidad.

En relación con el análisis objetivo, se evalúan cinco métricas cuantitativas que miden la calidad del audio generado en el caso de seleccionar siempre la nota con mayor probabilidad en la capa de salida (*music_wavenet_max*). Este audio se ha seleccionado como el más representativo, de un total de 15 audios generados sintéticamente, por su equilibrio entre creatividad y respeto a los patrones entrenados. Las métricas estudiadas son: polifonía, escala que mejor se adecua, consistencia, subsecuencias musicales únicas y rango tonal. La Figura 59 muestra los resultados obtenidos.

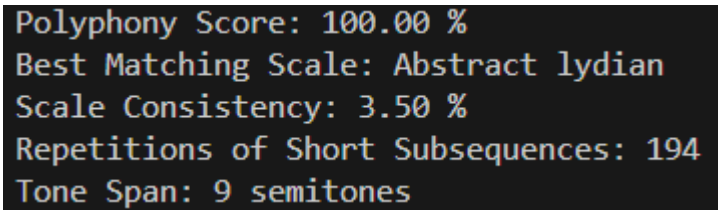


Figura 59. Resultados de las métricas de evaluación objetiva sobre el audio generado más representativo mediante el modelo Wavenet en el caso de seleccionar siempre la nota con mayor probabilidad en la capa de salida.

Se observa un nivel de polifonía del 100 %, lo que indica que en todo momento suenan múltiples notas simultáneamente, componiendo la totalidad del audio a partir de acordes. En cuanto a la escala que mejor se ajusta a la composición, se identifica la *abstract lydian*, asociada comúnmente a un carácter alegre. También se evalúa la proporción del audio que se mantiene dentro de esta escala El resultado del 3,5 % indica que la composición no sigue de manera consistente las notas propias de esta escala, lo que sugiere que la pieza busca explorar otros motivos musicales que aportan mayor libertad tonal a la melodía. En relación con la variedad de secuencias musicales presentes en la pieza y el rango tonal explorado, los resultados revelan una clara tendencia hacia la repetición, con predominio de estructuras melódicas recurrentes por encima de la diversidad creativa. Los resultados obtenidos con el resto de los audios generados sintéticamente son similares a la pieza analizada. En todos los casos, la polifonía alcanza valores próximos al 100 %, confirmando que el modelo prioriza sistemáticamente el uso de acordes sobre melodías. Respecto a la escala, *abstract lydian* domina en todas las composiciones, reforzando un carácter luminoso y enérgico que se alinea con el tono alegre descrito previamente. En cuanto a la diversidad estructural, la crítica central se repite: la mayoría de las piezas exhiben una repetición excesiva de patrones, con secuencias musicales que recurren a variaciones mínimas de los mismos motivos de acordes. Este enfoque, aunque garantiza cohesión, reduce la capacidad narrativa de las piezas, generando una sensación repetitiva y estancamiento creativo.

En cuanto a la evaluación subjetiva del audio generado sintéticamente en el caso de seleccionar siempre la nota con mayor probabilidad en la capa de salida (*music_wavenet_max*), la pieza establece una base armónica consistente en el inicio, con acordes que se enlazan de manera fluida y una transición limpia entre triadas. Esta cohesión inicial contribuye a una sensación de equilibrio y dirección musical clara. No obstante, el desarrollo inmediatamente posterior evidencia una recurrencia excesiva de los mismos patrones armónicos sin variaciones sustanciales. Si bien la repetición controlada puede funcionar como recurso expresivo, en este caso su persistencia sin contraste suficiente genera una progresión estática que reduce gradualmente el interés auditivo. Por tanto, el resultado global de la pieza insiste en patrones rítmicos y armónicos idénticos, lo que genera una sensación de repetición monótona que empobrece la experiencia auditiva prolongada. En cuanto a los demás audios generados, los resultados reflejan un comportamiento similar al descrito en la pieza representativa. La cohesión armónica inicial con acordes bien definidos se mantiene como un rasgo constante en todas las composiciones, sugiriendo que el modelo ha internalizado eficazmente los principios básicos de enlace entre triadas. Sin embargo, esta virtud inicial se ve opacada por una tendencia recurrente a reiterar progresiones idénticas sin introducir variaciones significativas. En el ámbito rítmico, se observa el mismo problema: una base

sólida y bien articulada al inicio, que gradualmente deriva en repeticiones mecánicas. Respecto a la dinámica, la mayoría de las piezas comparten un arco similar: un inicio prometedor que evoca claridad y equilibrio, seguido de un estancamiento progresivo donde la falta de contraste melódico empobrece la experiencia auditiva.

El método de selección estocástico de la siguiente nota generada sintéticamente consiste, en primer lugar, en dividir el intervalo [0,1] en 201 segmentos, correspondientes a las notas a predecir, de forma proporcional a las probabilidades calculadas por la capa de salida *Softmax* del modelo neuronal. Posteriormente, se selecciona aquella nota (segmento) que contiene el valor generado de forma uniformemente aleatoria mediante la función *np.random.choice*.

La elección estocástica introduce diversidad controlada en las predicciones, tratando de evitando evitar así los patrones repetitivos de la selección de la nota con mayor probabilidad y representando la incertidumbre propia de los dominios creativos como es la generación musical. Al iterar este proceso en un bucle, donde cada predicción generada se incorpora al contexto para generar la siguiente nota, se construyen secuencias que preservan la coherencia musical pero integran variaciones.

En este caso de seleccionar la siguiente nota de forma estocástica, el audio completo más representativo, se puede escuchar en el siguiente enlace:

music_wavenet_stoch

La Figura 60 muestra la partitura del audio anterior *music_wavenet_stoch*. En este caso, el modelo predice una secuencia de notas diferentes que forman una melodía musical. En ocasiones, introduce notas muy graves que no alteran la fluidez de la pieza, y emplea acordes de manera constante a lo largo de la composición.

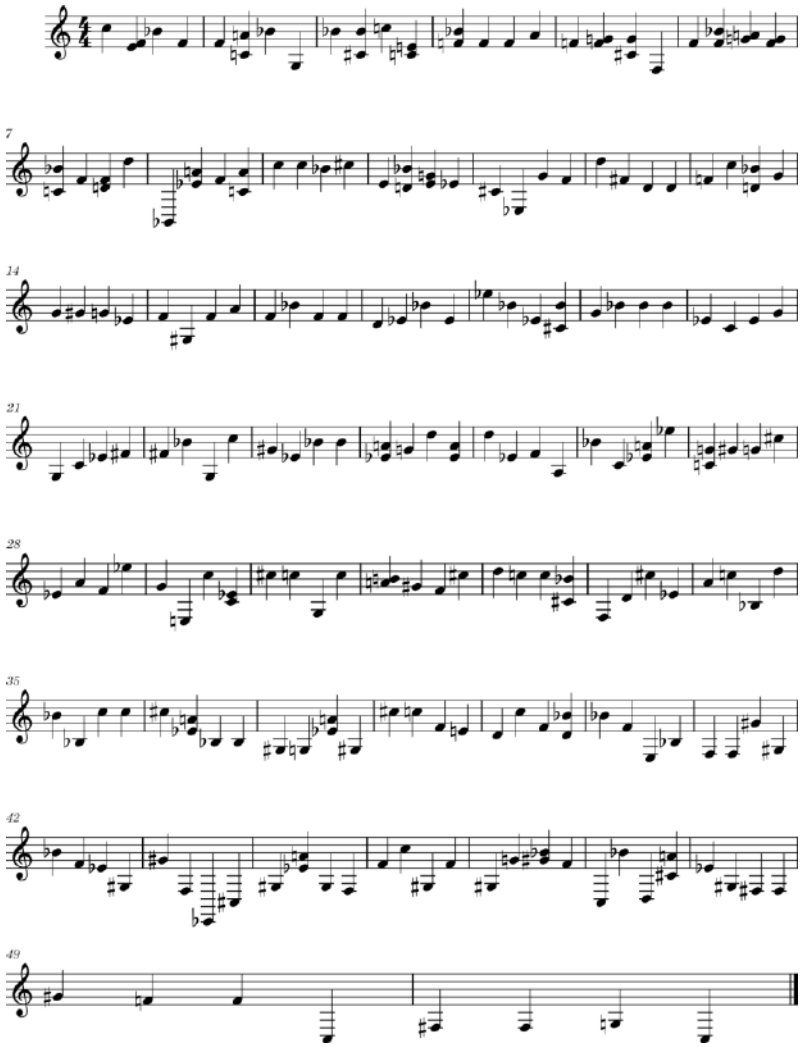


Figura 60. Partitura del audio generado más representativo mediante el modelo Wavenet en el caso de distribución de probabilidad.

En cuanto al análisis objetivo, se examina la calidad del audio generado (*music_wavenet_stoch*), seleccionado como el más representativo entre un total de 15 audios generados sintéticamente, mediante cinco métricas cuantitativas. La Figura 61 recoge los resultados.

```
Polyphony Score: 15.50 %
Best Matching Scale: Abstract dorian
Scale Consistency: 79.65 %
Repetitions of Short Subsequences: 6
Tone Span: 36 semitones
```

Figura 61. Resultados de las métricas de evaluación objetiva sobre el audio generado más representativo mediante el modelo Wavenet en el caso de distribución de probabilidad.

La primera métrica indica la puntuación en polifonía que alcanza esta pieza musical. El porcentaje de 15,50 % indica que en ciertas ocasiones se observan acordes sonando. La segunda métrica trata sobre el tipo de escala que mejor se ajusta al audio. El resultado de *abstract dorian* sugiere que la pieza tiene un tono melancólico, pero combinado con vivacidad. La tercera métrica revela que 79,65 % del audio se alinea con la escala *abstract dorian*, lo que señala que

la composición sigue mayormente su estructura tonal. La siguiente métrica examina la cantidad de secuencias musicales distintas en la composición, mientras que la última mide el rango tonal de la pieza. En este caso, el análisis muestra que la música generada tiene una mayor diversidad, introduciendo nuevas frases musicales sin depender de un patrón fijo. En cuanto a los demás audios generados, los resultados mantienen un perfil semejante al descrito, aunque con variaciones sutiles que refuerzan las tendencias observadas. La polifonía, se mantiene ligeramente igual en la mayoría de las composiciones, confirmando que el modelo evita saturar las piezas con acordes densos, pero incorpora momentos estratégicos de armonización para añadir profundidad. Este enfoque equilibrado se alinea con el carácter híbrido de la escala *abstract dorian*, que combina melancolía sutil con pulsos rítmicos vibrantes. Respecto a la tonalidad, la gran mayoría de los audios analizados restantes priorizan esta escala modal *abstract dorian*, lo que refuerza la dualidad emocional (nostalgia con toques de energía) identificada en la pieza representativa. Sobre la diversidad estructural, todas las piezas comparten la misma filosofía: evitan depender de un motivo fijo, explorando en su lugar frases melódicas independientes que se entrelazan de manera no lineal. Este enfoque, aunque arriesgado, genera una sensación de exploración continua, aunque en algunos casos deriva en falta de cohesión temática.

En relación con la valoración subjetiva a través de una escucha activa, la pieza más representativa (*music_wavenet_stoch*) inicia con una propuesta melódica coherente, donde se presenta una leve discordancia armónica en los primeros compases que se resuelve con agilidad mediante una transición efectiva, permitiendo un desarrollo natural del discurso musical. El empleo de frases contrastantes en secciones posteriores introduce variaciones temáticas que aportan dinamismo y evitan la sensación de estancamiento, demostrando una búsqueda consciente de diversidad en los patrones rítmicos y armónicos. El ritmo, aunque mantiene un pulso constante y moderado, logra transmitir serenidad sin caer en la monotonía. En el plano armónico, los enlaces entre acordes se ejecutan con precisión técnica, generando un flujo que contribuye a la vitalidad general de la composición. Estos contrastes interválicos, particularmente en las secciones medias, añaden capas de interés al introducir tensiones resueltas de manera satisfactoria. La estructura global demuestra solidez compositiva, con una progresión de notas que culmina en un cierre tonalmente consistente. Si bien la obra podría beneficiarse de mayores contrastes dinámicos, el resultado actual presenta un equilibrio notable entre intención expresiva y corrección técnica, logrando mantener la atención del oyente hasta su conclusión. En relación con los demás audios generados del conjunto de pruebas finales, los resultados reflejan un enfoque compositivo altamente coherente con la pieza analizada. La identificación de disonancias iniciales actúa como sello distintivo del modelo, demostrando su capacidad para convertir tensiones armónicas en transiciones fluidas y narrativamente satisfactorias. Al igual que en *music_wavenet_stoch*, la mayoría de las piezas inician con una propuesta melódica estructurada, donde pequeñas fricciones tonales se integran como recursos expresivos, no como errores, para luego resolverse en progresiones naturales que guían al oyente hacia una escucha agradable. En cuanto a la diversidad temática, el uso de frases contrastantes se mantiene. Estas variaciones en las frases musicales comparten el mismo propósito: evitar la repetición mediante cambios que añaden profundidad sin fragmentar el discurso musical. El ritmo, por su parte, mantiene el equilibrio entre constancia y expresividad observado en la pieza representativa. En el ámbito armónico, la precisión técnica en los enlaces entre acordes es notable.

Aunque las elecciones armónicas varían, todas comparten un mismo principio: mantener un flujo coherente que sostenga la atención sin sobresaturar. Respecto a la estructura global, todas las composiciones culminan en cierres tonales consistentes.

El método de selección de la siguiente nota generada sintéticamente basado en el suavizado del vector de probabilidades de la capa de salida consiste en dividir este vector de probabilidades por un hiperparámetro denominado *temperature*. Posteriormente, sobre el vector resultante, se aplica el método de selección estocástico descrito para elegir la siguiente nota musical de la pieza generada sintéticamente por el modelo neuronal. Después de varias pruebas, se elige el valor del hiperparámetro *temperature* = 2. Este valor permite que, además de las notas más probables, tengan oportunidad de seleccionarse otras con menor probabilidad. De este modo, se fomenta la exploración de combinaciones musicales menos predecibles y se enriquece la diversidad melódica de la pieza sin abandonar su coherencia musical.

En este caso de seleccionar la siguiente nota mediante el hiperparámetro *temperature*, el audio completo más representativo, se puede escuchar en el siguiente enlace:

music_wavenet_temp

La Figura 62 muestra la partitura del audio anterior *music_wavenet_temp*. Como se puede observar, la partitura revela que el modelo predice una secuencia de notas distintas que, al integrarse, conforman una melodía musical. En ocasiones, introduce notas muy graves que aportan profundidad sin afectar la fluidez de la pieza al igual que las notas muy agudas. Además, se evidencia una mayor presencia de acordes a lo largo de toda la composición.

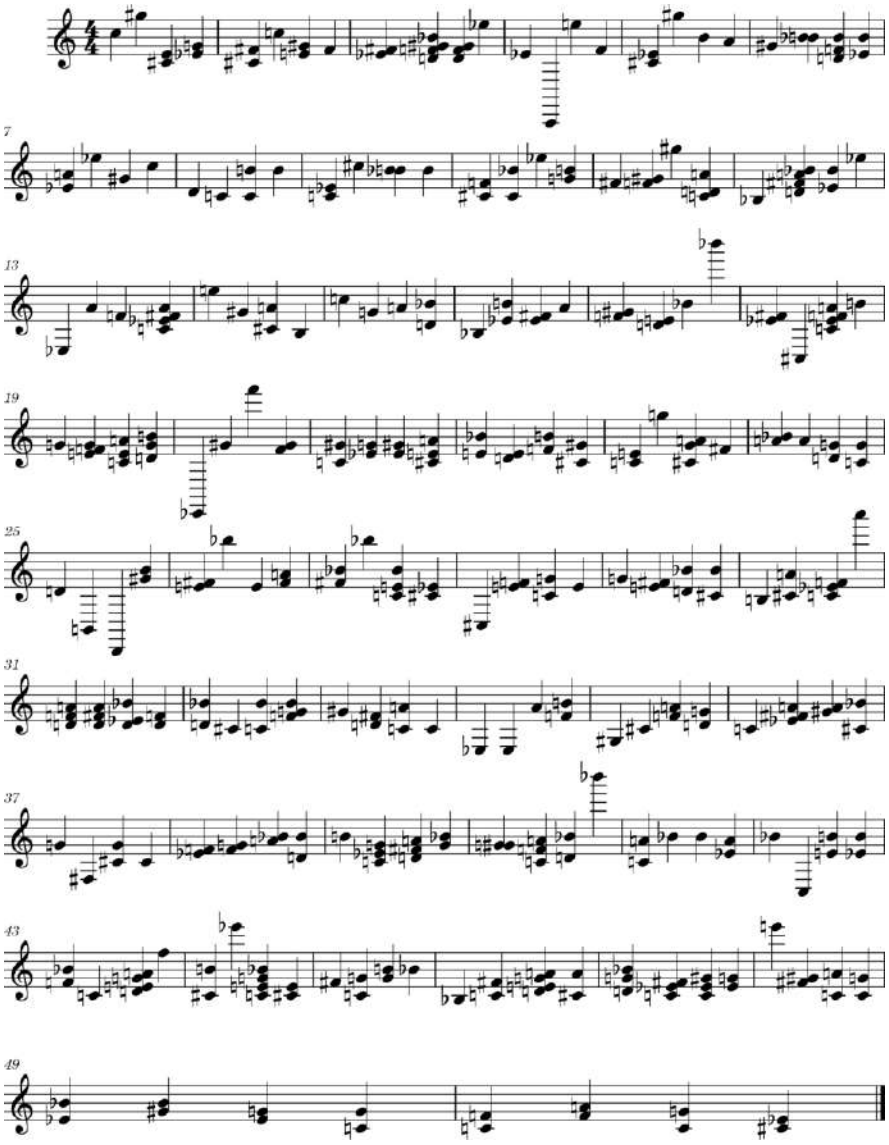


Figura 62. Partitura del audio generado más representativo mediante el modelo Wavenet en el caso de distribución de probabilidad con temperature.

La evaluación objetiva mide la calidad musical de la pieza más representativa (*music_wavenet_temp*) entre un total de 15 audios generados sintéticamente mediante cinco métricas cuantitativas. Las métricas consideradas abarcan aspectos como la presencia de polifonía, la escala más adecuada al contenido musical, la coherencia interna de la pieza, la diversidad de secuencias melódicas y la amplitud del rango tonal. La Figura 63 muestra los resultados obtenidos.

```
Polyphony Score: 58.00 %
Best Matching Scale: Abstract dorian
Scale Consistency: 59.26 %
Repetitions of Short Subsequences: 0
Tone Span: 58 semitones
```

Figura 63. Resultados de las métricas de evaluación objetiva sobre el audio generado más representativo mediante el modelo Wavenet en el caso de distribución de probabilidad con temperature.

Se observa un nivel de polifonía del 58 %, lo que indica la presencia frecuente de múltiples notas simultáneas, componiendo mayoritariamente la obra a través de acordes. En cuanto a la tonalidad predominante, el análisis identifica la escala *abstract dorian*, que sugiere un carácter melancólico pero fluido. También se calcula el porcentaje del audio generado que corresponde a dicha escala. La métrica muestra que 59,26 % del audio se enmarca en la escala *abstract dorian*, lo que refleja una adherencia parcial a su estructura tonal. Esto sugiere un equilibrio entre mantenerse dentro de la escala y explorar nuevos motivos musicales fuera de ella. Además, se evalúa la diversidad de secuencias musicales presentes en la composición, así como la amplitud del rango tonal explorado. En este caso, los resultados revelan una marcada riqueza estructural, reflejada en la incorporación de frases musicales variadas que evitan la rigidez de patrones repetitivos. En general, los audios restantes generados del conjunto de pruebas finales reflejan un patrón similar con la pieza analizada. La polifonía elevada se mantiene como una constante, confirmando que el modelo prioriza texturas sonoras basadas en acordes. Esta elección, aunque limita la claridad melódica individual, aporta una riqueza armónica que refuerza el carácter fluido y melancólico asociado a la escala *abstract dorian*. Respecto a la tonalidad, las piezas restantes se adhieren a esta misma escalar anterior, consolidando esa dualidad entre melancolía sutil y fluidez rítmica. En cuanto a la diversidad estructural, todos los audios comparten la misma filosofía: evitan esquemas repetitivos mediante frases novedosas interconectadas. Este enfoque, aunque técnicamente ambicioso, se ejecuta con coherencia, demostrando que el modelo puede gestionar transiciones melódicas sin perder el hilo musical.

En relación con la crítica subjetiva musical a través de una escucha activa, la pieza más representativa (*music_wavenet_temp*) comienza con una progresión armónica que establece un carácter introspectivo, utilizando acordes de cualidad oscura que imprimen un matiz reflexivo desde los primeros compases. Este fundamento tonal se mantiene con coherencia a lo largo del desarrollo, reforzando la unidad temática mediante variaciones que orbitan en torno a un núcleo melódico común. La elección de enlaces entre acordes, aunque técnicamente funcional, incorpora ocasionalmente tensiones no resueltas de forma inmediata, generando breves discordancias que contrastan con el flujo general. El ritmo, de pulsación constante, actúa como eje estructural que potencia la atmósfera de misterio sin caer en la rigidez métrica. Por último, a pesar de algunas disonancias puntuales que interrumpen brevemente la inmersión auditiva, la pieza logra reintegrarlas con habilidad, retomando su línea tonal principal hasta el final. De manera similar, los audios restantes reproducen el mismo enfoque compositivo, consolidando un sello característico en la generación de atmósferas introspectivas. La base armónica oscura y reflexiva se mantiene como eje central en todas las piezas, actuando como hilo conductor emocional. En cuanto a las tensiones no resueltas, estas aparecen como un recurso recurrente, aunque con distinto grado de intensidad. Las disonancias puntuales, lejos de ser errores, funcionan como marcadores musicales. Estos momentos, aunque breves, añaden profundidad al sugerir conflictos internos dentro de la propia estructura tonal, un recurso que humaniza la composición al simular las contradicciones emocionales que se encuentran en obras creadas por humanos. En el desarrollo melódico, todas las piezas orbitan en torno a un núcleo temático común, pero evitan repeticiones literales. Este enfoque refuerza la unidad sin sacrificar la evolución, aunque en algunos casos la transición resulta demasiado abrupta, revelando limitaciones en la gestión de estos cambios melódicos.

5.10 Comparación con otros modelos generativos

Los resultados obtenidos a través de los modelos implementados se comparan con otras soluciones existentes, como *MuseNet*, *C-RNN-GAN* y *GanSynth*, en la tarea de la generación sintética de piezas musicales a piano.

5.10.1 MuseNet

El primer modelo analizado en este estudio es *MuseNet*, desarrollado por *OpenAI*. *MuseNet* utiliza la arquitectura de *transformers* para la generación sintética de música. Su funcionamiento se basa en un mecanismo de atención a largo plazo, que permite capturar y procesar las relaciones entre las notas a lo largo de toda la composición. De esta manera, el modelo genera piezas polifónicas al combinar varias líneas melódicas y voces, considerando tanto los patrones musicales locales como la estructura global de la pieza.

Para la valoración objetiva y subjetiva musical, se ha elegido la pieza generada más representativa (*music_musenet*) procedente del repositorio público *SoundCloud*, la cual se alinea con el propósito central de este trabajo: la generación de piezas musicales a piano, sin intervención de otros instrumentos o voces humanas. La pieza elegida se puede escuchar en el siguiente enlace:

music_musenet

La evaluación objetiva de la pieza musical en base a las cinco métricas cuantitativas que miden la calidad del audio generado (*music_musenet*) obtiene los resultados mostrados en la Figura 64.

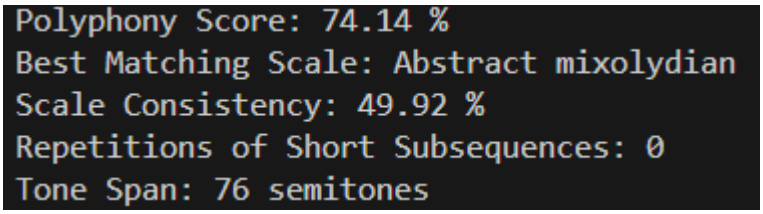


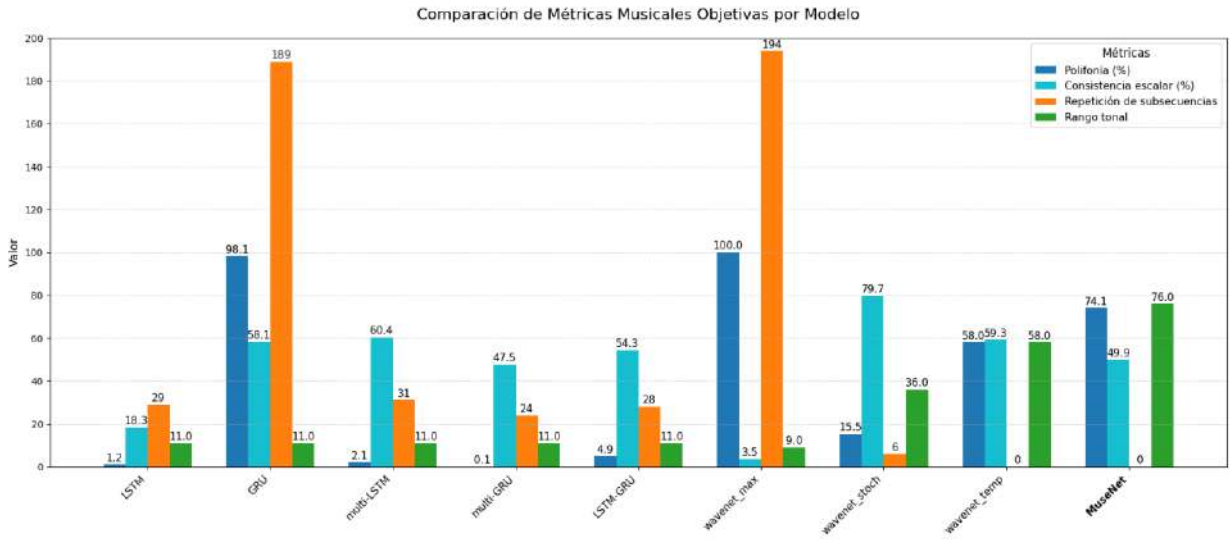
Figura 64. Resultados de las métricas de evaluación objetiva sobre el audio generado más representativo mediante el modelo MuseNet.

La primera métrica indica la puntuación en polifonía que alcanza esta pieza musical. El porcentaje de 74,14 % indica que en muchas ocasiones suenan, al menos, dos notas a la vez. La segunda métrica trata sobre el tipo de escala que mejor se ajusta al audio. El resultado de *abstract mixolydian* sugiere que la pieza tiene un tono alegre, pero combinado con un matiz reflexivo. La tercera métrica muestra el porcentaje del audio generado que corresponde a dicha escala. La siguiente métrica examina la cantidad de secuencias musicales distintas en la composición, mientras que la última mide el rango tonal de la pieza. En este caso, se puede observar que las repeticiones son nulas y el rango tonal es muy elevado. Esto sugiere que la pieza generada posee una gran variedad musical y abarca un mayor abanico de notas. El modelo demuestra su capacidad para trabajar con un amplio espectro tonal, integrando distintos registros y generando secuencias armónicas diversas.

En relación con la valoración subjetiva, la pieza más representativa (*music_musenet*) presenta un contraste dinámico pronunciado desde sus primeros compases, con transiciones abruptas entre pasajes suaves y secciones de alta intensidad sonora. Estos cambios bruscos de volumen, aunque llamativos, generan una sensación de discontinuidad en el flujo general. A esto se suma una sucesión de notas ejecutadas con velocidad extrema en segmentos

posteriores, donde la acumulación de figuras rítmicas complejas resta claridad al discurso melódico. El ritmo, caracterizado por variaciones constantes en su pulsación, contribuye a una percepción de inestabilidad estructural. Si bien la diversidad métrica podría interpretarse como un recurso expresivo, en este caso dificulta la identificación de un patrón conductor que unifique la obra. La ausencia de motivos recurrentes o variaciones temáticas reconocibles impide establecer un núcleo central alrededor del cual se articule la composición. Por tanto, aunque técnicamente se aprecia dominio en la ejecución de pasajes demandantes, la combinación de elementos dispares, los contrastes dinámicos extremos, las secuencias caóticas y los cambios rítmicos impredecibles, resulta en una propuesta auditiva fragmentada y nada agradable.

La evaluación de los ocho modelos detallados, junto con *MuseNet*, revela diferencias en cómo manejan los parámetros musicales objetivos estudiados. La Figura 65 (a.) muestra la polifonía, consistencia escalar, repetición de subsecuencias y rango tonal; mientras que (b.) indica la escala predominante a la que pertenece cada modelo.



a. *Métricas objetivas musicales principales por modelo.*

b. *Escala dominante de cada modelo.*

Figura 65. Comparación de las métricas de evaluación objetiva entre los modelos implementados y MuseNet.

En cuanto a la polifonía, los modelos se distribuyen en un espectro amplio. Los enfoques basados en los modelos *LSTM* (1,18 %), *multi-LSTM* (2,07 %) y *multi-GRU* (0,05 %) muestran preferencia por texturas monótonas, priorizando la claridad melódica lineal. En contraste, *GRU* (98,12 %) y *wavenet_max* (100 %) adoptan una perspectiva radicalmente opuesta, saturando el espacio sonoro con acordes complejos. *MuseNet* (74,14 %) y las variantes *wavenet_stoch* (15,5 %) y *wavenet_temp* (58 %) ocupan posiciones intermedias, combinando momentos de mayor carga armónica con pasajes melódicos definidos.

La consistencia escalar evidencia dos filosofías compositivas. Por un lado, *GRU* y *wavenet_stoch* mantienen altos porcentajes de adherencia (58,1 % y 79.65 % respectivamente), privilegiando la estabilidad tonal. Por otro, *wavenet_max* (3.5 %) y *LSTM* (18.31 %) desafían sistemáticamente los límites escalares, incorporando disonancias y modulaciones abruptas. *MuseNet* equilibra ambos extremos sin comprometer la integridad tonal de su correspondiente escala.

En cuanto a la repetición de subsecuencias, se identifican dos tendencias principales. Modelos como *GRU* y *wavenet_max* priorizan la recurrencia de motivos, generando cohesión a costa de originalidad. En oposición, el resto de las modelos y *MuseNet* maximizan la densidad de frases únicas, evitando repeticiones literales.

El rango tonal completa el panorama comparativo. *MuseNet* destaca con el rango más extenso, superando a modelos como *LSTM* (11 semitonos) y *multi-GRU*, entre otros. Esta amplitud se correlaciona con su capacidad para integrar registros contrastantes sin pérdida de coherencia narrativa. Por otro lado, *wavenet_max*, pese a su complejidad armónica, muestra menor variedad tonal.

La escala predominante presenta correlaciones interesantes con las decisiones estilísticas. Los modelos *LSTM*, *GRU*, *multi-LSTM* y *multi-GRU* convergen en la escala *abstract major*, asociada a expresiones emocionales positivas, mientras que las implementaciones *wavenet* muestran mayor variedad: *abstract lydian* (*wavenet_max*) y *abstract dorian* (*wavenet_stoch* y *wavenet_temp*), donde esta última se caracteriza por su dualidad *melancólico-vital*. *MuseNet* introduce la escala *abstract mixolydian*, combinando vitalidad rítmica con matices reflexivos, demostrando capacidad para sintetizar características escalares contrastantes.

De esta manera, *MuseNet* demuestra superioridad técnica al conciliar características aparentemente antagónicas: alta polifonía sin saturación textural, innovación melódica con coherencia escalar, y amplitud tonal con progresiones lógicas. Esta síntesis sugiere un modelo capaz de trascender las limitaciones entre estructura y creatividad presentes en las implementaciones analizadas.

La evaluación subjetiva de los modelos de generación musical sitúa a *MuseNet* en un extremo singular. Su enfoque contrasta radicalmente con la mayoría de las arquitecturas analizadas, revelando tanto su potencial innovador como sus limitaciones expresivas.

En el ámbito dinámico, *MuseNet* se distingue por cambios abruptos de intensidad sonora, alternando pasajes delicados con explosiones caóticas de volumen. Esta característica lo separa de modelos como *GRU* o *wavenet_stoch*, que priorizan transiciones graduales para mantener la cohesión narrativa. Por otro lado, mientras *wavenet_temp* emplea tensiones armónicas no resueltas como recurso expresivo calculado, *MuseNet* convierte la discontinuidad en un principio estructural, generando una sensación de caos que, aunque técnicamente audaz, fractura la inmersión auditiva. Este enfoque lo acerca más a los defectos de fragmentación observados en *multi-LSTM*.

El ritmo constituye otro eje de análisis. Frente a la pulsación estable de *GRU* o la cadencia introspectiva de *wavenet_temp*, *MuseNet* opta por variaciones métricas constantes y acumulaciones frenéticas de figuras rítmicas. Esta imprevisibilidad, lejos de percibirse como innovación, deriva en una falta de patrón conductor identificable, defecto ausente en *wavenet_stoch*, cuyas frases contrastantes mantienen un hilo narrativo claro. Incluso *LSTM-GRU*, criticado por su estancamiento armónico en fases medias, supera a *MuseNet* en claridad rítmica, demostrando que el dinamismo no requiere sacrificar la claridad melódica.

En el plano armónico, *MuseNet* evidencia su principal debilidad: la ausencia de motivos recurrentes. Mientras *wavenet_stoch* integra disonancias resolutivas dentro de marcos tonales definidos, y *LSTM-GRU* preserva una base melódica identificable pese a sus limitaciones, las composiciones de *MuseNet* se perciben como sucesiones de ideas desconectadas. Este fracaso en la cohesión lo distancia incluso de *wavenet_max*, cuyo enfoque repetitivo al menos mantiene una estructura armónica básica.

El desarrollo temático completa este panorama crítico. Frente a la evolución gradual de *wavenet_temp* o las variaciones sutiles de *GRU*, *MuseNet* carece de dirección clara, acumulando recursos dispares sin integración narrativa. Esta limitación refleja un desafío fundamental: la incapacidad para jerarquizar ideas musicales, resultando en piezas que, aunque técnicamente complejas, fracasan en construir un discurso significativo.

Por lo tanto, *MuseNet* encarna los riesgos de priorizar la innovación técnica sobre los principios de cohesión musical, situándose al lado opuesto de modelos como *GRU* o *wavenet_stoch*. Por ahora, se puede observar que es capaz de generar melodías que requieren una dosificación precisa entre creatividad y una estructura musical compacta.

5.10.2 C-RNN-GAN

El segundo modelo es *C-RNN-GAN*. Se trata de una arquitectura híbrida que combina redes neuronales recurrentes convolucionales (*C-RNN*) con redes generativas adversarias (*GAN*) para la composición musical automatizada. Su diseño integra la capacidad de las *GAN* para aprender distribuciones de datos complejas con la habilidad de las redes recurrentes y convolucionales para capturar dependencias temporales y espaciales en secuencias musicales.

Para la valoración objetiva y subjetiva musical, se ha elegido la pieza generada más representativa (*music_c-rnn-gan*) procedente del repositorio público *SoundCloud*, la cual responde nuevamente ante el objetivo central de este trabajo.

El audio elegido más representativo de este modelo se puede escuchar en el siguiente enlace:

music_c-rnn-gan

La evaluación objetiva de la pieza musical se basa en cinco métricas cuantitativas que permiten medir la calidad del audio generado por el modelo (*music_c-rnn-gan*). Estas métricas incluyen: polifonía, la escala más adecuada, consistencia, cantidad de subsecuencias musicales únicas y rango tonal. En la Figura 66 se presentan los resultados correspondientes.

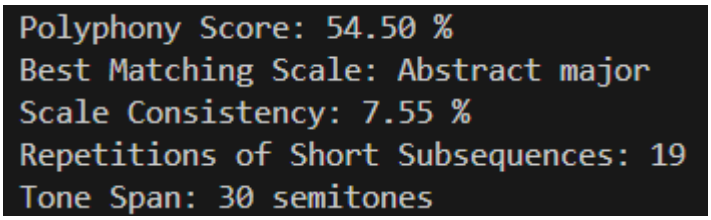
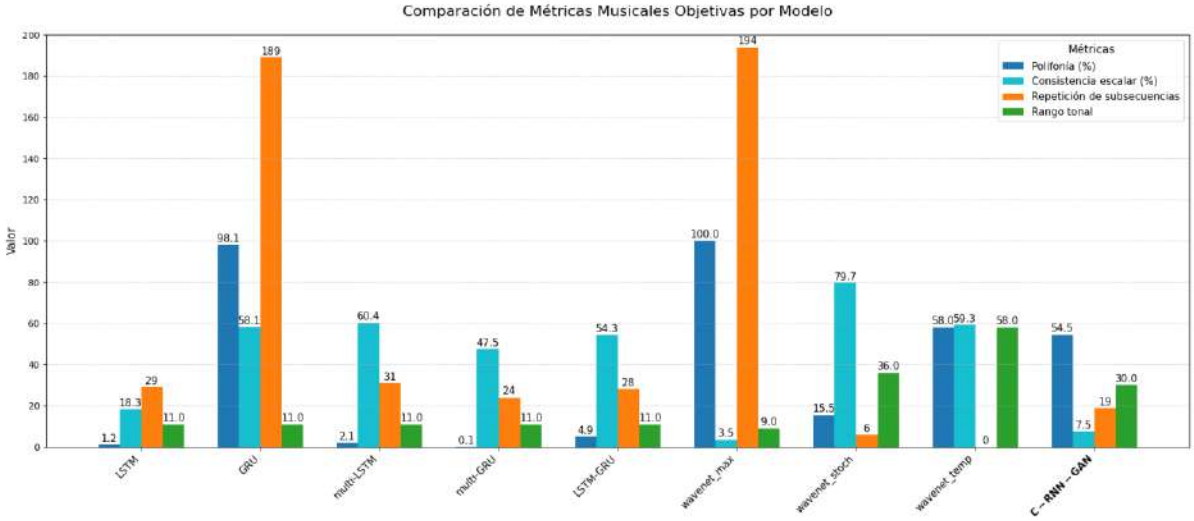


Figura 66. Resultados de las métricas de evaluación objetiva sobre el audio generado más representativo mediante el modelo C-RNN-GAN.

La métrica de polifonía muestra que un 54,50 % de la pieza presenta al menos dos notas sonando simultáneamente, lo que indica una presencia considerable de superposición armónica. En cuanto a la escala, el resultado *abstract major* sugiere un carácter alegre en la composición. El grado de consistencia refleja el porcentaje del audio que se ajusta a esa escala dominante. La siguiente métrica analiza el número de secuencias musicales únicas en la composición, mientras que la última muestra la amplitud tonal de la pieza. En este caso, se puede observar que las repeticiones son moderadas y el rango tonal es elevado. Esto indica que la pieza generada posee una mayor diversidad musical y es capaz de combinar distintas notas para crear frases musicales originales.

En cuanto a la evaluación subjetiva mediante escucha activa, la obra representativa (*music_c-rnn-gan*) se desarrolla manteniendo una intensidad inicial contenida, con un planteamiento melódico que busca establecer una idea musical definida desde los primeros compases. El ritmo, aunque constante y de pulsación rápida, sirve como base para articular frases estructuradas que demuestran intencionalidad compositiva. Por tanto, a lo largo de la pieza se percibe un esfuerzo por desarrollar un discurso coherente mediante la repetición variada de un núcleo temático principal. La obra consigue transmitir sensación de integridad gracias a la claridad de su idea y la ejecución precisa de los elementos rítmicos. No obstante, la persistencia en un mismo registro emocional rápido y una muy escasa variedad en los recursos melódicos empleados reduce su capacidad para sostener el interés auditivo.

Así mismo, se realiza una comparativa similar con *C-RNN-GAN* donde se analizan los parámetros musicales objetivos anteriores. En la Figura 67, (a.) presenta la comparativa entre estas métricas objetivas y (b.) la escala principal de cada modelo.



a. *Métricas objetivas musicales principales por modelo.*

b. *Escala dominante de cada modelo.*

Figura 67. Comparación de las métricas de evaluación objetiva entre los modelos implementados y C-RNN-GAN.

En el ámbito de la polifonía, *C-RNN-GAN* registra un 54,50 %, cifra que lo sitúa en una zona intermedia. Supera claramente la austeridad melódica de *multi-GRU* (0,05 %) y *LSTM* (1,18 %), pero se distancia de la saturación acórdica característica de GRU (98,12 %) y *wavenet_max* (100 %). Esta equidistancia refleja una estrategia compositiva que mide la complejidad armónica, combinando pasajes lineales con armonizaciones estratégicas, en sintonía con el enfoque moderado de *wavenet_temp* (58 %), aunque con mayor variedad en la disposición interválica.

La repetición de subsecuencias emerge como uno de sus atributos distintivos. Con repeticiones moderadas y un inventario amplio de frases únicas, supera la

rigidez cíclica de *GRU* y *wavenet_max*, aunque sin igualar la inventiva melódica de *multi-LSTM* o *LSTM-GRU*. Aquí se introduce un factor diferenciador: la capacidad para generar transiciones abruptas entre registros, rasgo que contrasta con la progresividad fluida de *wavenet_temp* y añade un matiz de imprevisibilidad controlada a sus composiciones.

En cuanto al rango tonal, el modelo demuestra un rango extenso, superando a *LSTM* (11 semitonos) y *multi-GRU*, aunque con menor cohesión narrativa que *wavenet_stoch*. Esta amplitud refleja su habilidad para integrar saltos interválicos, característica compartida parcialmente con las incursiones modales esporádicas de *LSTM-GRU*, pero ejecutada con mayor sistematicidad.

Respecto a la escala predominante, *C-RNN-GAN* se alinea con *LSTM*, *GRU* y *multi-LSTM* al adoptar *abstract major* como base tonal. Sin embargo, su implementación difiere sustancialmente. Mientras *GRU* replica la escala con fidelidad casi absoluta (98,12 % de coherencia), este modelo introduce disonancias calculadas y modulaciones puntuales, aproximándose a la libertad tonal de *LSTM* (18,31 %) sin caer en el experimentalismo radical de *wavenet_max* (3,5 %). Esta negociación entre tradición e innovación lo acerca a *wavenet_temp* en flexibilidad, aunque preservando el carácter alegre inherente a *abstract major*.

Por último, *C-RNN-GAN* representa un equilibrio entre principios antagónicos: disciplina armónica fija y coraje creativo para elaborar composiciones que destacan por incorporar frases musicales más fructíferas que el resto de los modelos estudiados.

Desde una perspectiva subjetiva, *C-RNN-GAN* se percibe como un modelo de equilibrio intermedio en el panorama de generación musical. Logra cierta coherencia estructural, pero presenta restricciones en su expresividad, lo que lo diferencia tanto de propuestas más arriesgadas como *wavenet_temp* como de opciones más tradicionales como *GRU*. Su análisis crítico sugiere una intención clara de mantener una composición coherente, aunque aún muestra limitaciones en cuanto a la variedad emocional.

C-RNN-GAN destaca por su capacidad para establecer un núcleo melódico definido desde los primeros compases, rasgo que lo distancia de modelos como *LSTM* o *multi-GRU*. Su enfoque de repetición variada del tema principal evita la monotonía literal de *wavenet_max*, acercándose a la estrategia de *wavenet_stoch* en el uso de frases melódicas contrastantes. Así mismo, mientras *wavenet_stoch* resuelve tensiones con precisión, *C-RNN-GAN* se limita a variaciones rítmicas superficiales sin explorar desarrollos tonales profundos.

El modelo emplea un ritmo constante y rápido como columna vertebral, estrategia compartida con *LSTM-GRU* en su fase inicial. No obstante, a diferencia de este, *C-RNN-GAN* mantiene un pulso sostenido, logrando fluidez comparable a *GRU* pero sin su tendencia a la previsibilidad. Este enfoque lo sitúa por encima de *multi-LSTM*, aunque por debajo de *wavenet_temp*, cuya pulsación constante se enriquece con atmósferas introspectivas. Su principal debilidad rítmica radica en la escasa modulación dinámica.

Por otro lado, su limitación más notoria es la persistencia en un mismo registro emocional. Frente a la melancolía reflexiva de *wavenet_temp* o la vitalidad contenida de *GRU*, *C-RNN-GAN* se estanca en un tono uniforme que, aunque técnicamente correcto, fracasa en evolucionar narrativamente. Esta carencia lo acerca a *wavenet_max* en monotonía expresiva, aunque con una base estructural más sólida.

C-RNN-GAN representa un paso hacia la composición automática musical equilibrada, demostrando que es posible generar obras estructuradas sin caer en la repetición mecánica de *wavenet_max*. No obstante, su avance queda incompleto. La falta de ambición emocional y desarrollo armónico lo condenan a un conservadurismo expresivo que, aunque funcional, no trasciende lo convencional. Una posible alternativa versa sobre tomar prestados recursos de *wavenet_stoch* o *LSTM-GRU* que ayuden a paliar estas limitaciones. No obstante, su verdadero valor reside en señalar un camino intermedio compuesto por la integración de su solidez técnica con mecanismos de innovación musical.

5.10.3 GanSynth

El último modelo es *GanSynth*. Se trata de una arquitectura innovadora que fusiona redes generativas adversarias (*GAN*) con técnicas de síntesis de espectrogramas para la generación musical automatizada. Su diseño aprovecha la capacidad de las *GAN* para producir datos de alta calidad y realismo, combinándolas con mecanismos de atención multiescalar y transformaciones de estilo que capturan características armónicas y temporales en la música

Para la valoración objetiva y subjetiva musical, se ha elegido la pieza generada más representativa (*music_gansynth*) procedente del repositorio público *SoundCloud*, que vuelve a coincidir con el objetivo principal de este estudio.

El audio elegido más representativo de este modelo se puede escuchar en el siguiente enlace:

music_gansynth

La evaluación objetiva sobre la pieza musical se realiza mediante cinco métricas cuantitativas que miden la calidad del audio generado (*music_gansynth*). La Figura 68 muestra los resultados obtenidos.

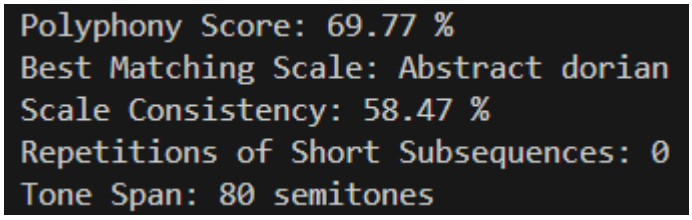


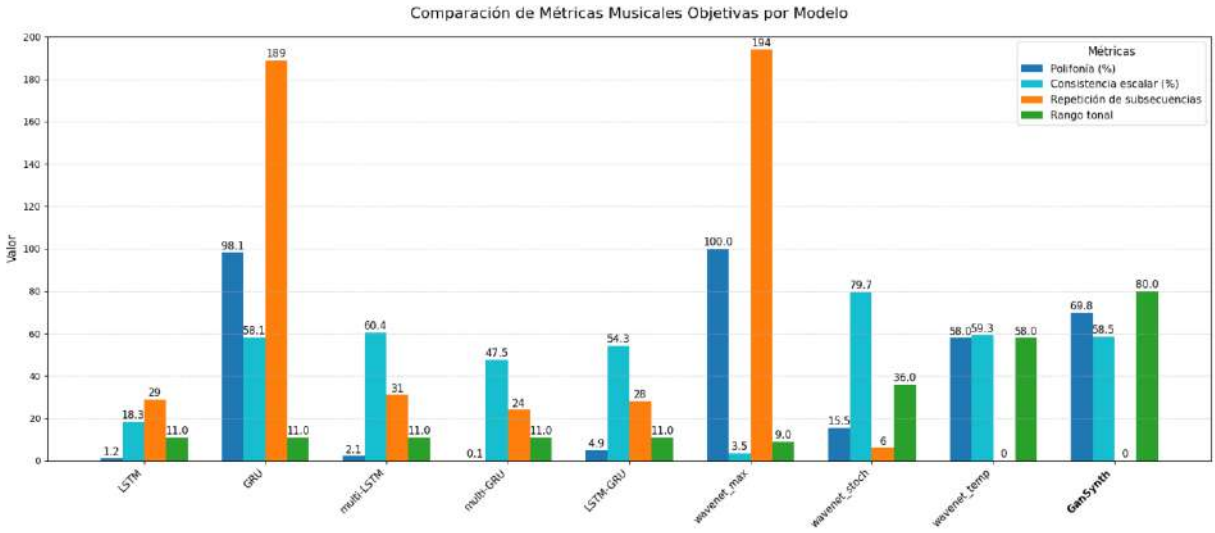
Figura 68. Resultados de las métricas de evaluación objetiva sobre el audio generado más representativo mediante el modelo GanSynth.

La primera métrica presenta la puntuación en polifonía que alcanza esta pieza musical. El porcentaje de 69,77 % indica que es muy frecuente que al menos dos notas suenan a la vez durante la pieza. La segunda métrica trata sobre el tipo de escala que mejor se ajusta al audio. El resultado de *abstract dorian* indica que la pieza posee un tono apagado pero despierto. La tercera métrica muestra el porcentaje del audio generado que corresponde a dicha escala. La siguiente métrica examina la cantidad de secuencias musicales distintas en la composición, mientras que la última mide el rango tonal de la pieza. En este caso, se puede observar que las repeticiones son nulas y el rango tonal es muy elevado. Esto sugiere que la pieza generada presenta una gran variedad musical y logra integrar diferentes notas para construir frases melódicas únicas y creativas.

Con relación a la crítica musical subjetiva, la pieza más representativa (*music_gansynth*) se desarrolla desde un planteamiento rítmico inicial de

carácter animado y fluido, estableciendo una base que aporta vitalidad sin resultar apresurada. La introducción de contrastes temporales en secciones medias, con pasajes más pausados que alternan con los motivos originales, genera una dinámica efectiva, permitiendo destacar las frases musicales mediante articulaciones precisas en la ejecución de las notas. El material melódico, aunque estructurado en torno a un núcleo central recurrente, demuestra versatilidad al explorar diferentes registros de la escala, particularmente hacia el tramo final. La integración funcional de la armonía mantiene el discurso fluido y sin interrupciones. A nivel compositivo, se aprecia coherencia en el tratamiento temático y dominio de los recursos básicos de desarrollo. Si bien la persistencia en patrones rítmicos similares limita la aparición de contrastes profundos, la obra logra sostener el interés mediante variaciones y cambios sutiles. El resultado final ofrece una experiencia auditiva equilibrada que combina energía rítmica con claridad estructural.

De igual manera, se compara *GanSynth* con los anteriores modelos estudiados La Figura 69 indica las diferencias entre las métricas objetivas musicales. En (a.) se señala la polifonía, la estabilidad tonal, la diversidad estructural y el rango tonal; y en (b.) la escala dominante en cada modelo.



a. *Métricas objetivas musicales principales por modelo.*

b. *Escala dominante de cada modelo.*

Figura 69. Comparación de las métricas de evaluación objetiva entre los modelos implementados y GanSynth.

En polifonía, *GanSynth* registra un 69,77 %, superando claramente a modelos como *wavenet_temp* (58 %), aunque sin alcanzar la saturación absoluta de *wavenet_max* (100 %). Este valor lo posiciona como el segundo modelo más denso en superposiciones armónicas, pero con una diferencia cualitativa. *GanSynth* genera texturas en evolución constante, donde las notas interactúan dinámicamente sin perder individualidad melódica.

Respecto a la escala predominante, adopta *abstract dorian*, lo que le confiere un carácter dual: melancolía introspectiva combinada con pulsos rítmicos vibrantes. Sin embargo, difiere de sus homólogos *WaveNet*. Estas implementaciones alternan entre una adherencia estricta (79,65% en *wavenet_stoch*) y libertad moderada (59,26 % en *wavenet_temp*). En cambio, *GanSynth* logra un equilibrio más orgánico, integrando disonancias dentro del marco escalar sin porcentajes explícitos de coherencia, pero manteniendo una identidad tonal reconocible.

En repeticiones de subsecuencias, destaca al eliminarlas por completo, superando incluso a *multi-GRU* y *wavenet_stoch* en densidad de frases únicas. Su arquitectura genera transiciones impredecibles pero coherentes, evitando tanto el caos de *LSTM* (18,31 % de ajuste escalar) como la rigidez de *wavenet_stoch* (79,7 % de consistencia escalar).

El rango tonal emerge como otro de sus pilares, con un rango que supera a la mayoría de los modelos analizados. Esta cualidad lo acerca a *wavenet_temp* en versatilidad registral de diferentes notas.

Por tanto, *GanSynth* representa una síntesis innovadora entre corrientes antagónicas. De los modelos *WaveNet* hereda la densidad armónica y la exploración modal, pero evita sus excesos mediante una gestión más dinámica del espacio tonal. De las redes recurrentes (*LSTM*, *GRU* y variantes) adopta principios de progresión temática, aunque trascendiendo sus limitaciones en originalidad melódica.

La valoración crítica musical de *GanSynth* respecto a los demás modelos implementados lo sitúa como un modelo destacado en la generación de composiciones equilibradas, combinando vitalidad rítmica con coherencia estructural.

GanSynth sobresale por su ritmo animado y fluido, evitando tanto la rigidez de *multi-LSTM* como la repetición mecánica de *wavenet_max*. A diferencia de *LSTM-GRU*, introduce contrastes temporales efectivos a través de pasajes pausados que alternan con motivos originales, estrategia comparable a *wavenet_stoch* en su uso de tensiones resolutivas. Este enfoque lo distancia de *GRU*, cuyo *tempo andante* carece de modulaciones significativas.

El modelo demuestra versatilidad registral, explorando distintos rangos de la escala sin caer en la fragmentación de *multi-GRU*. Su núcleo melódico recurrente evita la aleatoriedad de *LSTM*, acercándose a la cohesión de *wavenet_temp*, aunque sin su profundidad introspectiva. La integración armónica funcional supera a *wavenet_max* en variedad, pero no iguala la precisión técnica de *wavenet_stoch* en el manejo de disonancias.

Por otro lado, la capacidad para sostener el interés mediante variaciones sutiles constituye su mayor fortaleza. Frente al estancamiento de *LSTM-GRU* en desarrollos prolongados o la previsibilidad de *GRU*, *GanSynth* logra un equilibrio cercano a *wavenet_stoch*, aunque con menor sofisticación. Su estrategia de exploración gradual del núcleo temático musical supera la linealidad de *multi-LSTM*, demostrando habilidad para renovar ideas sin fracturar la cohesión. No obstante, carece del arco narrativo definido de *wavenet_temp*, cuyas composiciones evolucionan hacia un apogeo mejor construido.

En el ámbito de la expresividad, *GanSynth* ofrece una energía contenida más cercana a *GRU*. Su tono predominante lo distancia de la melancolía reflexiva de *wavenet_temp* y la serenidad estática de *wavenet_stoch*. Si bien logra transmitir claridad y optimismo, su registro emocional se oscurece frente a la dualidad de *wavenet_stoch* que combina melancolía con pulsos vibrantes.

GanSynth representa un avance significativo en composición automática musical, ofreciendo fiabilidad estructural sin caer en la frialdad expresiva de modelos como *multi-GRU*. Sin embargo, su virtud de equilibrio se convierte en limitación al evitar riesgos creativos que mejorarían singularidad artística.

5.10.4 Comparativa global

Se realiza a continuación una comparativa general entre los modelos implementados y las tres arquitecturas existentes: *MuseNet*, *C-RNN-GAN* y *GanSynth*. La Figura 70 resume las métricas objetivas consideradas en el análisis musical.

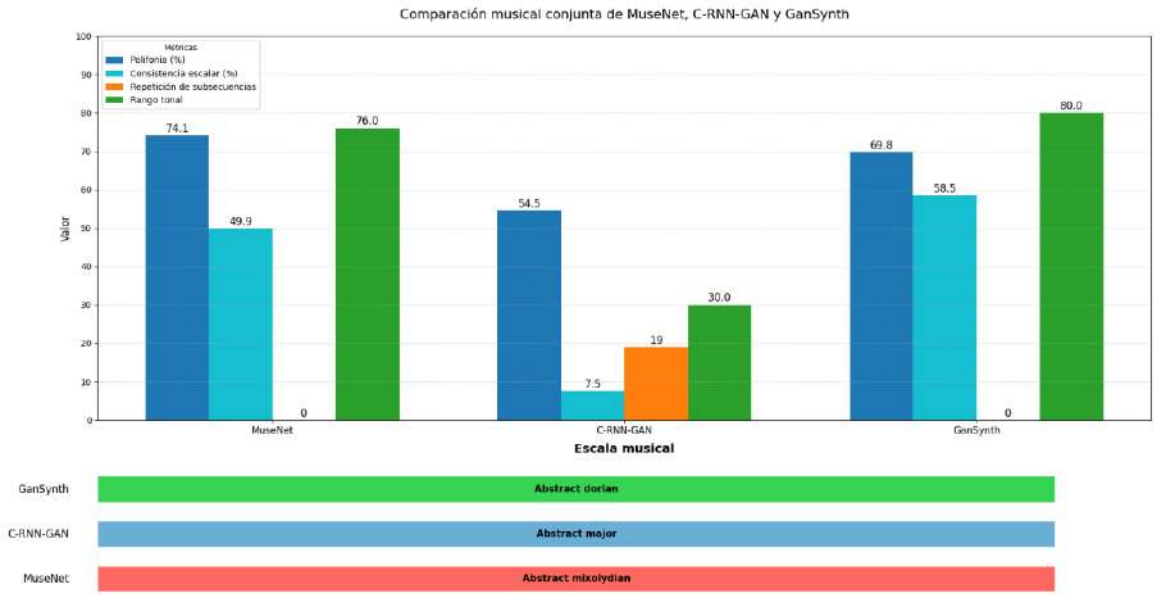


Figura 70. Comparación global de las métricas de evaluación objetiva entre los modelos de generación musical actuales.

Cada arquitectura muestra perfiles únicos. *LSTM* brilla en creatividad melódica (baja polifonía: 1,18 %, frases diversas), pero sufre desviaciones tonales que generan malestar auditivo. *GRU*, con su alta polifonía (98,12 %) y transiciones fluidas, asegura coherencia estructural, aunque sacrifica originalidad al recurrir a repeticiones predecibles. Los modelos *multi-LSTM* y *multi-GRU*, pese a su innovación en patrones no repetitivos, caen en rigidez rítmica y fragmentación armónica, limitando su expresividad. *LSTM-GRU* combina una energía vibrante inicial y variaciones rítmicas dinámicas, pero su desarrollo armónico se estanca, revelando dificultades en gestionar progresiones prolongadas.

En el ámbito de *WaveNet*, *wavenet_max* (100 % polifonía) prioriza texturas melódicas densas, pero su insistencia en patrones cíclicos lo condena a la monotonía. Así mismo *wavenet_stoch* destaca al equilibrar disonancias resolutivas (79,65 % de ajuste a *abstract dorian*) con fluidez narrativa, aunque su impacto emocional decrece frente a *wavenet_temp*, que utiliza tensiones no resueltas para crear atmósferas introspectivas (59,26 % de coherencia en *abstract dorian*), logrando profundidad sin perder cohesión. *MuseNet*, aunque incorpora contrastes dinámicos y complejidad, fracasa en establecer una dirección musical, derivando en caos estructural. *C-RNN-GAN* y *GanSynth* representan el equilibrio. El primero ofrece polifonía moderada (54,50 %) y seguridad tonal; el segundo, vitalidad rítmica (69,77 % polifonía) y claridad melódica, aunque ambos carecen de expresividad emocional.

Como reflexión final, *wavenet_temp* se consolida como el modelo óptimo para generación musical automatizada al sintetizar de manera excepcional tres pilares fundamentales en el ámbito musical: innovación expresiva, coherencia melódica y profundidad emocional. Su capacidad para integrar tensiones armónicas no resueltas (59,26 % de ajuste a *abstract dorian*) como recurso narrativo le permite construir atmósferas musicales introspectivas sin sacrificar la fluidez estructural. Además es capaz de incorporar complejidad musical evidenciado en su polifonía del 58 %, a la vez que su ritmo constante actúa como eje unificador, permitiendo evoluciones melódicas únicas sin repeticiones uniformes. Estas características lo posicionan como la opción más completa

entre los modelos evaluados, demostrando que la composición automática musical mediante redes neuronales puede trascender la simple imitación de patrones y aspirar a una expresividad musical auténtica.

6 Conclusiones

Este trabajo fin de grado ha analizado de forma crítica y experimental distintas estrategias computacionales para generar composiciones musicales mediante inteligencia artificial, centrándose en evaluar su potencial creativo, coherencia armónica y estructura musical. Se comparan dos enfoques principales. Por una parte, se han desarrollado modelos basados en redes neuronales recurrentes (*LSTM* y *GRU*, con una y varias capas de celdas, y combinaciones de las anteriores). Por otra parte, se ha implementado una versión optimizada de *WaveNet*, diseñada para operar con recursos limitados. Para enmarcar los resultados dentro del estado actual del campo, se ha realizado un análisis comparativo con sistemas generativos actuales como *MuseNet*, *C-RNN-GAN* y *GanSynth*, lo que ha permitido identificar tanto avances como desafíos futuros.

Los modelos neuronales basados en *LSTM* y *GRU* demuestran ser eficaces para generar secuencias melódicas con ritmos variados y estructuras musicales correctas, aunque con limitaciones en la polifonía. Las implementaciones monocapa de *LSTM* logran generar piezas con bajo porcentaje de acordes (1,18 %), priorizando melodías lineales y evitando disonancias armónicas, aunque con cierta monotonía rítmica. En contraste, las *LSTM* multicapa elevan el índice de polifonía a un 2,07 %. Además, la pieza representativa generada presenta un ritmo más dinámico y una melodía más agradable. Por otro lado, el modelo *GRU* monocapa muestra mayor capacidad para integrar notas simultáneas (98,12 % de polifonía). Así mismo, el audio generado por este modelo logra alcanzar una estructura coherente, con una base melódica sólida y una progresión rítmica fluida. Sin embargo, el modelo *GRU* multicapa no alcanza un índice de polifonía tan elevado (0,05 %). Además, la pieza representativa generada muestra un ritmo constante pero rígido, con una articulación entrecortada que afecta la fluidez. La uniformidad rítmica la vuelve monótona, mientras que la falta de cohesión armónica refuerza una sensación de aleatoriedad, debilitando su estructura. La combinación híbrida *LSTM-GRU* destaca por su dinamismo y fluidez inicial, aunque su desarrollo posterior evidencia repeticiones temáticas que limitan la complejidad expresiva.

La adaptación implementada de *WaveNet*, pese a su tendencia al sobreajuste solventado mediante técnicas de regularización, genera composiciones abundantes en acordes, con variedad tonal y de escalas. No obstante, su dependencia de secuencias predefinidas resulta en repeticiones predecibles, especialmente con el método de mayor probabilidad (*wavenet_max)*. La implementación del método estocástico (*wavenet_stoch*) permite diversificar las secuencias generadas, lo que resulta en melodías con mayor variedad rítmica, exploración de nuevas frases musicales y agradables al oído. Este enfoque, aplicado a escalas como la dórica, introduce matices melancólicos pero dinámicos, a la vez que unas correctas transiciones armónicas. La incorporación del parámetro de temperatura (*temperature*) facilita un equilibrio entre innovación y coherencia. Tras ajustar este parámetro, se logran composiciones más sorprendentes como *wavenet_temp*, donde las notas menos probables pero armónicamente compatibles enriquecen la textura musical. Sin embargo, este incremento en la exploración tonal también acentúa disonancias puntuales que no afectan de manera negativa a la melodía. Ambos métodos muestran que la gestión de la incertidumbre durante la generación es clave para expandir la creatividad de la pieza sin comprometer la calidad auditiva.

Como conclusión, es posible afirmar que la adaptación de *WaveNet* genera resultados aceptables en entornos con recursos limitados, siempre que se apliquen técnicas de regularización y ajustes probabilísticos. Así mismo, la combinación de *LSTM* y *GRU* es una alternativa viable para composiciones breves y dinámicas, aunque su eficacia disminuye en piezas extensas debido a la recurrencia temática.

Al comparar estos resultados con modelos generativos actuales, se observa que *MuseNet* y *C-RNN-GAN* destacan por su capacidad para generar piezas con alta diversidad tonal y complejidad polifónica. *MuseNet* explora contrastes dinámicos extremos y modulaciones impredecibles, reflejando un potencial creativo cercano al experimentalismo humano. *C-RNN-GAN* tiende a priorizar patrones rápidos y repetitivos, sacrificando profundidad expresiva. No obstante, ambas aproximaciones presentan inconsistencias estructurales, como transiciones abruptas entre secciones y secuencias rítmicamente caóticas, lo que limita su naturalidad. Por su parte, *GanSynth* se distingue por una mayor coherencia armónica y fluidez en las progresiones, gracias a su enfoque en la representación espectral de sonidos. Por tanto, resulta en un audio generado equilibrado que combina una polifonía notable (69,77 %) con transiciones suaves entre acordes, manteniendo una estructura rítmica fluida y evitando contrastes disruptivos.

Este trabajo presenta además una metodología para evaluar la calidad musical generativa, integrando métricas objetivas (polifonía, ajuste a escalas, diversidad de frases) con análisis subjetivos basados en escucha activa. La implementación de un programa en Python para cuantificar aspectos como la armonía o el rango tonal constituye una herramienta innovadora que facilita la comparación sistemática entre modelos.

En el ámbito práctico, el trabajo subraya la importancia del preprocesamiento de datos, como la normalización de octavas y duraciones en archivos MIDI, para optimizar el entrenamiento de los modelos. Así mismo, evidencia que la elección de hiperparámetros, como el número de capas convolucionales causales en *WaveNet* o el número de capas de celdas en *LSTM* y *GRU*, incide directamente en la creatividad y estabilidad de las generaciones. En particular, se resalta que en *GRU*, al tener una arquitectura más simple con solo dos puertas (*reset* y *update*), aumentar el número de capas no incrementa significativamente la complejidad computacional. Esto permite entrenamientos más eficientes sin comprometer la coherencia melódica. Así, se refuerza que la simplicidad estructural de *GRU* lo convierte en una opción robusta para entornos con restricciones de recursos, siempre que se priorice la fluidez rítmica y melódica sobre la polifonía.

Si bien los modelos evaluados no igualan la riqueza compositiva humana, se establece un marco de referencia para futuros trabajos en inteligencia artificial generativa para la composición musical a piano. Los resultados sugieren que la combinación de enfoques, como integrar la polifonía de *wavenet_temp* con la fluidez rítmica de *GRU* y aprovechar la capacidad del modelo híbrido *LSTM-GRU* para generar piezas breves con un equilibrio entre estructura melódica coherente y variaciones rítmicas dinámicas, puede superar las limitaciones individuales de cada arquitectura. De esta manera, este trabajo no solo avanza en la comprensión técnica de las redes neuronales aplicadas a la generación musical, sino que también ofrece pautas para equilibrar innovación, corrección armónica y viabilidad computacional en este ámbito emergente de la generación musical automatizada.

7 Líneas futuras

Este trabajo de fin de grado plantea líneas futuras orientadas a profundizar en la integración de elementos compositivos y métricas innovadoras para la generación y evaluación musical mediante redes neuronales.

En primer lugar, se plantea desarrollar modelos que gestionen la polifonía integrando arquitecturas híbridas, las cuales combinen redes neuronales recurrentes y mecanismos de atención. Este método permitiría procesar múltiples voces o instrumentos en paralelo, emulando la riqueza estructural de las composiciones humanas. La incorporación de atención mejoraría la detección y el énfasis en patrones melódicos o armónicos críticos durante la creación.

Así mismo, se sugiere incorporar mecanismos de atención bidireccionales en redes *LSTM* y *GRU* para modular dinámicamente cómo influyen las secuencias pasadas y futuras en la generación. Esto permitiría ajustar en tiempo real la coherencia de las piezas, priorizando elementos melódicos según parámetros preestablecidos como la complejidad musical. Adicionalmente, estos sistemas facilitarían adaptar patrones a géneros diversos (transiciones entre clásico y jazz), manteniendo la melodía base mientras se transforma el estilo y se exploran nuevas variaciones creativas.

En relación con el modelo *WaveNet*, se propone incrementar su complejidad para mejorar la calidad y coherencia de las secuencias musicales generadas. Para ello, se exploraría la ampliación de su arquitectura basada en convoluciones dilatadas, aumentando tanto la profundidad de las capas como el alcance de sus campos receptivos. Esto permitiría capturar dependencias a más largo plazo en las estructuras musicales, esenciales para modelar aspectos como la repetición temática o la progresión armónica.

Finalmente, se plantea el desarrollo de métricas objetivas centradas en detectar disonancias armónicas producidas durante la generación musical, además de evaluar parámetros como la energía o la intensidad melódica. De esta forma, se establecería un sistema de evaluación dual que combine precisión técnica con percepción humana.

8 Análisis de impacto

En este capítulo se analiza el impacto sociocultural derivado de este trabajo, centrado en la generación automática de música mediante inteligencia artificial (IA), explorando su relación con los Objetivos de Desarrollo Sostenible (ODS) de la Agenda 2030. La tecnología analizada no solo es un asunto técnico; también influye en la sociedad, la cultura y la economía. Por ello, es clave examinar sus efectos con una mirada crítica, considerando tanto sus oportunidades como los desafíos éticos que plantea, en línea con reflexiones recientes sobre responsabilidad en el uso de la IA generativa musical. A continuación, se desglosan sus alcances y su conexión con los ODS, los cuales se pueden observar en la Figura 71 [93].

Figura 71. Objetivos de Desarrollo Sostenible relacionados con este Trabajo de Fin de Grado [93].

La democratización de plataformas de composición asistida por algoritmos de IA elimina requisitos económicos y técnicos, permitiendo que individuos sin capacitación formal desarrollen proyectos creativos. Este hecho se conecta directamente con el ODS 10 (*Reducción de las desigualdades*), específicamente con la meta 10.2: «*Potenciar y promover la inclusión social, económica y política de todas las personas, independientemente de su edad, sexo, discapacidad, raza, etnia, origen, religión o situación económica u otra condición*». Al suprimir barreras de entrada en sectores culturales, se redistribuyen oportunidades históricamente concentradas en grupos privilegiados. Sin embargo, como señalan algunos expertos [94], el acceso a estas herramientas no garantiza equidad si los conjuntos de datos utilizados reflejan sesgos históricos, como priorizar estilos musicales occidentales sobre el folclore local, lo que puede perpetuar exclusiones.

Por otro lado, surge simultáneamente un dilema legal: la ambigüedad en la autoría de piezas musicales creadas mediante inteligencia artificial desafía los sistemas tradicionales de derechos de autor. Esta problemática se vincula al ODS 16 (*Paz, justicia e instituciones sólidas*), en particular a la meta 16.6: «*Crear*

instituciones eficaces, responsables y transparentes a todos los niveles». Casos como *Lost Tapes of the 27 Club*, donde se generaron canciones de artistas fallecidos mediante IA [95], evidencian la urgencia de actualizar normativas. Japón, por ejemplo, ha modificado sus leyes para diferenciar entre obras *totalmente generadas por IA* y aquellas con intervención humana, un modelo que podría inspirar reformas globales [96]. Por tanto, resulta fundamental crear reglas claras que definan las responsabilidades entre el trabajo humano y el automatizado, lo que asegura un reconocimiento justo y evita conflictos legales que puedan afectar la confianza en las instituciones.

En el ámbito artístico, la combinación de diferentes tradiciones musicales a través de algoritmos de IA puede revitalizar el patrimonio cultural mundial. Este potencial se alinea con el ODS 11 (*Ciudades y comunidades sostenibles*), concretamente con la meta 11.4: «*Redoblar los esfuerzos para proteger y salvaguardar el patrimonio cultural y natural del mundo*». No obstante, este avance debe proteger las expresiones musicales locales, buscando un equilibrio entre innovación y preservación cultural.

Las aplicaciones educativas de estos sistemas permiten experimentar con teoría musical sin instrumentos físicos, mientras que en el mercado del entretenimiento agilizan procesos creativos. Estas funciones apoyan el ODS 4 (*Educación de calidad*), enfocándose en la meta 4.4: «*Aumentar considerablemente el número de jóvenes y adultos que tienen las competencias necesarias, en particular técnicas y profesionales, para acceder al empleo, el trabajo decente y el emprendimiento*». Paralelamente, dinamizan el ODS 8 (*Trabajo decente y crecimiento económico*) mediante la meta 8.2: «*Lograr niveles más elevados de productividad económica mediante la diversificación, la innovación tecnológica y la creatividad*», al generar empleos emergentes en sectores creativos.

Por lo tanto, la interacción entre la composición musical automatizada y desarrollo sostenible presenta así un doble filo: mientras amplía el acceso cultural (ODS 10), preserva tradiciones (ODS 11) y moderniza la formación técnica (ODS 4 y 8), también exige reformular marcos jurídicos (ODS 16). Para aplicarlo de manera efectiva es necesario adoptar enfoques éticos que prioricen la transparencia, la equidad en el diseño de algoritmos y la protección de derechos fundamentales en los compositores y géneros musicales tradicionales.

9 Bibliografía

[1] D. Herremans, J. D. Martín, E. Chew. *A Functional Taxonomy of Music Generation Systems*, *ACM Computing Surveys*, vol. 50, no. 5, pp. 1–30, Sep. 2017. Accedido el 15 de septiembre de 2024. [En línea]. Disponible: https://doi.org/10.1145/3108242

[2] D. Eck, J. Schmidhuber. *Learning the Long-Term Structure of the Blues*, in *Artificial Neural Networks — ICANN 2002*, Lecture Notes in Computer Science, vol. 2415, Springer, Berlin, Heidelberg, 2002, pp. 284–289. Accedido el 15 de septiembre de 2024. [En línea]. Disponible: https://link.springer.com/chapter/10.1007/3-540-46084-5_47

[3] Y. Wu, T. Hayashi, P. L. Tobing, K. Kobayashi, T. Toda. *Quasi-Periodic WaveNet Vocoder: A Pitch Dependent Dilated Convolution Model for Parametric Speech Generation*. Published 2019 Jul 1. Accedido el 17 de septiembre de 2024. [En línea]. Disponible: https://doi.org/10.48550/arXiv.1907.00797

[4] A. Wali, S. Mahamad, S. Sulaiman. *Task Automation Intelligent Agents: A Review*, *Future Internet*, vol. 15, n.º 6, p. 196, mayo de 2023. Accedido el 18 de septiembre de 2024. [En línea].
Disponible: https://doi.org/10.3390/fi15060196

[5] W. S. McCulloch, W. Pitts. *A logical calculus of the ideas immanent in nervous activity*, *Bull. Math. Biophys*, vol. 5, n.º 4, pp. 115–133, diciembre de 1943. Accedido el 18 de septiembre de 2024. [En línea]. Disponible: https://doi.org/10.1007/bf02478259

[6] R. Prieto, A. Herrera, J. L. Pérez, A. Padrón. *El modelo neuronal de Mcculloch y Pitts: Interpretación Comparativa del Modelo*, *Laboratorio de Computación Adaptativa, Centro de Instrumentos, UNAM*, México D.F. Accedido el 18 de septiembre de 2024. [En línea]. Disponible: https://www.researchgate.net/publication/343141076_EL_MODELO_NEURONAL_DE_McCULLOCH_Y_PITTS_Interpretacion_Comparativa_del_Modelo

[7] F. Rosenblatt. *The perceptron: A probabilistic model for information storage and organization in the brain*, *Psychology. Rev.*, vol. 65, n.º 6, pp. 386–408, 1958. Accedido el 19 de septiembre de 2024. [En línea]. Disponible: https://doi.org/10.1037/h0042519

[8] *Grupo us. Conceptos básicos sobre redes neuronales.* Accedido el 20 de septiembre de 2024. [En línea]. Disponible: https://grupo.us.es/gtocoma/pid/pid10/RedesNeuronales.htm

[9] C. Clabaugh, D. Myszewski, J. Pang. *Neural Networks - Neuron*. Computer Science. Accedido el 20 de septiembre de 2024. [En línea]. Disponible: https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/Neuron/index.html

[10] G. S. Ahmed. *SELU (Scaled Exponential Linear Unit) activation function*. OpenGenus IQ: Learn Algorithms, DL, System Design. Accedido el 23 de septiembre de 2024. [En línea]. Disponible: https://iq.opengenus.org/scaled-exponential-linear-unit/

[11] S. Linnainmaa. *Taylor expansion of the accumulated rounding error*, *BIT*, vol. 16, n.º 2, pp. 146–160, junio de 1976. Accedido el 24 de septiembre de 2024. [En línea]. Disponible: https://doi.org/10.1007/bf01931367

[12] D.E. Rumelhart, G.E. Hinton, R.J. Williams. *Learning representations by back-propagating errors, Nature*, vol. 323, n.º 6088, pp. 533–536, octubre de 1986. Accedido el 24 de septiembre de 2024. [En línea]. Disponible: https://doi.org/10.1038/323533a0

[13] *Neural Networks and Deep Learning*. Coursera. Accedido el 25 de septiembre de 2024. [En línea]. Disponible: https://www.coursera.org/learn/neural-networks-deep-learning

[14] S. Mudadla. *Deep Neural Networks vs Dense Neural Networks*. Medium. Accedido el 25 de septiembre de 2024. [En línea]. Disponible: https://medium.com/@sujathamudadla1213/deep-neural-networks-vs-dense-neural-networks-bad5918a5b2a

[15] D. Burrueco. *Redes Neuronales Prealimentadas | Interactive Chaos*. Home page | Interactive Chaos. Accedido el 25 de septiembre de 2024. [En línea]. Disponible: https://interactivechaos.com/es/manual/tutorial-de-deep-learning/redes-neuronales-prealimentadas

[16] V. Choubey. *Activation Functions in Neural Network: Steps and Implementation*. Medium. Accedido el 25 de septiembre de 2024. [En línea]. Disponible: https://medium.com/codex/activation-functions-in-neural-network-steps-and-implementation-df2e4c858c21#:~:text=Softmax%20activation%20is%20ideal%20for,suitable%20for%20use%20in%20backpropagation.

[17] K. Deshie. *How to Choose the Right Activation Function for Your Neural Network*. Medium. Accedido el 25 de septiembre de 2024. [En línea]. Disponible: https://medium.com/@gamelover9899/how-to-choose-the-right-activation-function-for-your-neural-network-843d8dc85806

[18] Wikimedia contributors. *Multilayer perceptron - Wikipedia*. Wikipedia, the free encyclopedia. Accedido el 25 de septiembre de 2024. [En línea]. Disponible: https://en.wikipedia.org/wiki/Multilayer_perceptron#:~:text=If%20a%20multilayer%20perceptron%20has,-layer%20input-output%20model.

[19] A. Torres. *Descenso de gradiente: ejemplo de algoritmo de aprendizaje automático*. freeCodeCamp.org. Accedido el 27 de septiembre de 2024. [En línea]. Disponible: https://www.freecodecamp.org/espanol/news/descenso-de-gradiente-ejemplo-de-algoritmo-de-aprendizaje-automaticod/

[20] D. B. Abril. *Mínimos cuadrados • Machine Learning Visual*. La Máquina Oráculo. Accedido el 27 de septiembre de 2024. [En línea]. Disponible: https://lamaquinaoraculo.com/machine-learning/regresion-lineal/minimos-cuadrados/

[21] M. Khan. *Optimizers in Deep Learning*. Medium. Accedido el 27 de septiembre de 2024. [En línea]. Disponible: https://musstafa0804.medium.com/optimizers-in-deep-learning-7bf81fed78a0

[22] R. Egele, F. Mohr, T. Viering, P. Balaprakash. *The unreasonable effectiveness of early discarding after one epoch in neural network hyperparameter optimization*, *Neurocomputing*, vol. 597, núm. 127964, p. 127964, 2024. Accedido el 13 de enero de 2025. [En línea]. Disponible: https://arxiv.org/html/2404.04111v1

[23] K. Diamantaras, W. Duch, L. S. Iliadis. *Artificial neural networks - ICANN 2010: 20Th international conference, Thessaloniki, Greece, September 15-18, 2010, proceedings, part III*. Berlín, Alemania: Springer, 2010.

[24] IBM. *What is overfitting? Ibm.com*, 19-dic-2024. Accedido el 13 de enero de 2025. [En línea]. Disponible: https://www.ibm.com/think/topics/overfitting

[25] J. A. Rodrigo. *Redes neuronales con R, Cienciadedatos.net*. Accedido el 13 de enero de 2025. [En línea].
Disponible: https://cienciadedatos.net/documentos/68-redes-neuronales-r.html

[26] JSSST University. *Unit 2 - Deep Neural Network*. Old Site JSS STU. Accedido el 27 de septiembre de 2024. [En línea]. Disponible: https://oldsite.jssstuniv.in/wp-content/uploads/2022/11/2.-Deep-Neural-Network.pdf

[27] N. Malingan. *Optimizers in Deep Learning - Scaler Topics*. Scaler Topics. Accedido el 27 de septiembre de 2024. [En línea]. Disponible: https://www.scaler.com/topics/deep-learning/optimizers-in-deep-learning/

[28] G. Hinton, N. Srivastava, K. Swersky. *Lecture 6a: Overview of mini-batch gradient descent*" Coursera Lecture Slides, 2012. Accedido el 14 de enero de 2025. [En línea].
Disponible: https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf

[29] M. Molina. Lecture slides of the Course *Machine Learning II* - Section 1.2. 2024. Graduate Degree in Computer Science and Artificial Intelligence. Department of Artificial Intelligence. Universidad Politécnica de Madrid.

[30] S. Ruder. *An overview of gradient descent optimization algorithms*, 2016. Accedido el 14 de enero de 2025. [En línea]. Disponible: https://doi.org/10.48550/arXiv.1609.04747

[31] D.P. Kingma, J. Ba. Adam. *A method for stochastic optimization*, 2014. Accedido el 15 de enero de 2025. [En línea]. Disponible: https://doi.org/10.48550/arXiv.1412.6980

[32] S. R. Labhsetwar, S. Haridas, R. Panmand, R. Deshpande, P.A. Kolte, S. Pati. *Performance analysis of optimizers for plant disease classification with convolutional neural networks*, *4th Biennial International Conference on Nascent Technologies in Engineering (ICNTE)*, 2021. Accedido el 15 de enero de 2025. [En línea]. Disponible:
https://doi.org/10.1109/ICNTE51185.2021.9487698

[30 Z. Zhu, H. Sun, C. Zhang. *Effectiveness of optimization algorithms in deep image classification*, 2021. Accedido el 15 de enero de 2025. [En línea]. Disponible:
https://doi.org/10.48550/arXiv.2110.01598

[34] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, y B. Recht, *The marginal value of adaptive gradient methods in machine learning*, 2017. Accedido el 15 de enero de 2025. [En línea]. Disponible:
https://doi.org/10.48550/arXiv.1705.08292

[35] P. Doornbusch. *The Music of CSIRAC: Australia's First Computer Music*. Common Ground, 2005.

[36] University of Melbourne. *The music played by CSIRAC*. School of Computing and Information Systems. Accedido el 3 de octubre de 2024. [En línea]. Disponible: https://cis.unimelb.edu.au/about/csirac/music/music-played

[37] P. Westergaard, L. A. Hiller, L. M. Isaacson. *Experimental Music. Composition with an Electronic Computer*, *J. Music Theory*, vol. 3, n.º 2, p. 302, noviembre de 1959. Accedido el 3 de octubre de 2024. [En línea]. Disponible: https://doi.org/10.2307/842857

[38] C. Roads, M. Mathews. *Interview with Max Mathews*, *Comput. Music J.*, vol. 4, n.º 4, p. 15, 1980. Accedido el 4 de octubre de 2024. [En línea]. Disponible: https://doi.org/10.2307/3679463

[39] M. Battier, J. Chadabe. *Electric Sound: The Past and Promise of Electronic Music*, *Leonardo Music J.*, vol. 7, p. 100, 1997. Accedido el 4 de octubre de 2024. [En línea]. Disponible: https://doi.org/10.2307/1513256

[40] Myriad Software, *Harmony Assistant* Myriad Software. Accedido el 29 de enero de 2025. [En línea]. Disponible: https://www.myriad-online.com/en/products/harmony.htm.

[41] MakeMusic. *Finale: Music Notation Software* MakeMusic. Accedido el 29 de enero de 2025. [En línea]. Disponible: https://www.finalemusic.com/

[42] MuseScore BV. *MuseScore: Free Music Composition & Notation Software*, MuseScore. Accedido el 29 de enero de 2025. [En línea]. Disponible: https://musescore.org/.

[43] AIVA Technologies. *AIVA: Artificial Intelligence Virtual Artist*, AIVA. Accedido el 29 de enero de 2025. [En línea]. Disponible: https://www.aiva.ai/.

[44] Magenta Team. *Magenta Studio: Tools for Music Generation with Machine Learning*, Magenta. Accedido el 29 de enero de 2025. [En línea]. Disponible: https://magenta.tensorflow.org/studio.

[45] OpenAI. *MuseNet: A Deep Neural Network for Music Generation*, OpenAI. Accedido el 29 de enero de 2025. [En línea]. Disponible: https://openai.com/research/musenet.

[46] J. Martorell *¿Qué es el MIDI?: La guía del principiante para la herramienta musical más poderosa.* LANDR Blog. Accedido el 14 de octubre de 2024. [En línea]. Disponible: https://blog.landr.com/es/que-es-el-midi-la-guia-del-principiante-para-la-herramienta-musical-mas-poderosa/

[47] M. Ledesma *¿Que es el MIDI y cuál es su aplicación en la Composición Musical?* Componiendo mi Vida. Accedido el 14 de octubre de 2024. [En línea]. Disponible: https://www.componiendomivida.com/que-es-el-midi/

[48] J. Cubides. *Octavas y sus números MIDI.* Curso de programación y música. Accedido el 14 de octubre de 2024. [En línea]. Disponible: https://music-and-programation-course.readthedocs.io/es/latest/chuck/basic/octavesAndMIDINumbers.html

[49] *Free Midi - Best Free High Quality Midi Site*. Free Midi - Best Free High Quality Midi Site. Accedido el 30 de octubre de 2024. [En línea]. Disponible: https://freemidi.org/

[50] C. Devadharshini, S. Nishanth, R. Hemavathi, *Music generation using neural network*, *Interantional J. Scientific Res. Eng. Manage.*, vol. 08, n.º 05, pp. 1–5, mayo de 2024. Accedido el 30 de octubre de 2024. [En línea]. Disponible: https://doi.org/10.55041/ijsrem33813

[51] A. Ranjan, V. N. J. Behera, M. Reza. *Using a bi-directional LSTM model with attention mechanism trained on MIDI data for generating unique music.* arXiv

[cs.SD]. Accedido el 30 de octubre de 2024. [En línea]. Disponible: https://arxiv.org/abs/2011.00773

[52] C. O. D. Kamp, D. Hunter. *History of Intellectual Property in 50 Objects*. Cambridge Univ. Press, 2019. Accedido el 2 de noviembre de 2024. [En línea]. Disponible: https://doi.org/10.1017/9781108325806

[53] J. P. Briot, G. Hadjeres, F. D. Pachet. *Deep Learning Techniques for Music Generation - A survey*, 2017. Accedido el 2 de noviembre de 2024. [En línea]. Disponible: https://arxiv.org/abs/1709.01620

[54] L. Atassi. *Generating symbolic music using diffusion models*, 2023. Accedido el 2 de noviembre de 2024. [En línea]. Disponible: https://arxiv.org/abs/2303.08385

[55] N. Boulanger-Lewandowski, Y. Bengio, P. Vincent. *Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription*, 2012. Accedido el 2 de noviembre de 2024. [En línea]. Disponible: https://arxiv.org/abs/1206.6392

[56] H. Cardot. *Recurrent neural networks for temporal data processing*. InTech, 2011.

[57] A. I. Miller. *17 Project Magenta: AI Creates Its Own Music,* in *The Artist in the Machine: The World of AI-Powered Creativity* , MIT Press, 2019, pp.137-144. Accedido el 4 de febrero de 2025. [En línea]. Disponible: https://doi.org/10.7551/mitpress/11585.001.0001

[58] Z. Yi, K. K. Tan. *Convergence Analysis of Recurrent Neural Networks*, vol. 13. Boston, MA: Springer, 2004, ch. 2, pp. 15–45. Accedido el 10 de febrero de 2025. [En línea]. Disponible: https://doi.org/10.1007/978-1-4757-3819-3_2

[59] A. Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 3rd Edition. O'Reilly Media, 2022.

[60] J. Hession. *Recurrent neural networks and LSTM*, *Medium*, 19-feb-2024. [En línea]. Accedido el 26 de noviembre de 2024. [En línea]. Disponible: https://medium.com/@hession520/recurrent-neural-networks-and-lstm-2b15ce45d4bf

[61] E. Wisam. *Why do RNNs have Short-term Memory?* Towards Data Science, 20-sep-2022. Accedido el 27 de noviembre de 2024. [En línea]. Disponible en: https://towardsdatascience.com/a-true-story-of-a-gradient-that-vanished-in-an-rnn-56437c1eea45

[62] J. F. Kolen, S. C. Kremer. *Gradient Flow in Recurrent Nets: the Difficulty of Learning Long-Term Dependencies*. Wiley-IEEE Press, 2001.

[63] S. Hochreiter, J. Schmidhuber. *Long short-term memory*, *Neural Comput.*, vol. 9, núm. 8, pp. 1735–1780, 1997. Accedido el 2 de diciembre de 2024. [En línea]. Disponible: https://doi.org/10.1162/neco.1997.9.8.1735

[64] M. Conner, L. Gral, K. Adams, D. Hunger, R. Strelow, A. Neuwirth. *Music Generation Using an LSTM*, 2022. Accedido el 4 de diciembre de 2024. [En línea]. Disponible: https://doi.org/10.48550/arXiv.2203.12105

[65] H. Rafraf. *Automated Music Generation Using LSTM Networks with Representation Based on Melodic and Harmonic Intervals*. 2021. Accedido el 5 de diciembre de 2024. [En línea]. Disponible: https://doi.org/10.48550/arXiv.2108.10449

[66] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio. *Learning phrase representations using RNN encoder–decoder for statistical machine translation*, en *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014. Accedido el 9 de diciembre de 2024. [En línea]. Disponible: https://doi.org/10.48550/arXiv.1406.1078

[67] A. Nama. *Understanding Gated Recurrent Unit (GRU) in deep learning*, *Medium*, 04-may-2023. Accedido el 9 de diciembre de 2024. [En línea]. Disponible: https://medium.com/@anishnama20/understanding-gated-recurrent-unit-gru-in-deep-learning-2e54923f3e2

[68] A. A. S. Gunawan, A. P. Iman, D. Suhartono, *Automatic music generator using recurrent neural network*, *Int. J. Comput. Intell. Syst.*, vol. 13, núm. 1, p. 645, 2020. Accedido el 10 de diciembre de 2024. [En línea]. Disponible: https://www.atlantis-press.com/journals/ijcis/125941516

[69] S. S. Patil, S. H. Patil, A. M. Pawar, R. Shandilya, A. K. Kadam, R. B. Jadhav. *Music Generation Using RNN-LSTM with GRU*, en *2023 International Conference on Integration of Computational Intelligent System (ICICIS)*, 2023. Accedido el 11 de diciembre de 2024. [En línea]. Disponible: https://ieeexplore.ieee.org/document/10430293

[70] D. Foster. *Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play*. O'Reilly Media, Inc., 2023.

[71] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio. *Generative Adversarial Networks*, 2014. Accedido el 7 de noviembre de 2024. [En línea]. Disponible: https://arxiv.org/abs/1406.2661

[72] P. Luc, C. Couprie, S. Chintala, J. Verbeek. *Semantic Segmentation using Adversarial Networks*, 2016. Accedido el 7 de noviembre de 2024. [En línea]. Disponible: https://arxiv.org/abs/1611.08408

[73] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta. *Photo-realistic single image super-resolution using a generative adversarial network*, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[74] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, I. Sutskever. *Zero-shot text-to-image generation*, 2021. Accedido el 7 de noviembre de 2024. [En línea]. Disponible: https://arxiv.org/abs/2102.12092

[75] H. W. Dong, W. Y. Hsiao, L. C. Yang, Y. H. Yang. *MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment*, *Proc. AAAI Conf. Artif. Intell.*, vol. 32, n.º 1, abril de 2018. Accedido el 7 de noviembre de 2024. [En línea]. Disponible: https://doi.org/10.1609/aaai.v32i1.11312

[76] J. Engel, K. K. Agrawal, S. Chen, I. Gulrajani, C. Donahue, A. Roberts. *GANSynth: Adversarial Neural Audio Synthesis*, 2019. Accedido el 7 de noviembre de 2024. [En línea]. Disponible: https://arxiv.org/abs/1902.08710

[77] A. Brock, J. Donahue, K. Simonyan. *Large scale GAN training for high fidelity natural image synthesis*, 2018. Accedido el 29 de enero de 2025. [En línea]. Disponible: https://arxiv.org/abs/1809.11096.

[78] J. Yoon, D. Jarrett, M. van der Schaar. *Time-series generative adversarial networks*, en *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, pp. 5508-5518, 2019. Accedido el: 29 de enero de 2025. [En línea]. Disponible: https://papers.nips.cc/paper/2019/hash/c9efe5f26cd17ba6216bbe2a7d26d490-Abstract.html.

[79] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin. *Attention is all you need*, 2017. Accedido el 8 de noviembre de 2024. [En línea]. Disponible: https://arxiv.org/html/1706.03762v7

[80] S. Wang, M. Jiang, J. Liu, Y. Liu. *Vector Databases and Vector Embeddings-Review*, *2023 IEEE International Conference on Big Data (Big Data)*, Seattle, WA, USA, 2023, pp. 1234-1243. Accedido el 4 de febrero de 2025. [En línea]. Disponible: https://doi.org/10.1109/IWAIIP58158.2023.10462847

[81] Y. Fernández. *Guía de inteligencia artificial: principales características de los principales modelos de IA, puntos a favor y en contra, y comparativa,* Xataka.com, 01-mar-2025. Accedido el 10 de febrero de 2025. [En línea]. Disponible en: https://www.xataka.com/basics/guia-inteligencia-artificial-principales-caracteristicas-principales-modelos-ia-puntos-a-favor-comparativa.

[82] O. Sanseviero, P. Cuenca, A. Passos, J. Whitaker. *Hands-On Generative AI with Transformers and Diffusion Models*. O'Reilly Media, Inc., 2024.

[83] A. Pal, S. Saha, R. Anita. *Musenet: Music Generation using Abstractive and Generative Methods*, International Journal of Innovative Technology and Exploring Engineering, vol. 9, no. 6, pp. 784–788, Apr. 2020. Accedido el 8 de noviembre de 2024. [En línea]. Disponible: https://doi.org/10.35940/ijitee.f3580.049620

[84] Y. Bengio, A. Courville, I. Goodfellow. *Deep Learning*. MIT Press, 2016.

[85] Y. LeCun. *Deep learning & convolutional networks*, *2015 IEEE Hot Chips 27 Symposium (HCS)*, Cupertino, CA, USA, 2015, pp. 1-9. Accedido el 10 de febrero de 2025. [En línea]. Disponible: https://doi.org/10.1109/HOTCHIPS.2015.7477328

[86] V. Dumoulin, F. Visin. *A guide to convolution arithmetic for deep learning*, 2016. Accedido el 11 de noviembre de 2024. [En línea]. Disponible: https://doi.org/10.48550/arXiv.1603.07285

[87] M. E. Tenekeci, S. T. Abdulazeez, K. Karadağ, M. Modanli. *Edge detection using the Prewitt operator with fractional order telegraph partial differential equations (PreFOTPDE*), *Multimed. Tools Appl.*, 2024.Accedido el 4 de febrero de 2025. [En línea]. Disponible: https://doi.org/10.1007/s11042-024-19440-0

[88] M. Jogin, Mohana, M. S. Madhulika, G. D. Divya, R. K. Meghana, S. Apoorva. *Feature extraction using convolution neural networks (CNN) and deep learning*, *2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, 2018 pp. 2319-2323. Accedido el 11 de noviembre de 2024. [En línea]. Disponible: https://ieeexplore.ieee.org/document/9012507

[89] D. Scherer, A. Müller, S. Behnke. *Evaluation of pooling operations in convolutional architectures for object recognition*, *Artificial Neural Networks – ICANN 2010*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 92–101. Accedido el 11 de noviembre de 2024. [En línea]. Disponible: https://www.ais.uni-bonn.de/papers/icann2010_maxpool.pdf

[90] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, K. Kavukcuoglu. *WaveNet: A generative model for raw audio*, 2016. Accedido el 22 de noviembre de 2024. [En línea]. Disponible: https://arxiv.org/abs/1609.03499

[91] J. Engel, C. Resnick, A. Roberts, S. Dieleman, D. Eck, K. Simonyan, M. Norouzi. *Neural audio synthesis of musical notes with WaveNet autoencoders*, 2017. Accedido el 4 de febrero de 2025. [En línea]. Disponible: https://doi.org/10.48550/arXiv.1704.01279

[92] C. Hawthorne, A. Stasyuk, A. Roberts, I. Simon, C. A. Huang, S. Dieleman, E. Elsen, J. Engel, D. Eck. *Enabling factorized piano music modeling and generation with the MAESTRO dataset*, 2018. Accedido el 4 de febrero de 2025. [En línea]. Disponible: https://doi.org/10.48550/arXiv.1810.12247

[93] M. J. Gamez, *Objetivos y metas de desarrollo sostenible*, *Desarrollo Sostenible*, 2015. Accedido el 31 de marzo de 2025. [En línea]. Disponible: https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/

[94] C. Flick, K. Worrall. *The Ethics of Creative AI*, in Springer series on cultural computing, 2022, pp. 73–91. Accedido el 23 de abril de 2025. [En línea]. Disponible: https://doi.org/10.1007/978-3-031-10960-7_5

[95] R. Páez. *'Lost Tapes of the 27 Club', La inteligencia artificial y la salud mental en la música*, 2021. Accedido el 23 de abril de 2025. [En línea]. Disponible: https://www.diffusionmagazine.com/index.php/nosotros/noticias/504-lost-tapes-of-the-27-club-la-inteligencia-artificial-y-la-salud-mental-en-la-musica

[96] A. Shujiro. *General understanding on AI and copyright in Japan*, *APAA e-Newsletter*, no. 43, Oct. 2024. Accedido el 23 de abril de 2025. [En línea]. Disponible: https://apaaonline.org/article/general-understanding-on-ai-and-copyright-in-japan/

10 Anexo

Se incluye el resumen del informe de originalidad generado por la herramienta Turnitin.

RUBEN SANCHEZ FERNANDEZ

Memoria_Final_TFG_RubenSanchezFernandez_GCDIA.pdf

Turnitin Memoria Final

TFG ETSIINF (Moodle PP)

Universidad Politecnica de Madrid

Detalles del documento

Identificador de la entrega

trn:oid:::1:3267603236

Fecha de entrega

3 jun 2025, 11:50 a.m. GMT+2

Fecha de descarga

3 jun 2025, 11:57 a.m. GMT+2

Nombre de archivo

8746_RUBEN_SANCHEZ_FERNANDEZ_Memoria_Final_TFG_RubenSanchezFernandez_GCDIA_83714_....pdf

Tamaño de archivo

2.4 MB

111 Páginas

37.829 Palabras

210.854 Caracteres

2% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text

Top Sources

0% Internet sources
0% Publications
2% Submitted works (Student Papers)

Top Sources

0% Internet sources
0% Publications
2% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1 Student papers
Universidad Politécnica de Madrid <1%

2 Student papers
Universidad Internacional de la Rioja <1%

3 Student papers
UNIBA <1%

4 Student papers
Universidad Carlos III de Madrid <1%

5 Student papers
Universidad TecMilenio <1%

6 Student papers
CORPORACIÓN UNIVERSITARIA IBEROAMERICANA <1%

7 Student papers
tec <1%

8 Student papers
Universidad de Cantabria <1%

9 Student papers
Universidad de Guadalajara <1%

10 Student papers
ITESM: Instituto Tecnologico y de Estudios Superiores de Monterrey <1%

11 Student papers
Universidad Francisco de Vitoria <1%

12 Student papers

Universidad de Alcalá <1%

13 Student papers

Lappeenrannan teknillinen yliopisto <1%

14 Student papers

Universitat Politècnica de València <1%

15 Student papers

University of Birmingham <1%

16 Student papers

University of Sussex <1%

17 Student papers

Consorcio CIXUG <1%

18 Student papers

Les Roches Marbella <1%

19 Student papers

Universidad Católica San Pablo <1%

20 Student papers

Universidad Autónoma de Bucaramanga,UNAB <1%

Este documento esta firmado por

	**Firmante**	CN=tfgm.fi.upm.es, OU=CCFI, O=ETS Ingenieros Informaticos - UPM, C=ES
	Fecha/Hora	Wed Jun 04 09:52:00 CEST 2025
	Emisor del Certificado	EMAILADDRESS=camanager@etsiinf.upm.es, CN=CA ETS Ingenieros Informaticos, O=ETS Ingenieros Informaticos - UPM, C=ES
	Numero de Serie	561
	Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)