¡Todo lo que tenés que saber del proyecto!

Dafne Yudcovsky

Junio 2025

1 Introducción

En el presente trabajo práctico se investiga la performance de distintos modelos pre-entrenados en separación de fuentes (source separation). Estos sistemas reciben un audio como input, que luego de ser procesados bajo algún modelo, devuelven un output también en forma de audio. En particular, el enfoque se sitúa en modelos de machine learning, en donde se entrena un modelo bajo algún algoritmo y medido con una cierta métrica, con el fin de mejorar el rendimiento de la tarea a realizar.

Para esta tarea se comparan los modelos de Open-Unmix-Pytorch, iniciativa llevada adelante por SIGSEP (que suele usarse como *base case* de la performance de los modelos); Spleeter, desarrollado por Deezer Labs (MIT support) y HTDemucs, creado por Meta (Facebook). En primer lugar, se estudian las arquitecturas y la performance SRD de cada modelo pre-entrenado. En segundo lugar, se plantean ejemplos de canciones conocidas que puedan comprobar o desafiar sus performance de alguna manera. Para esta parte, se emplea la consulta a participantes en un experimento en el que deberán puntuar los audios resultantes de los tres modelos y cada fragmento de canción según la métrica MOS. Para finalizar, se exponen los resultados de dicho experimento, sumado a una conclusión general.

2 Estado del arte

En 2015, se llevó adelante la campaña de evaluación de separación de señales (SiSEC). A partir de la misma, la comunidad científica dedicada a la separación de fuentes musicales se ha centrado principalmente en entrenar modelos supervisados para separar canciones en 4 fuentes: batería, bajo, voz y otros. El dataset de referencia fue MUSDB18, compuesto por 150 canciones en alta y moderada calidad. Su conjunto de entrenamiento contiene 87 canciones, lo cual es relativamente chico comparado con otras tareas de aprendizaje profundo, donde distintas arquitecturas de redes neuronales han sido ampliamente adoptadas.

Los métodos de MSS se reparten entre modelos basados en espectrograma y modelos basados en forma de onda. Entre los primeros se encuentran modelos como Open-Unmix (lo veremos más adelante) [**?**] o D3Net, que usa bloques convolucionales dilatados con conexiones densas. Recientemente, se ha favorecido el uso de espectrogramas complejos como entrada y salida, proporcionando una representación más interpretable. El modelo Band-Split RNN combina esta idea con múltiples RNNs de doble camino, cada una actuando en bandas de frecuencia diseñadas específicamente. Actualmente alcanza el estado del arte en MUSDB con 8.9 dB.[**?**]

Una de las preguntas principales que surge en la separación de fuentes musicales (MSS) es si el contexto es útil, o si las características acústicas locales son suficientes para ducha separación. Cuando se trata de contexto, los Transformers basados en atención han demostrado su capacidad para integrar información para secuencias de contexto largas dentro de otros campos.[**?**]

Los modelos basados en forma de onda comenzaron con Wave-U-Net, que sirvió de base para Demucs[**?**], una U-Net en el dominio temporal con una bi-LSTM entre codificador y decodificador. Otro ejemplo es Conv-TasNet que utiliza bloques convolucionales dilatados residuales (otra arquitectura parecida) [**?**]. La tendencia actual es combinar los dominios temporal y espectral (esquema híbrido), como en KUIELAB-MDX-Net o en Hybrid Demucs, que utiliza una estructura bi-U-Net con una parte compartida. Ésto permite utilizar tanto la información local como el contexto. Hybrid Demucs[**?**] fue la arquitectura mejor clasificada en la última competencia MDX MSS [**?**], aunque ya fue superada por Band-Split RNN. [1]

[1] Para artículos interesantes sobre el tema, varios blogs me mandaban a visitar [**?**], que es la IEEE Signal Processing

Las aplicaciones prácticas de estos modelos son muy variadas. En primer lugar, estos modelos permiten separar una mezcla para grabar la canción con otra percusión, por decir un ejemplo. También, permite evaluar por separado la afinación de cada una de las partes. Como las fuentes también suelen tener un procesamiento de filtrado, limpia el ruido que pueda haber quedado en la grabación.

3 Métricas de performance

El rendimiento de los modelos de separación de fuentes se evalúa mediante métricas objetivas estandarizadas, siendo las más relevantes el Signal-to-Distortion Ratio (SDR), el Signal-to-Interference Ratio (SIR) y el Signal-to-Artifacts Ratio (SAR). Estas métricas cuantifican, respectivamente, la calidad general de la señal estimada, la supresión de las fuentes no deseadas y la cantidad de nuevos artefactos (sonidos no presentes en la fuente original) introducidos durante el proceso de separación. [**?**] [**?**]

- **SDR (Signal to Distortion Ratio):** Mide la calidad general de la separación. Spleeter muestra ser competitivo, especialmente en SDR para todos los instrumentos.
- **SIR (Signal to Interference Ratio):** Evalúa la supresión de otras fuentes.
- **SAR (Signal to Artifacts Ratio):** Indica la ausencia de artefactos no deseados.
- **ISR (Source Image to Spatial distortion Ratio):** Relevante en el caso de fuentes multicanal.

En el contexto de este trabajo, por ser la que se utilizó en las competencias, tomaremos como referencia la métrica SRD de entre las anteriores. A esto le vamos a sumar la métrica **MOS** (*Mean Opinion Score*), que se detalla a continuación.

Mean Opinion Score (MOS)

El Mean Opinion Score (MOS) es una métrica utilizada en el ámbito de la Calidad de Experiencia (QoE) y la ingeniería de telecomunicaciones para estudiar el funcionamiento de un estímulo o sistema. Mide la calidad percibida de un sistema en una escala predefinida, y se define como el **promedio aritmético entre las opiniones de N sujetos**.

$$\text{MOS} = \frac{\sum_{n=1}^{N} R_n}{N}$$

Donde R_n son las calificaciones individuales y N es el número de sujetos.

Calificación	Etiqueta
1	Malo
2	Pobre
3	Regular
4	Bien
5	Excelente

Table 1: Escala de Calificación de Categoría Absoluta (ACR)

En el contexto de *source separation*, la importancia de MOS recae en la evaluación subjetiva. Ésta es crucial porque las medidas objetivas muchas veces no reflejan completamente la calidad perceptual de las señales de audio y música. Ésta métrica se utilizará en sus formatos MOS Quality (calificación de 1 a 5 de la naturalidad y ausencia de artefactos por oyentes humanos) y MOS Contamination (calificación de 1 a 5, donde 5 es cero contaminación por otras fuentes) como métricas de precisión. [**?**]

El MOS permite cuantificar la "experiencia del usuario". Si bien es costoso y requiere mucho tiempo obtener calificaciones MOS de humanos, en este campo de investigación es importante porque muchas veces se utilizan estos algoritmos con fines artísticos y estéticos, donde el oído humano y la calidad sonora es lo que más interesa cautivar, más allá de que sea una métrica relativa a cada sujeto y contexto. Al

Magazine y contiene información sobre nuevas tecnologías relacionadas.

utilizar una métrica subjetiva en cuanto al puntaje, puede haber varios factores que hagan varias bastante las puntuaciones según las personas.

Para reducir el sesgo individual, es habitual acompañar el formulario con una serie de preguntas demográficas y contextuales que permitan interpretar las puntuaciones. En este trabajo, además del puntaje MOS, se recolectarán datos sobre:

- Familiaridad con el estilo musical de la canción evaluada.
- Preferencia general por distintos géneros musicales.
- Experiencia previa con tareas de evaluación de audio (oyente casual vs. entrenado).
- Estado de atención y percepción de distracción durante la escucha.

También se incluye una breve prueba de calibración auditiva con ejemplos extremos (por ejemplo, un audio limpio y uno con fuerte distorsión) para alinear la escala perceptiva entre participantes. A los sujetos se los instruirá para juzgar la naturalidad y la contaminación entre fuentes desde una perspectiva perceptual y no técnica.

4 Open-Unmix-Pytorch

Open-Unmix (UMX) es una implementación de referencia de una red neuronal profunda para la separación de fuentes musicales. Desarrollada en Python y utilizando el framework PyTorch, se diseñó con el propósito de ofrecer un modelo entendible y reproducible para la comunidad de investigación en Music Information Retrieval (MIR). La filosofía de diseño de Open-Unmix priorizó la simplicidad y la claridad del código sobre la optimización del rendimiento, con el objetivo de servir como una línea base (*baseline*) para otras investigaciones [**?**] [**?**] [**?**]. Esta decisión facilita la comprensión y la extensión del modelo por parte de otros investigadores.

La arquitectura de este sistema recibe la señal de audio de entrada y la transforma en su espectrograma en el dominio de la frecuencia utilizando la Transformada de Fourier de Corto Tiempo (STFT). Los espectogramas capturan la información de la estructura subyacente de las fuentes musicales y presentan atributos temporales (como la variación lenta para señales sostenidas, como un *vibrato*)[**?**] que son muy útiles para la separación.

El núcleo de cada modelo de fuente de Open-Unmix es una red LSTM (Long Short-Term Memory) profunda y bidireccional de tres capas. La naturaleza recurrente de la red permite que el modelo procese señales de audio de longitud arbitraria (introduciendo *padding*), lo cual es ventajoso para manejar pistas musicales completas sin un tamaño fijo. Sin embargo, su carácter bidireccional implica que el modelo requiere "información futura" del audio para su predicción óptima, lo que le impide operar en tiempo real (tiene que ver toda la señal para poder realizar la separación).

El objetivo principal del modelo es aprender a predecir el espectrograma de magnitud de una fuente objetivo (por ejemplo, las voces) a partir del espectrograma de magnitud de la mezcla de entrada. Esta predicción se logra mediante la aplicación de una máscara sobre el espectrograma de entrada. La optimización de esta máscara se realiza en el dominio de la frecuencia, minimizando el error cuadrático medio (MSE) como función de pérdida.

Para la separación en fuentes (como voces, batería, bajo y "otros" instrumentos), Open-Unmix utiliza un componente llamado models.Separator. Este Separator agrupa los modelos individuales de espectrograma de Open-Unmix, cada uno entrenado para una fuente específica. Las salidas de estos modelos individuales se combinan mediante un filtro de Wiener generalizado multicanal [**?**]. Para poder realizar el entrenamiento en esta red, es importante que éste filtro es diferenciable y no requiere parámetros adicionales. Finalmente, las representaciones espectrales separadas se convierten de nuevo al dominio temporal utilizando la Transformada Inversa de Fourier de Corto Tiempo (iSTFT).

Los modelos pre-entrenados de Open-Unmix, como umxhq y umx, fueron entrenados principalmente en el dataset MUSDB18 [**?**], un corpus de referencia en la investigación de separación de fuentes musicales. Específicamente, umxhq se basa en la versión MUSDB18-HQ, que ofrece audio sin comprimir con un ancho

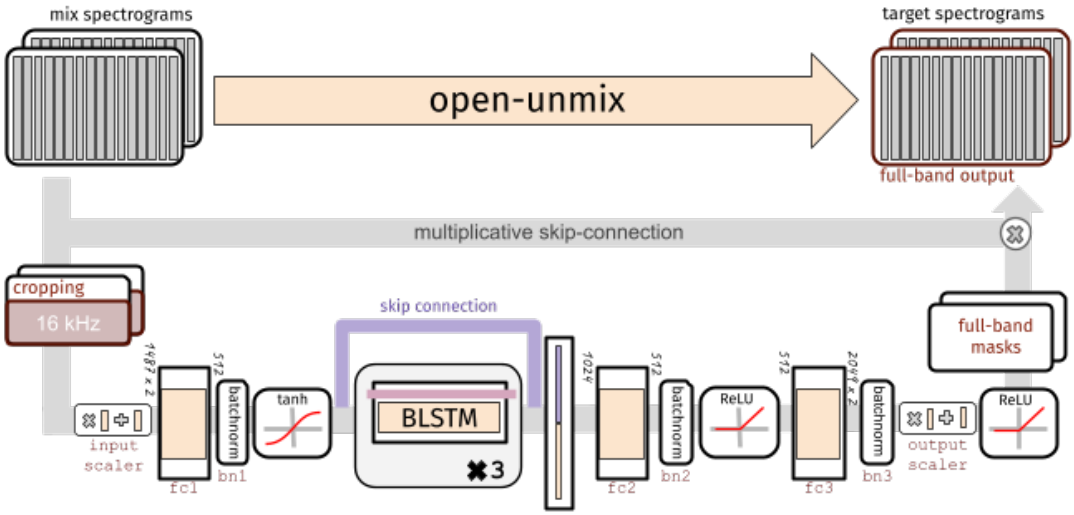

Figure 1: Caption

de banda completo de 22050 Hz. Por otro lado, umx se entrenó con la versión estándar de MUSDB18, con un ancho de banda limitado a 16 kHz debido a la compresión AAC, lo que lo hace adecuado para comparaciones con métodos más antiguos. Un modelo posterior, umxl, fue entrenado con datos adicionales, mostrando mejoras significativas en el rendimiento y la generalización.

En las campañas de evaluación como SiSEC 2018, Open-Unmix ha demostrado resultados de vanguardia, siendo muy competitivo frente a otros sistemas participantes. Por ejemplo, el modelo UMXL alcanzó un SDR mediano de 7.21 dB para la voz, 6.02 dB para el bajo, 7.15 dB para la batería y 4.89 dB para "otros" instrumentos. [**?**] [**?**]

5 Spleeter

Spleeter es una herramienta desarrollada por Deezer Research que tiene un rendimiento destacado especialmente en el ámbito de la Recuperación de Información Musical (MIR) [**?**]. Ésta área de investigación se enfoca en el desarrollo de algoritmos para analizar y entender datos musicales (información no provista en las fuentes, pero es una descripción general del campo). Es un híbrido entre un modelo de machine learning y un sistema experto, ya que utiliza conocimiento musical para el entrenamiento del modelo. Su objetivo principal es llegar a resultados de separación más fieles a la separación de fuentes que hace el cerebro humano cuando escucha varios instrumentos a la vez.

Esta herramienta está basada en Pytorch y es de código abierto [**?**]. Ofrece una librería (y una API de Python) de modelos pre-entrenados de *Deep Learning*, junto con su documentación sobre las funciones a aplicar en cada tarea. Este enfoque es *"data-driven"*, en donde se busca aprender un mapeo directo entre una mezcla de audio y las fuentes individuales o sus representaciones a partir de una base de datos.[**?**]

La arquitectura de red neuronal utilizada en los modelos pre-entrenados de Spleeter es la U-Net, una Red Neuronal Convolucional (CNN) de 12 capas (6 capas para el codificador y 6 para el decodificador) con *skip connections*. Las CNNs son eficientes para la paralelización de cálculos (multiplicación de matrices con GPU) y encontrar las *features* principales de la señal. Se utiliza una representación tiempo-frecuencia (TF) del audio, que a lo largo de la materia se la representó con un espectrograma resultante de la Transformada de Fourier de Tiempo Corto (STFT) [**?**]. En esta representación, las fuentes tienden a superponerse menos que en la forma de onda original, pudiendo aprovechar la forma de la onda para el aprendizaje y facilitando la selección de porciones de la mezcla que corresponden a una sola fuente.

El objetivo de estas redes es estimar una máscara de tiempo-frecuencia para cada fuente o, alternativamente, directamente los espectrogramas de las fuentes. Una máscara TF actúa como un ecualizador que cambia su configuración cada pocos milisegundos, aplicando una ganancia entre 0 y 1 a cada elemento del espectograma para estimar la señal deseada. Una vez que se multiplica la mezcla por la máscara, la señal separada se recupera mediante con la transformada inversa. Spleeter, de hecho, estima una *"soft*

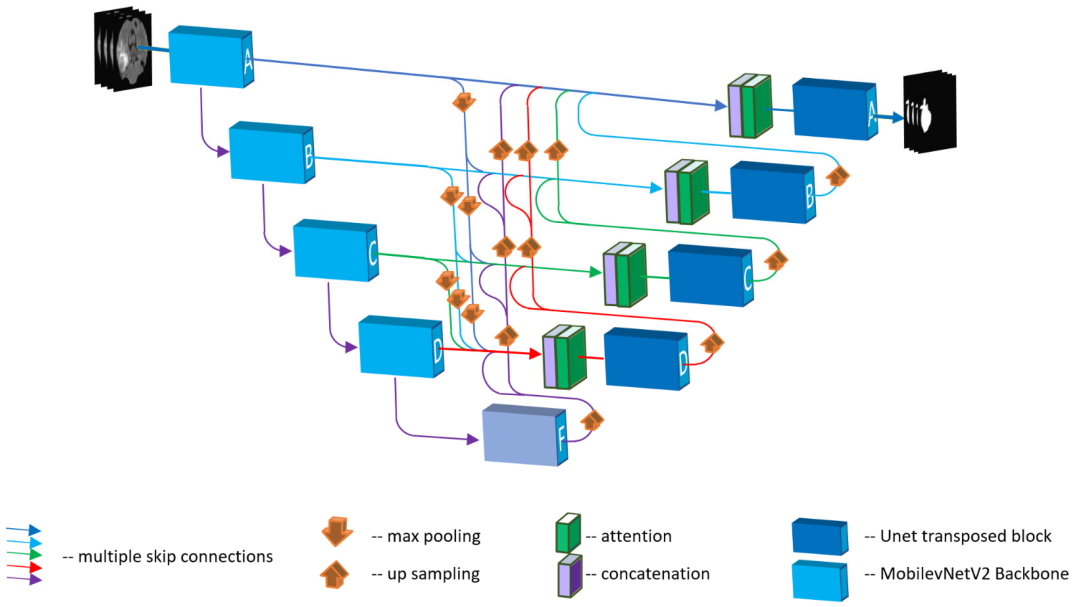

Figure 2: U-Net con *skip connections*, se eligen de manera aleatoria qué conexiones se prenden y apagan en cada capa de encoder-decoder

*mask"*para cada fuente, y la separación se realiza a partir de los espectrogramas de fuente estimados. Como post-procesamiento se puede aplicar el filtro de Wiener para refinar ésta separación.[**?**]

$$\text{Máscara}_j(f,t) = \frac{|\hat{S}_j(f,t)|}{\sum_{k=1}^{K} |\hat{S}_k(f,t)|}$$

Donde $\hat{S}_j(f,t)$ es el valor estimado del espectrograma de la fuente j en la frecuencia f y tiempo t, K es el número total de fuentes estimadas y para cada punto (f,t), las máscaras de todas las fuentes suman 1: $\sum_{j=1}^{K} \text{Máscara}_j(f,t) = 1$. Como la cantidad de máscaras es igual a la cantidad de fuentes, entonces j $=\{2,4,5\}$ en el marco de este trabajo. [**?**]

Como se mencionó, el modelo ofrece una separación en dos partes: voz y acompañamiento; cuatro partes: voz, batería, bajo y otros instrumentos; y cinco partes: igual que 4 pero distinguiendo el piano. Todos ellos con un sample_rate de 44.1 KHtz (requerido en la consigna). Según los desarrolladores, Spleeter fue el primer modelo lanzado al público capaz de realizar la separación en cinco. Además, cuando se ejecuta en una GPU, Spleeter tiene una velocidad de separación de aproximadamente dos minutos para tres horas de audio, en la configuración de cuatro partes.

Los detalles sobre el entrenamiento precisos no están publicados, aunque se sabe el dataset está compuesto por los datos **MUSDB + 25000** [**?**] canciones privadas, y que se utilizó la norma L1 como función de pérdida. Como último detalle del modelo, puede funcionar bien aunque haya una disponibilidad limitada de datos. Ésto permite poder separar exitosamente una sola canción a la vez (como es el caso de este tranajo) y tener un buen resultado.[**?**]

6 HTDemucs

HTDemucs es la cuarta versión publicada de código abierto del modelo Demucs para separación de audio desarrollado por Meta. Su nombre viene de *Hybrid Transformer Demucs* que hace alusión a su tipo de arquitectura particular, en donde se busca utilizar información tanto del espectograma como de la forma de onda original, lo que facilita la selección de porciones que corresponden a una única fuente. Es capaz de separar voces, batería, bajo y otros instrumentos. Una versión experimental de 6 fuentes puede añadir guitarra y piano, aunque se señala que la calidad del piano no es óptima actualmente. Se destaca entre otros modelos por su alta performance en términos de la métrica SRD, que mide la calidad de la separación.[**?**][**?**]

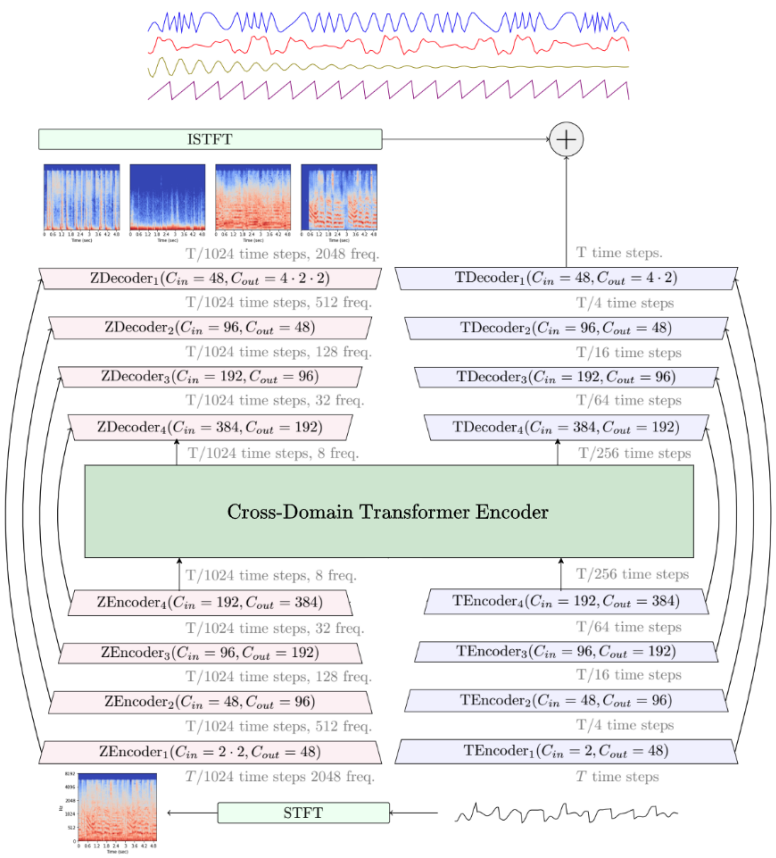

Figure 3: Arquitectura de HT Demucs con *self attention* y *cross attention*

El modelo original en el que se basa es *Hybrid Demucs*. Éste consta de dos redes U-Nets, una en el dominio temporal (con convoluciones temporales) y otra en el dominio espectral (con convoluciones sobre el eje de frecuencia)[**?**]. Cada U-Net tiene cinco capas de codificación y cinco de decodificación. Luego de la quinta capa del codificador, ambas representaciones tienen la misma forma y se suman antes de pasar a una sexta capa compartida. Similarmente, la primera capa del decodificador es compartida, y su salida se envía a ambas ramas. La salida espectral se convierte en forma de onda usando la iSTFT y se suma con la salida de la rama temporal. [**?**]

HT Demucs reemplaza las capas ocultas por un codificador -transformer entre dominios-, que aplica self-attention dentro de cada dominio y cross-attention entre ellos. Como los transformers necesitan gran cantidad de datos, se entrenó el modelo con **MUSDB + 800**[**?**] canciones adicionales seleccionadas cuidadosamente. El entrenamiento se realizó en 8 GPUs V100, durante 1200 épocas, con pérdida L1 sobre las formas de onda y optimización Adam. Se aplicaron aumentos como repitching, tempo stretch y mezcla de fuentes. Luego, se aplicó fine-tuning por fuente durante 50 épocas sin aumentos, lo cual mejoró el SDR final en 0.25 dB.

Como optimización, se incorporaron kernels de atención dispersa mediante Locally Sensitive Hashing[2] para ampliar el contexto sin exceder la memoria, alcanzando un 90% de dispersión. Esta variante se denomina Sparse HT Demucs, y con esta optimización se alcanzan 9.20 dB de SDR. Su rendimiento mejoró especialmente al aumentar el contexto de entrada y ajustar hiperparámetros como profundidad y dimensión del Transformer.

[2]Locality Sensitive Hashing (LSH) es una técnica que aproxima la *similitud* (bajo alguna métrica) entre dos puntos reduciendo la dimensión de los datos, pero manteniendo la distancia local entre los mismos.