Diplomado Inteligencia Artificial y Ciencia de Datos: Métodos fundamentales, PrograLógica, Cieort - AlexNet Challencia de Computación e Ingeniería de Computación.

Reporte: Audio Style Transfer

Students: García Martínez Zoé Ariel, Bernal Rodriguez Carolina
E-mails: zoe.garcma@gmail.com, cro971206.brdz@gmail.com

TL;DR: En este proyecto se hizo uso de arquitecturas de modelos de machine learning como VGG19 y Encoder-Decoder para hacer transferencia de estilo de Audio (Audio Style Transfer).

Aspectos Generales

- Se usó Colab Pro para correr todos los modelos
- Todos los modelos fueron corridos en GPU (A100)
- Las canciones que usamos fueron un sample de:
    - Bohemian Rhapsody como contenido.
    - La Bruja como estilo.
- Los audios usados en las arquitecturas tienen 45s de duración y su contenido es simple, pues solo contiene muestras de instrumentos aislados de las canciones.
- Las muestras se recolectaron desde youtube y se editaron con librosa en python para que tuvieran la misma duración.

Transferencia de estilo de Audio

Justificación

La transferencia de estilo es una de las aplicaciones más innovadoras en el campo del Machine Learning. Sin embargo, su aplicación dentro del dominio del audio no ha recibido tanta atención como lo ha hecho la parte visual de la misma. Es por eso que este proyecto tiene como propósito usar técnicas de transferencia de estilo, originalmente desarrolladas para imágenes (VGG19) y traducción automática (Encoder/Decoder), con el objetivo de hacer transferencia de audio.

El desarrollo de técnicas de **Audio Style Transfer** permite abrir nuevas posibilidades en campos como:

- **Producción musical:** facilitando la creación de mezclas híbridas de estilos sonoros de manera automática.
- **Generación de contenido multimedia:** apoyando la generación de paisajes sonoros únicos para cine, videojuegos o arte digital.
- **Conservación y restauración:** permitiendo adaptar estilos de grabaciones antiguas a formatos más modernos o generar versiones estilizadas de registros históricos.
- **Investigación en representación de datos:** mejorando la comprensión de cómo se modelan características acústicas y texturales dentro de redes neuronales profundas.

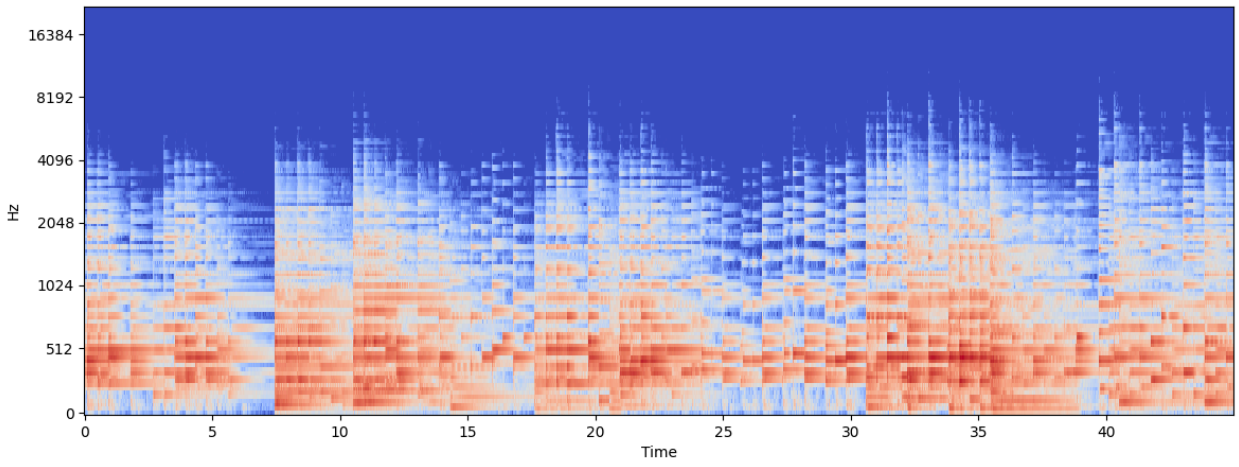
Por estas razones, se justifica la importancia de desarrollar este proyecto, no sólo por su aporte académico y técnico, sino también por su potencial para impactar distintas industrias creativas y científicas.

Contenido y Estilo

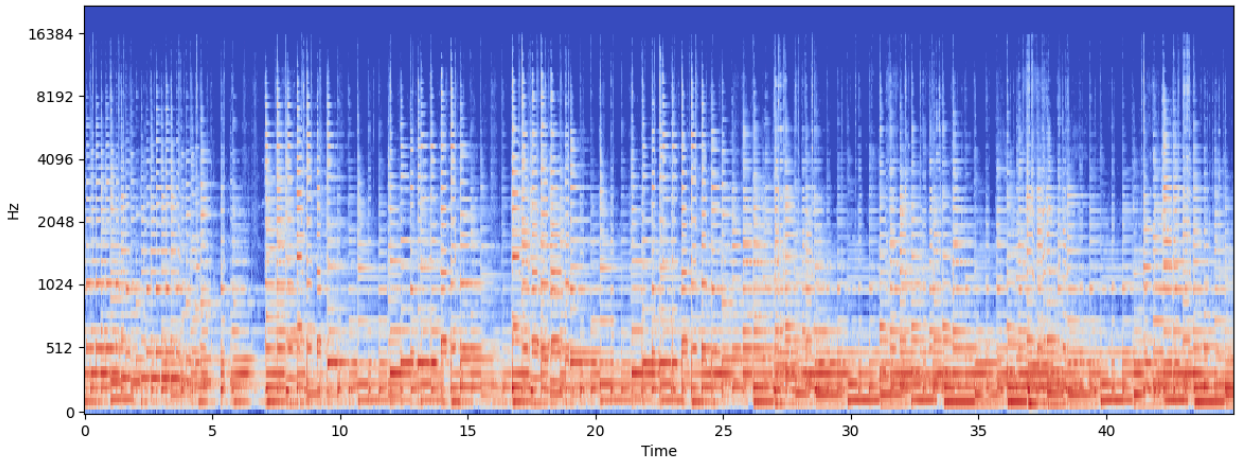
Es importante resaltar las características de la data para poder distinguir con mejor detalle el resultado gráfico del archivo generado cuando se procesa el contenido y el estilo a través de las distintas arquitecturas mostradas.

El contenido en cuestión es un sample de Bohemian Rhapsody, en piano limpio, para poder captar mejor sus características.

Mientras que el estilo es un sample de La bruja, en guitarra clásica, para poder transferir sus características de manera controlada.



IMG1: Audio del Contenido



IMG2: Audio del Estilo

Podemos notar en ambos espectrogramas que los Hz del Estilo son más puntiagudos, pues alcanzan frecuencias más altas, llegando hasta los 16K, por la naturaleza de la canción que se eligió, mientras que en el contenido se encuentra una frecuencia con naturaleza menos llegando tan solo a los 4K Hz.

Por lo que se espera que el audio generado tenga frecuencias altas como en el estilo pero una forma similar a la envolvente del espectrograma del audio de contenido.

Para muestra:

- El audio del contenido se encuentra aquí.
- El audio del estilo se encuentra aquí.

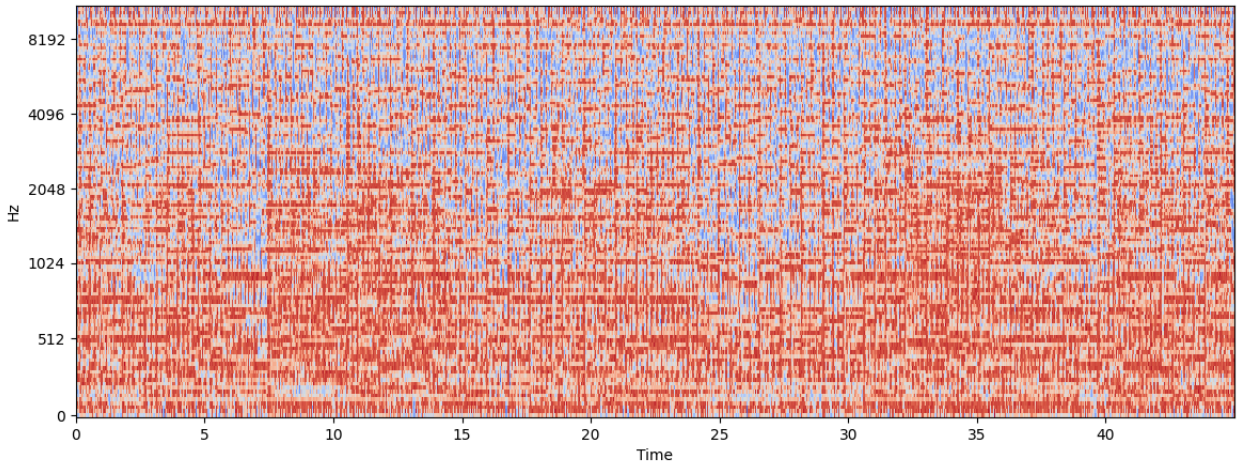
VGG19

VGG19 primer intento:

En esta parte se hizo uso del ejemplo sugerido en TF [ref 1], en este ejemplo se hace una transferencia de estilo exitosa entre imágenes, por lo que se decidió probar directamente esta arquitectura para traspasar el estilo.

Pero los resultados obtenidos con esta arquitectura dejaron mucho que desear, ya que los resultados que se obtuvieron a partir de la arquitectura se perciben de volumen bajo y con ruido añadido, lo que dificulta la escucha.

Para muestra podemos escuchar el audio aquí.



IMG3: Style Audio ruidoso

En la gráfica podemos notar una sobresaturación del contenido original hacia frecuencias altas y adhesión de frecuencias entre el máximo y los 4K Hz de la canción original. Por lo que podemos anotar que se añadieron artefactos, sin transferir correctamente el estilo.

Se corrió el algoritmo cambiando LR y modificando los pesos de estilo, pero no se percibió ninguna mejora en la calidad del audio resultante.

Al revisar la pérdida de estilo y contenido, se encontró que la pérdida de estilo estaba convergiendo a 0 de manera muy rápida, aún aumentando el peso del estilo esto no mejoró.

VGG19 segundo intento:

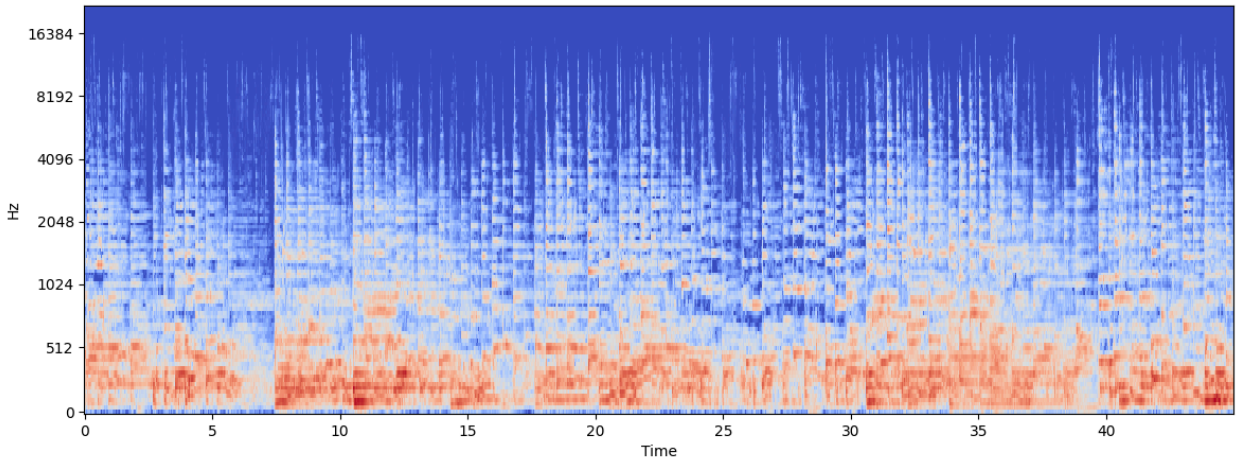
Tras notar una transferencia no exitosa con este modelo, se decidió partir con otro nuevo, ahora usando PyTorch, pues nos sentimos más cómodos usándolo con VGG19.

Usando este modelo se realizaron dos cambios principales con respecto al primer modelo:

1. Se hizo uso de capas distintas a las recomendadas en el archivo original.
2. Se realizó un cambio en el cálculo de la función de pérdida.

Con estos cambios aplicados, los resultados obtenidos fueron mejores, ya que ahora se puede notar transferencia del estilo, ya que ahora el archivo generado tiene un tono más parecido al de una guitarra y se puede distinguir el arpegio de los dedos contra las cuerdas.

Gráficamente:



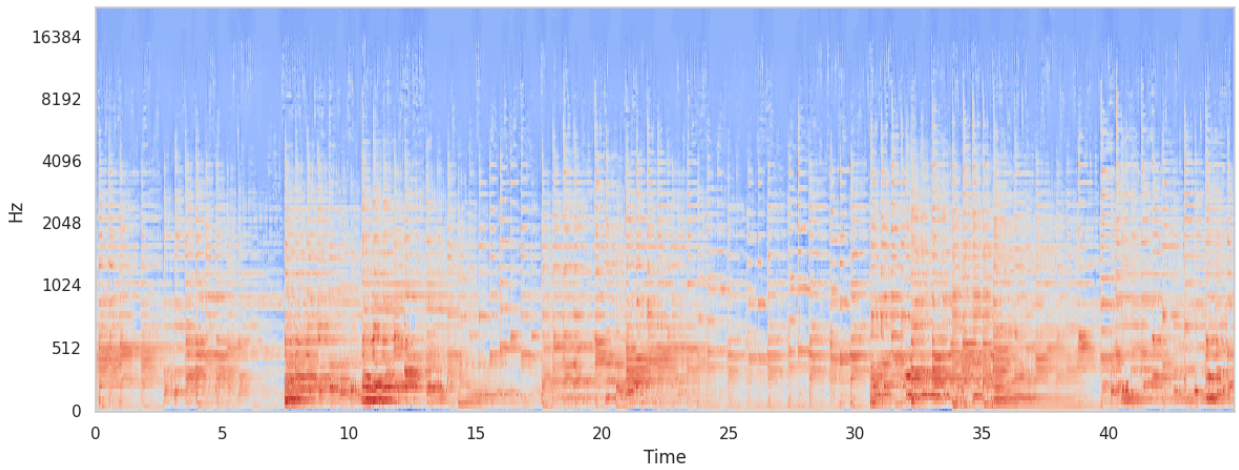
IMG3: Style Audio LR = 0.1

IMG4: Loss function LR = 0.1

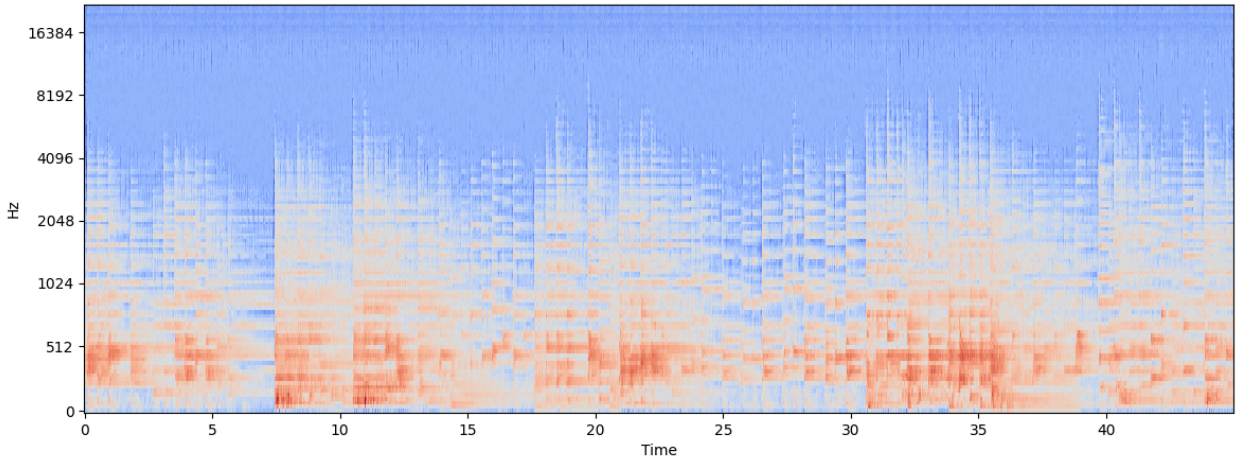
Con este algoritmo se corrieron distintas configuraciones de LR, con Weight Style = 10e8:

- LR = 0.1, el audio correspondiente aquí
- LR = 0.01, el audio correspondiente aquí
- LR = 0.001, el audio correspondiente aquí

Variando el LR se encontró que los espectrogramas se encontraban con mayor ruido con respecto al LR de 0.1, por lo que se eligió como la mejor configuración.

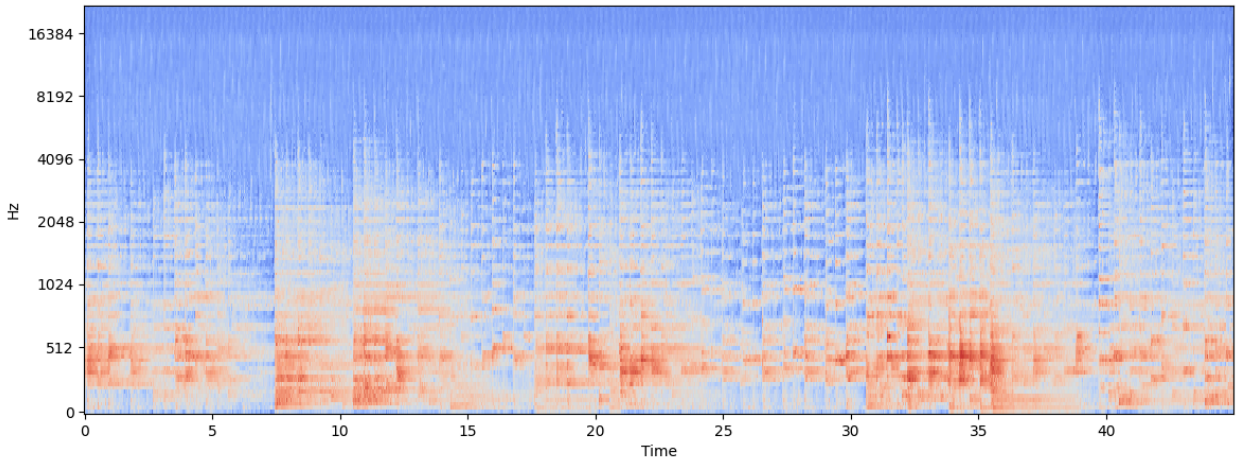


IMG5: Style Audio LR = 0.01

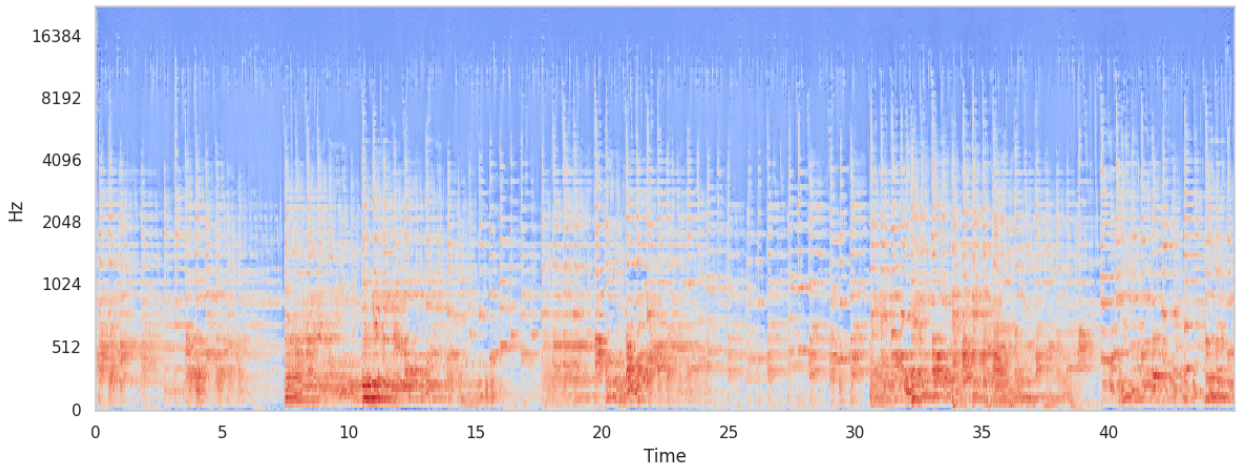


IMG6: Style Audio LR = 0.001

Cuando se configuró el Weight Style a otros valores como WS = 10e3 o WS = 10e5, se percibió un decremento en la calidad del audio y también un incremento en el ruido. Para muestra la gráfica de ambos:



IMG6: Style Audio LR = 0.1 y WS = 10e3



IMG7: Style Audio LR = 0.1 y WS = 10e5

Entre ambas versiones se puede apreciar de manera auditiva una mejora en el ruido de fondo y conservación de la calidad del contenido, mientras el peso del estilo mejora.

Para Muestra los audios:

- Audio con WS = 10e3 aquí
- Audio con WS = 10e5 aquí

Por lo que se eligió WS = 10e8 como una configuración más estable.

Encoder-Decoder

Encoder-Decoder primer intento:

En esta parte del proyecto, se implementó una primera versión del modelo Encoder-Decoder para realizar la transferencia de estilo de audio.

El procedimiento fue el siguiente:

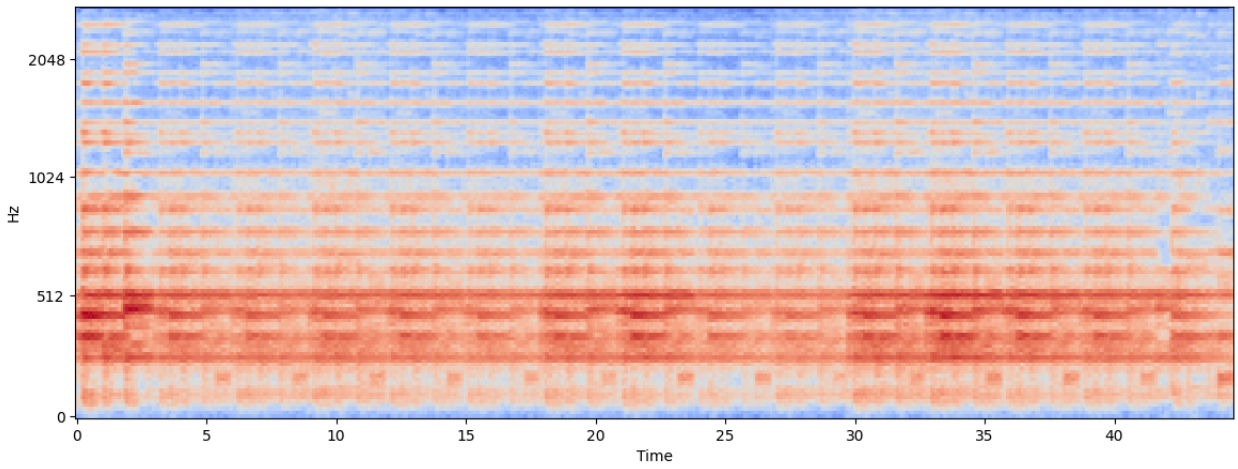
- Se utilizó una ventana temporal para dividir el audio en secciones de un tamaño definido.
- Para la reconstrucción del audio, se hizo uso de la transformada inversa de Fourier usando librosa.istft().
- La estilización del audio se aplicó sobre las ventanas generadas en la partición.
- Para el entrenamiento del encoder-decoder se configuraron **50 epochs** con un **batch size de 8**.
- La transferencia de estilo se realizó utilizando tanto las ventanas como el encoder-decoder entrenado.

La configuración utilizada para el entrenamiento fue:

- Optimizer: Adam
- Learning Rate (LR) = 0.001
- Loss function: MSE (Error Cuadrático Medio)
- Weight Style (alpha) = 0.03

Una de las principales mejoras que se encontraron con respecto al primer modelo VGG19 fue que el audio generado ya no presentaba tantos artefactos aleatorios. Sin embargo, en esta primera versión, el audio presentaba repeticiones periódicas de la misma mezcla, debido al modo en que se deslizaba la ventana temporal.

Adicionalmente, el audio no se empalmaba de manera correcta, generando discontinuidades notables entre ventanas. Esto motivó realizar cambios en la segunda versión del modelo.

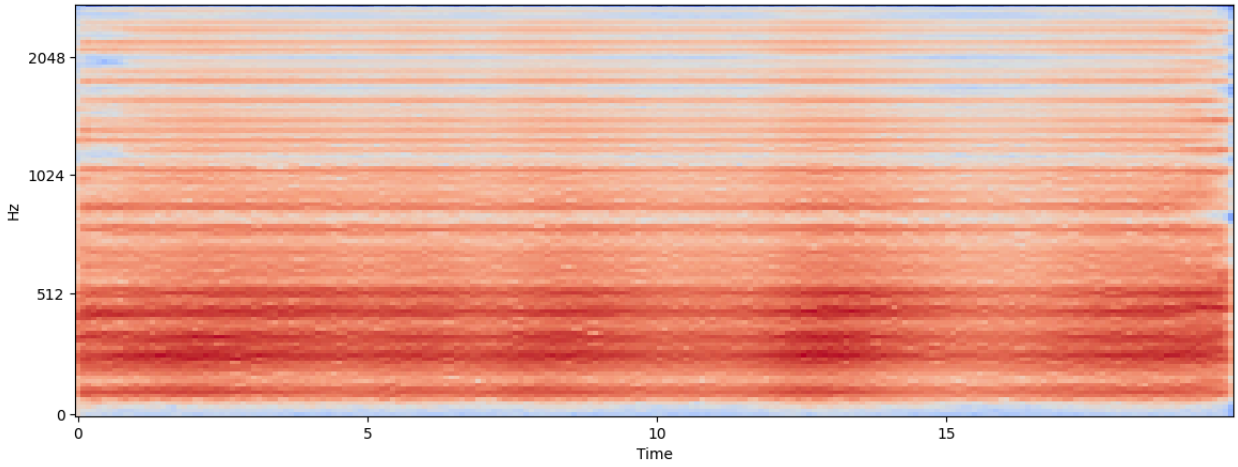


IMG8: Style Audio LR = 0.001 y WS = 0.03

Para muestra del audio generado click aquí.

Se realizó una segunda iteración con los siguientes parámetros:

- Optimizer: Adam
- Learning Rate (LR) = 0.001
- Loss function: MSE (Error Cuadrático Medio)
- Weight Style (alpha) = 0.8



IMG8: Style Audio LR = 0.001 y WS = 0.03

Para muestra del audio aquí.

Encoder-Decoder-v2

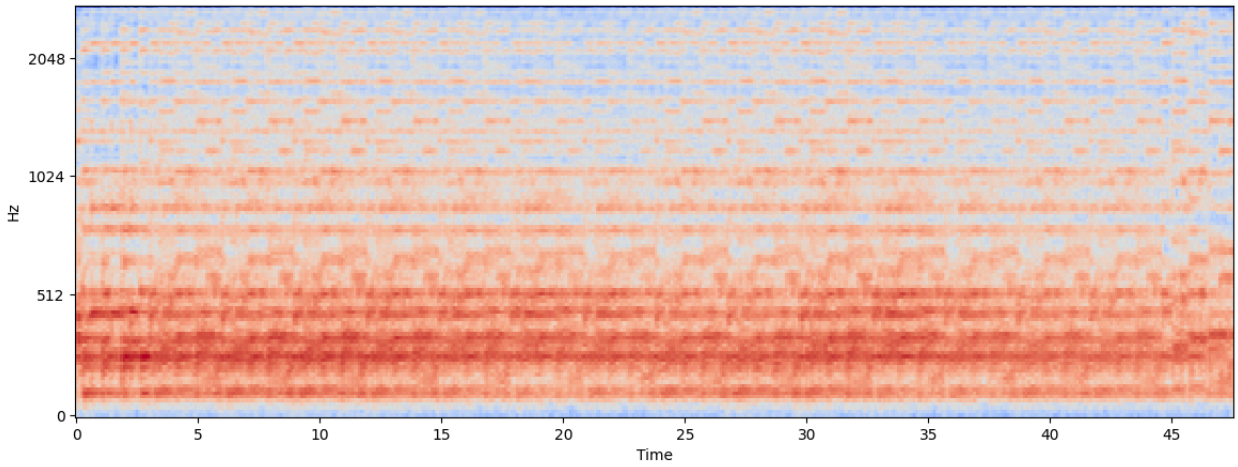
Encoder-Decoder segundo intento:

Con base en los problemas detectados en la primera versión, se realizaron los siguientes cambios para una segunda versión del modelo:

- Se implementó una **normalización** y **padding** de las ventanas, mejorando la limpieza y la configuración de los espectrogramas.
- Se rediseñó el modelo **Encoder-Decoder** agregando **más capas convolucionales** para capturar mejor las características del estilo y contenido.
- Se utilizó un enfoque basado en **VAE (Variational Autoencoder)** para generar mezclas de audio de mejor calidad.

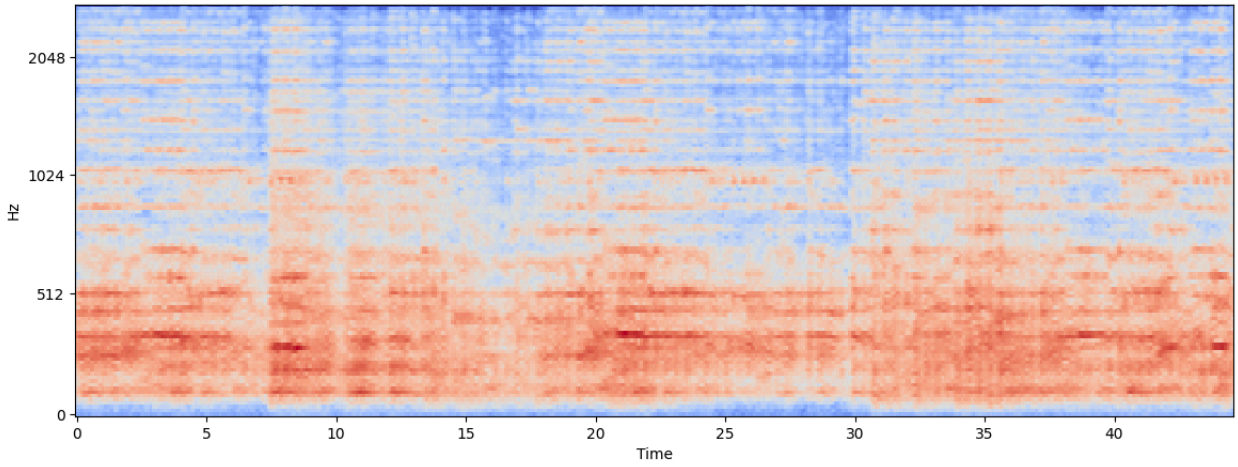
Se realizaron dos iteraciones variando el Weight Style:

En la primer iteración con WS = 0.75 y LR = 0.01



IMG9: Style Audio LR = 0.01 y WS = 0.75

Para muestra del Audio aquí.



Para muestra del audio aquí.

En ambos podemos notar una mejora significativa con respecto del primer intento, sin embargo se nota cierta repetición en el audio, lo cual no es deseado.

Pensamientos Generales acerca de la transferencia de estilo en Audio:

Un aspecto general en todos los modelos fue que la forma del loss function es fundamental para poder realizar una transferencia de estilo exitosa, en el primer intento de VGG19 se hacía un cálculo que hacía converger a cero rápidamente el loss function. En la segunda versión de VGG19, se pudo corregir esto, logrando que el loss function del estilo convergen

de manera suave a cero, junto con el Total Loss function, lo que llevó a una transferencia con mejor calidad.

Se notó que usar muestras de audio más sencillas puede mejorar significativamente la transferencia de la textura del audio, por ejemplo, en las iteraciones de VGG19, se intentó hacer transferencia de estilo con audios más complejos, pero los resultados eran inferiores en calidad. Mientras que con audios sencillos se pudo replicar la textura de la guitarra en el Audio la bruja dentro de Bohemian Rhapsody, que es un audio instrumental en piano.

Al jugar con el Weight Style se logró comprobar esto, pues al hacer que el weight style sea alto, se logra que la función de pérdida total converge de manera más suave a cero también, logrando una mejora significativa en el style transfer.

Al trabajar con Encoder/Decoder se notó que había una mejora significativa con respecto a la primera versión de VGG19, lo que se atribuye principalmente a que esta arquitectura trabaja en bloques más pequeños y que puede enfocar "atención" en segmentos más cortos de audio.

Recordemos que los algoritmos Encoder/Decoder son buenos enfocando files, por lo que usar atención implica que el algoritmo estará más enfocado en el las características clave de estilo.

Revisitando el segundo modelo, implementado VAE, se encontró que los vectores latentes usados siguen una distribución normalizada, lo que también ayuda a hacer interpolaciones más suaves, evitando ruidos extraños en el resultado.

Algo que debemos hacer notar, es que fue un reto adaptar los modelos que se tenían como muestra, ya que su última edición fue hace más de 4 años, por lo que comprenderlos y hacerlos correr también fue parte del reto.

Conclusiones

En este proyecto de **Audio Style Transfer** se exploraron distintas arquitecturas para lograr una transferencia de estilo satisfactoria entre dos muestras de audio con características bien diferenciadas.

- **Transferencia usando VGG19:**
  Se comprobó que adaptar arquitecturas diseñadas originalmente para imágenes no es trivial para el dominio del audio. La primera implementación en TensorFlow mostró limitaciones importantes como generación de ruido y pérdida de volumen. Sin embargo, al migrar a PyTorch, modificar las capas utilizadas y rediseñar la función de pérdida, se obtuvieron resultados audiblemente superiores, logrando conservar mejor el contenido original mientras se aplicaba el estilo deseado.
- **Importancia de la configuración de hiper parámetros:**
  Tanto el *Learning Rate* como el *Weight Style* demostraron tener un impacto directo

en la calidad final del audio generado. Una selección adecuada de estos parámetros permitió mejorar la estabilidad de la transferencia, reduciendo el ruido y favoreciendo la preservación de la estructura del contenido.

- **Transferencia con Encoder-Decoder:**
  La implementación de un modelo Encoder-Decoder permitió abordar los problemas detectados en VGG19, especialmente relacionados con artefactos y falta de continuidad. Aun en la primera versión del Encoder-Decoder, aunque se presentaban repeticiones y desalineaciones, la calidad general del audio ya mostraba una mejora significativa respecto a los resultados con la primera versión de VGG19.
- **Mejora usando VAE:**
  La incorporación de un **Variational Autoencoder (VAE)** en la segunda versión del modelo permitió lograr una mezcla más suave y continua entre ventanas de audio. La estructura del espacio latente normalizado facilitó generar audios estilizados de mayor calidad, sin saltos abruptos ni ruidos inconsistentes.
- **Impacto de la simplicidad del contenido:**
  Se observó que trabajar con muestras de audio sencillas (por ejemplo, un piano limpio o una guitarra aislada) favorece la transferencia de estilo, al evitar saturación de frecuencias y conflictos entre diferentes fuentes de sonido.
- **Retos enfrentados:**
  Un reto adicional fue la necesidad de adaptar y entender modelos que no habían sido actualizados en más de cuatro años, lo cual implicó tiempo de análisis, refactorización de código, y reinterpretación de algunos conceptos para ajustarlos al contexto de audio.
- **Mejora propuesta - hibridación de arquitecturas:**
  A partir de los resultados obtenidos en las distintas fases del proyecto, se identificaron varias oportunidades para mejorar la transferencia de estilo de audio combinando las ventajas observadas tanto en el modelo basado en VGG19 como en el de Encoder-Decoder con VAE.

  Se propone desarrollar una arquitectura híbrida que aproveche la extracción de características profunda de VGG19 sobre los espectrogramas, para capturar mejor las texturas del estilo, combinada con un modelo Encoder-Decoder que actúe como refinador, encargándose de reconstruir y mezclar las ventanas de audio de manera más continua y controlada.

Referencias:

1. TensorFlow. (s.f.). *Transferencia de estilo neuronal*. Consultado el 5 de abril de 2025, en: https://www.tensorflow.org/tutorials/generative/style_transfer?hl=es-419

2. Gatys, L.A., Ecker, A.S., & Bethge, M. (2015). *A Neural Algorithm of Artistic Style*. https://arxiv.org/pdf/1508.06576

3. Verma, P., & Smith, J. (2018). *Neural Style Transfer for Audio Spectrograms*. https://arxiv.org/abs/1801.01589

4. Inzva. (s.f.). *Audio Style Transfer (Repositorio GitHub)*. https://github.com/inzva/Audio-Style-Transfer

5. Grinstein, E., & Duong, Q. (2017). *Audio Style Transfer*. https://hal.science/hal-01626389/document

6. Ulyanov, D. (s.f.). *Audio Texture Synthesis and Style Transfer*. https://dmitryulyanov.github.io/audio-texture-synthesis-and-style-transfer/