Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA AGRONÓMICA Y BIOCIENCIAS

NEKAZARITZAKO INGENIARITZAKO ETA BIOZIENTZIETAKO GOI MAILAKO ESKOLA TEKNIKOA

SEPARACIÓN DE HABLANTES INDIVIDUALES EN ENTORNOS "COCKTAIL-PARTY" MEDIANTE REDES DE APRENDIZAJE PROFUNDO

presentado por

ANA CANTALAPIEDRA ARELLANO*(e)k*

aurkeztua

GRADO EN CIENCIAS DE DATOS
GRADUA DATUEN ZIENTZIETAN

Septiembre, 2023

Resumen

El efecto cóctel presenta un desafío significativo para las personas con discapacidad auditiva al participar en conversaciones en entornos ruidosos. Este fenómeno, que se refiere a la dificultad de separar fuentes sonoras y detectar la fuente de interés en situaciones de múltiples hablantes, ha sido objeto de investigación en el campo de la audición y la percepción auditiva.

En la última década, se han desarrollado diversos sistemas de separación de hablantes, especialmente basados en técnicas de aprendizaje profundo (deep learning), que han demostrado mejoras significativas en la separación de fuentes. En este trabajo, se empleó una base de datos de cocktail party, luego, se aplicaron los algoritmos Conv-TasNET y DPRNN para separar las fuentes de audio y se evaluaron sus capacidades de rendimiento.

Índice

1. Introducción **5**

2. Estado del arte **6**
2.1. Separación de fuentes de audio 6
2.2. Separación de fuentes de audio con aprendizaje profundo 8
2.2.1. Aprendizaje no supervisado 8
2.2.2. Aprendizaje semi-supervisado 9
2.2.3. Aprendizaje supervisado 10

3. Marco teórico **10**
3.1. Definición de audio 10
3.2. Aprendizaje profundo 11
3.2.1. Redes Convolucionales (CNN) 11
3.2.2. Redes Neuronales Recurrentes (RNN) 12
3.3. Separación de hablantes 13
3.3.1. Definición del problema 13
3.4. Modelos de separación 13
3.4.1. Arquitectura base 15
3.4.2. Conv-TasNet 18
3.4.3. Dual-Path RNN 24

4. Marco experimental **26**
4.1. Datos 26
4.2. Métricas de evaluación 28
4.2.1. Relación fuente-distorsión (SDR) 28
4.2.2. Relación señal-ruido invariante de escala (SI-SNR) 29
4.3. Problema de permutación de etiquetas 29
4.4. Configuración de parámetros 32
4.5. Proceso de entrenamiento 33

5. Resultados **35**

6. Conclusiones **40**

7. Lineas futuras **40**

Índice de figuras

1. Diagrama de la arquitectura base de las redes de separación [Luo and Mesgarani, 2018] 15
2. Diagrama de la unidad convolucional 1D [Zhang, 2021] 16
3. Arquitectura de la red de separación de audio en el dominio temporal TasNet (Luo & Mesgarani, 2018) 18
4. Diagrama de flujo de la arquitectura Conv-TasNet[Luo and Mesgarani, 2019] 21
5. Representación gráfica del campo receptivo en una operación de convolución dilatada, con variados niveles de dilatación [Cui et al., 2019] 22
6. Estructura de un bloque convolucional (1-D Conv) Arquitectura ConvTasNet [Luo and Mesgarani, 2019] 22
7. Bloque de convolución separable en profundidad S-conv [Hossain et al., 2020] 23
8. Diagrama de flujo de la arquitectura DPRNN[Luo et al., 2020] 24
9. Modelo de separación de dos hablantes convencional 30
10. Modelo de separación de dos hablantes con entrenamiento invariante de permutaciones [Yu et al., 2017] 31
11. Evolución de los valores de la función de pérdida durante el entrenamiento - Entorno no ruidoso 37
12. Evolución de los valores de la función de pérdida durante el entrenamiento - Entorno ruidoso 38
13. Evolución de los valores de la función de pérdida durante el entrenamiento - Entorno con música 39

Índice de tablas

1. Hiperparámetros arquitectura ConvTasNet 33
2. Hiperparámetros arquitectura DPRNN 33
3. Resultados de los Modelos - Entorno no ruidoso 36
4. Resultados de los modelos - Entorno ruidoso 37
5. Resultados de los modelos - Entorno con música 38

1. Introducción

En numerosas ocasiones la comunicación entre las personas se da en entornos con varios individuos y múltiples hablantes, donde las personas con discapacidad auditiva a menudo tienen dificultades al participar en conversaciones, estos entornos sociales son conocidos como cocktail party o fiesta cóctel, donde múltiples hablantes conversan al mismo tiempo. Este desafío se refiere al efécto cóctel, que describe la capacidad del oído humano para discriminar una fuente sonora específica en medio de una conjunto de ruidos de fondo en entornos acústicos complejos [Fischer et al., 2020].

Mientras que las personas con audición normal realizan esta tarea aparentemente compleja con facilidad, aquellos con deficiencias auditivas encuentran dificultades en dichas situaciones. Con el propósito de asistir a individuos que utilizan audífonos e implantes cocleares, los científicos están trabajando en la emulación del sistema auditivo humano.

En contextos caracterizados por el efecto cóctel, el cerebro humano presenta la capacidad de separar señales verbales simultaneas. No obstante, el proceso exacto de procesamiento de señales que el cerebro lleva a cabo para este fin todavía no se comprende en su totalidad.

El término 'cocktail party problem' engloba dos retos a los que el oyente se enfrenta en dicha situación: la segregación de fuentes sonoras y la detección del oyente atendido. Dicho término fue introducido por Colin Cherry [Cherry, 2005]. Cherry se centró en el componente atencional del problema al introducir el paradigma de escucha dicótica. Su propósito consistía en examinar la capacidad de los observadores para seleccionar una señal de habla ignorando el resto [Bronkhorst, 2015]. Fue en 1990 cuando Albert Bregman introdujo un nuevo término para el área de investigación de la segregación de fuentes sonoras: el Análisis de Escena Auditiva (ASA).

El Análisis de Escena Auditiva (ASA) se refiere a la capacidad del sistema auditivo humano para organizar y separar los diferentes componentes de una mezcla de sonidos. [Agrawal et al., 2023] El cerebro humano posee la capacidad de procesar todos los sonidos de un entorno mediante diferentes procesos cognitivos y neurales, logrando descomponer la mezcla de sonidos en fuentes individuales y asignar a cada componente la fuente correcta. El ASA se basa en principios de agrupación perceptual, donde el cerebro identifica patrones en las características de los sonidos, como la frecuencia, la intensidad y la duración, para agrupar los sonidos que provienen de una misma fuente y separarlos de otras fuentes [Szabó et al., 2016].

A lo largo de este trabajo, nos centraremos en el primer reto: los diferentes sonidos se entrelazan para formar la señal que finalmente llega al oído. Sin embargo, este conjunto de sonidos resulta tener un valor limitado para el organismo, el cual tiende a focalizarse en fuentes sonoras particulares de mayor relevancia. Nuestro sistema auditivo naturalmente busca identificar y distinguir estas fuentes de sonido específicas para poder procesar el entorno acústico del oyente.

Uno de los desafíos centrales de este problema es la identificación precisa de distintos interlocutores dentro de la conversación, lo que, a su vez, podría facilitar el reconocimiento del hablante al que el oyente está prestando atención. La mayoría de los estudios sobre la Decodificación de la Atención Auditiva han asumido la separación de fuentes sonoras como un elemento establecido, utilizando conjuntos de datos en los cuales las diversas fuentes de audio ya están aisladas.

Durante la última década, se han propuesto multitud de sistemas para la separación de hablantes, en inglés denominada "speech separation". Estos sistemas están diseñados para procesar audios correspondientes a una mezcla auditiva de múltiples hablantes con el objetivo de aislar las intervenciones individuales de los hablantes, tarea que para los seres humanos y en especial para el sistema auditivo es de mucha facilidad. Entre estos sistemas han destacado aquellos basados en técnicas de aprendizaje profundo o deep learning ya que han proporcionado mejoras significativas respecto a métodos más tradicionales. El objetivo del presente trabajo será probar dos modelos de aprendizaje profundo y evaluar su capacidad de separación de fuentes empleando una base de datos con señales de audio en distintos escenarios de cocktail party.

2. Estado del arte

La revisión del estado del arte que se presenta ha sido extraída del artículo titulado "A review on speech separation in cocktail party environment: challenges and approaches" [Agrawal et al., 2023]. Este artículo se centra en analizar y abordar los desafíos asociados con la separación de habla en situaciones de cocktail party.

2.1. Separación de fuentes de audio

Los métodos tradicionales para separar el habla en situaciones de un solo canal, en las que solo disponemos de un único micrófono, se centran en abordar sistemas en los cuales dicha separación está limitada por esta restricción. En consecuencia, se utilizan características perceptuales basadas en aspectos espectrales del habla, tales como la continuidad del tono,

la presencia de armonías y la identificación de patrones de ondas compartidas.

Dentro de este contexto, es posible identificar tres enfoques distintos: los métodos impulsados por la fuente (Source-driven methods), los métodos impulsados por modelos (Model-driven methods) y los métodos de fusión/combicación (Fusion/combination methods).

- **Métodos Impulsados por la Fuente (Source-driven methods):** estos métodos se centran en el análisis y la manipulación de características específicas de las fuentes de audio individuales. Utilizan conocimiento previo o características intrínsecas de las señales de audio para intentar separarlas. Pueden incluyen el uso de patrones espectrales únicos de las fuentes o información temporal para identificar y aislar las fuentes.
- **Métodos Impulsados por Modelos (Model-driven methods):** estos métodos se basan en la creación y adaptación de modelos matemáticos o estadísticos para describir cómo se mezclan las fuentes de audio. Los modelos intentan capturar las relaciones entre las señales mezcladas y, mediante el ajuste de parámetros, se busca inferir y separar las fuentes originales. Ejemplos comunes son los enfoques basados en descomposición en componentes independientes (ICA) o en modelos probabilísticos.
- **Métodos de Fusión/Combinación (Fusion/combination methods):** estos métodos buscan mejorar la separación de fuentes combinando los resultados de diferentes enfoques. Pueden combinar las salidas de métodos impulsados por la fuente y métodos impulsados por modelos para obtener una separación más precisa y robusta. La idea es que diferentes enfoques puedan complementarse entre sí para superar sus limitaciones individuales.

Por otro lado, podemos clasificar los algoritmos de separación de fuentes en función de la cantidad de información que se tiene disponible sobre las fuentes y el proceso de mezcla, obteniendo dos posibles categorías: Separación Ciega de Fuentes (BSS) y la Separación de Fuentes Informadas (ISS).[Chandna, 2016]

- **Separación Ciega de Fuentes (BSS)**: los algoritmos de BSS asumen un conocimiento mínimo sobre las fuentes, como la independencia estadística de las señales por lo que las fuentes que componen la señal son estadísticamente independientes en si en algún sentido. Este es el concepto en el que se basa este tipo de algoritmos, gracias a la independencia se identifican y separan las fuentes individuales en base a sus patrones estadísticos únicos.

  Dentro de este grupo podemos encontrar algoritmos de separación como el Análisis de Componentes Independientes (ICA), la Factorización de Matrices No Negativas (NMF), los Métodos en el dominio tiempo-frecuencia. [Chandna, 2016]

- **Separación de Fuentes Informadas (ISS)** : la principal característica de la ISS es que se basa en el conocimiento previo o en la información específica sobre las fuentes que se desean separar. Esto puede incluir información sobre las características estadísticas, espectrales o temporales de las fuentes. [Chandna, 2016]

2.2. Separación de fuentes de audio con aprendizaje profundo

Los métodos convencionales de separación de fuentes de audio en entornos de cocktail party (donde múltiples fuentes de sonido se mezclan en una grabación) se vieron limitados por varios factores.

En primer lugar, la complejidad de las señales representa un desafío para los métodos convencionales. Aquellos basados en características manuales o enfoques lineales pueden tener dificultades para lidiar con la complejidad derivada de la superposición de múltiples fuentes de sonido, lo que afecta su capacidad para separar de manera precisa las fuentes.

Por otro lado, la dependencia de características manuales es una limitación clave de los métodos convencionales. A menudo, estos métodos dependen de características diseñadas a mano que deben ser adaptadas específicamente para cada problema. Sin embargo, estas características pueden no capturar adecuadamente la variabilidad y sutilezas presentes en las señales de audio en situaciones de cóctel party.

Además, otra limitación importante está relacionada con la variabilidad en las condiciones de los entornos cocktail party. Estas condiciones pueden variar ampliamente, incluyendo el número de hablantes, el contenido de las fuentes de sonido y la relación señal-ruido en la mezcla. Los métodos convencionales pueden no ser lo suficientemente flexibles como para adaptarse eficazmente a estas variaciones. Esto llevó al surgimiento de las técnicas basadas en Aprendizaje Profundo para abordar estos desafíos.

Los algoritmos de Aprendizaje Profundo (DL) han demostrado gran éxito en abordar el desafío de la separación de voz en entornos de cóctel party, inspirados en los logros en reconocimiento de voz y separación de hablantes en entornos con múltiples locutores. La separación de habla se clasifica en tres grupos basados en el conocimiento previo disponible: no supervisado, semi-supervisado y supervisado.

2.2.1. Aprendizaje no supervisado

El aprendizaje no supervisado en la separación de fuentes de audio se refiere a separar automáticamente las diferentes fuentes de sonido presentes en la mezcla de audio sin requerir

información previa o conocimiento a priori sobre las fuentes específicas.

En este enfoque, el algoritmo analiza la mezcla de audio sin tener acceso directo a las señales originales de las fuentes. Su objetivo es identificar patrones, características y regularidades en la mezcla que le permitan separarlas. Pero la implementación de un algoritmo de separación basado en técnicas suervisada tiene una gran dificultad.

En el aprendizaje no supervisado, es difícil separar señales de habla desconocidas en situaciones de una o varias personas mezcladas en un entorno complicado con una sola señal de canal. En la situación semi-supervisada, se crea un diccionario faltante durante el proceso de separación debido a la falta de datos de habla o ruido.

2.2.2. Aprendizaje semi-supervisado

En el contexto del aprendizaje semi-supervisado, se aborda un enfoque de aprendizaje automático en el que el sistema carece de un conjunto completo de datos etiquetados, es decir, datos en los que se conoce con precisión la identidad de cada componente. Esto implica que parte de la información necesaria para entrenar el sistema está ausente o es limitada.

Dentro de este contexto, se presentan dos desafíos principales. En primer lugar, existe una escasez de datos de habla, lo que dificulta la enseñanza al sistema para reconocer y diferenciar entre la voz humana y el ruido ambiental. En segundo lugar, los datos disponibles están contaminados con ruido, que se refiere a información no deseada o perturbaciones que pueden afectar negativamente la calidad del reconocimiento de la voz.

Para abordar estos desafíos, se incorpora una etapa denominada etapa de separación y construcción de un diccionario faltante. Aquí, se genera un diccionario faltante, que constituye una colección de patrones de sonido esenciales para que el sistema pueda reconocer y separar eficazmente la voz del ruido.

Para llevar a cabo este proceso, los investigadores utilizaron la Factorización de Matrices No Negativas (NMF). En este contexto, se emplea para descomponer el sonido en sus componentes fundamentales, como la voz y el ruido, lo que facilita su posterior análisis y separación.

Se logra la generación de bases de habla estables a partir de los datos de entrenamiento, lo que implica la identificación y almacenamiento de patrones de sonido específicos relacionados con la voz. Cuando se opera en tiempo real, es decir, durante la utilización práctica del sistema, se calculan los componentes de ruido utilizando los datos actuales. Esto permite

al sistema llevar a cabo la separación de la voz y el ruido en tiempo real, contribuyendo así a una mayor eficacia en diversas aplicaciones.

2.2.3. Aprendizaje supervisado

La aplicación de técnicas de aprendizaje supervisado y aprendizaje profundo para separar el habla del ruido en señales de audio ha revolucionado la forma en que abordamos el procesamiento de audio en diversas aplicaciones. Esta técnica, conocida como Separación Supervisada del Habla (SCSS), se basa en un enfoque que combina el poder del aprendizaje automático con la capacidad de las redes neuronales profundas para mejorar la inteligibilidad del habla en condiciones adversas. En SCSS, se utilizan conjuntos de datos de entrenamiento que contienen grabaciones de audio con etiquetas que indican qué partes son habla y cuáles son ruido.

En la última década, se han logrado avances significativos en algoritmos de separación supervisada, especialmente con la adopción de técnicas de aprendizaje profundo. El objetivo principal de SCSS es generar máscaras binarias, como la Máscara de Binarios Ideales (IBM), que indican qué partes de la grabación corresponden al habla y cuáles al ruido. Esto convierte el problema en una tarea de clasificación binaria, que es una área bien establecida en el aprendizaje automático.

Además de las máscaras binarias, se han desarrollado otras máscaras y técnicas para evaluar y comparar los resultados de la separación de habla. También se han utilizado diversas arquitecturas de aprendizaje profundo, como redes neuronales convolucionales (CNN), redes neuronales recurrentes (RNN) y redes generativas adversarias (GAN) para mejorar el rendimiento de SCSS.

3. Marco teórico

3.1. Definición de audio

La principal representación del audio se hace en forma de ondas. Una señal de audio representa el cambio de la presión del aire provocado por las vibraciones a lo largo del tiempo. Una señal de audio está formada principalmente por tres componentes: frecuencia, amplitud y la fase.

- Frecuencia: define el número de variaciones de presión por segundo medido en Hertz (Hz). La frecuencia se relaciona con el tono de los sonidos, una frecuencia alta se percibirá como un sonido más agudo mientras que una frecuencia baja se percibirá como un sonido más grave.

- Amplitud: define la magnitud de las variaciones de presión por segundo medida en decibelios (dB). La amplitud se relaciona con el volumen de un sonido por lo que una mayor amplitud se percibirá como un sonido más alto y una menor amplitud como un sonido más bajo.
- Fase: representa el punto del ciclo en el que se encuentra una onda en un momento especifico en relación a su posición inicial.

Matemáticamente podemos representar la señal de audio como una secuencia discreta de valores x_i

$$x_i = x\left(t = \frac{i}{f_s}\right) \tag{1}$$

Donde t es el tiempo continuo, i es el índice del punto de muestra y f_s es la frecuencia de muestreo.

3.2. Aprendizaje profundo

El aprendizaje profundo (deep learning), es una subdisciplina del aprendizaje automático que se centra en el entrenamiento de modelos de redes neuronales artificiales profundas para realizar tareas de análisis de datos y toma de decisiones. La característica principal del aprendizaje profundo es la utilización de redes neuronales con múltiples capas (redes neuronales profundas) para aprender automáticamente representaciones jerárquicas de los datos, lo que permite el procesamiento de información compleja y la extracción de características.[Rebala et al., 2019]

Dos de las arquitecturas más básicas y conocidas dentro del aprendizaje profundo son las Redes Neuronales Convolucionales (CNN) y las Redes Neuronales Recurrentes (RNN).

3.2.1. Redes Convolucionales (CNN)

Las CNN (Convolutional Neural Network) son un tipo de red neuronal diseñada específicamente para tareas de visión por ordenador, como el reconocimiento de objetos, la detección de características y la clasificación de imágenes. Estas redes están compuestas por capas de convolución que tienen la capacidad de aprender y extraer características relevantes de los datos de entrada. Estas capas aplican filtros para detectar patrones en las señales y son fundamentales para el procesamiento de la información en el contexto de las CNN.

La convolución es una operación matemática fundamental que se utiliza para procesar datos en forma de matrices, como imágenes. Las redes neuronales convolucionales utilizan la

operación de convolución como su principal componente.

La convolución implica deslizar una pequeña matriz llamada filtro o kernel sobre la matriz de entrada para realizar operaciones locales en regiones específicas. Cada elemento del filtro se multiplica con el valor correspondiente en la matriz de entrada y luego se suman estos productos para generar un valor único en la matriz de salida, conocida como "mapa de características". Esta operación (2) se repite en toda la matriz de entrada, generando un mapa de características que resalta ciertos patrones o características presentes en los datos de entrada.

$$S(i,j) = (X * K)(i,j) = \sum_m \sum_n X(i+m, j+n) \cdot X(m,n) \tag{2}$$

Donde $S(i,j)$ es el valor resultante en la posición (i,j) de la salida convolucionada, X es la entrada y K es el filtro (kernel) Los índices (i,j) recorren las posiciones en la salida convolucionada y los índices (n,m) recorren las posiciones dentro del filtro.[Yamashita et al., 2018]

3.2.2. Redes Neuronales Recurrentes (RNN)

Las Redes Neuronales Recurrentes (RNN) son un tipo especial de arquitectura de redes neuronales diseñadas especialmente para abordar la manipulación de datos en forma de secuencias. Estas secuencias son comunes en diversas áreas problemáticas, como en el análisis de series temporales o en la interpretación de datos que exhiben patrones repetitivos. Las Redes Neuronales Recurrentes ofrecen una solución al desafío de modelar secuencias al permitir retener información de datos previos y procesar la entrada actual tomando en cuenta las entradas anteriores y posteriores en la secuencia. Estas redes han demostrado un notable éxito al resolver problemas relacionados con la comprensión de contexto y la identificación de patrones repetitivos. [Rebala et al., 2019]

Una Red Neuronal Recurrente (RNN) es un tipo de arquitectura de redes neuronales diseñada para trabajar con datos secuenciales, donde la información tiene una dependencia temporal. A diferencia de las redes neuronales convencionales, las RNN tienen conexiones internas que les permiten mantener una especie de memoria o estado oculto que puede influir en las predicciones futuras.

En cada paso de tiempo, la RNN recibe una entrada x_t, que puede ser un vector que representa el elemento actual de la secuencia y mantiene un estado oculto h_t , que es una representación interna que almacena información relevante de pasos de tiempo anteriores. El estado oculto de la red puede representarse como se meustar en la ecuación (3).

$$h_t = \mathrm{f}\left(W_{h-h} \cdot h_{t-1} + W_{xh} \cdot x_t + b_h\right) \tag{3}$$

Donde h_t es el estado oculto en el paso de tiempo t, h_{t-1} es el estado oculto en el paso de tiempo anterior, W_{hh} es la matriz de pesos que relaciona el estado oculto anterior con el estado oculto actual, W_{xh} es la matriz de pesos que relaciona la entrada con el estado oculto actual y b_h es el sesgo del estado oculto actual.

La salida de la red se calculará como

$$y_t = \mathrm{f}\left(W_{yh} \cdot h_t + b_y\right) \tag{4}$$

Donde y_t es la salida en el paso de tiempo t, h_t es el estado oculto en el paso de tiempo t, W_{hy} es la matriz de pesos que relaciona el estado oculto con la salida y b_y es el sesgo de la salida.[Rebala et al., 2019]

3.3. Separación de hablantes

3.3.1. Definición del problema

El problema de separación que se plantea puede formularse en términos de la estimación de las fuentes presentes en la señal de audio de mezcla $s_1(t) \ldots s_c(t) \in R^{1\times T}$, dada una señal de mezcla de fuentes $x(t) \in R^{1\times T}$ [Luo and Mesgarani, 2019] como se muestra en la ecuación (5).

$$x(t) = \sum_{i=1}^{C} s_i(t) \tag{5}$$

3.4. Modelos de separación

La mayoría de los algoritmos de separación de hablantes están basado en la representación tiempo-frecuencia (T-F), tomando el espectrograma de la señal de mezcla se busca descomponer dicho espectrograma en las señales de audio individuales que corresponde a cada fuente presente en la mezcla. Esto se puede lograr mediante dos métodos diferentes: la regresión lineal y la estimación de máscara. [Luo and Mesgarani, 2019]

La regresión lineal utiliza un modelo matemático no lineal para estimar los espectrogramas individuales de los hablantes. Mediante el proceso de entrenamiento el modelo aprende a mapear las características del espectrograma de la señal mezclada a los de las fuentes separadas.

Mientras que en la estimación de máscara se utiliza una máscara por cada fuente individual para indicar que partes del espectrograma del audio de mezcla corresponde a cada fuente. Mediante dichas mascaras una vez aplicadas al espectrograma de mezcla podremos obtener los espectrogramas estimados de las fuentes individuales.

Tanto en el método directo como en el método de estimación de máscaras, la forma de onda de cada fuente se calcula utilizando la transformada inversa de Fourier de corto tiempo (iSTFT) del espectrograma de magnitud estimado de cada fuente, junto con la fase original o modificada del sonido de mezcla.

Ambos métodos trabajan con la representación en tiempo-frecuencia por tanto es necesario emplear un método que nos permita pasar de la señal de audio al espectrograma y posteriormente obtener la señal de audio estimada partir del espectrograma estimado. Para ello utilizaremos la transformada de Fourier en tiempo corto (STFT).

La STFT divide la señal en segmentos más pequeños denominados ventanas y luego calcula la Transformada de Fourier para cada una de estas. Las ventanas se superponen entre sí, lo que implica que una ventana no comienza justo donde termina la anterior. Esto permite observar cómo la frecuencia de una señal cambia a lo largo del tiempo. Los resultados de las Transformadas de Fourier para cada ventana se representan mediante una matriz, donde el eje horizontal representa el tiempo y el eje vertical representa la frecuencia. Por otro lado, la transformada inversa de Fourier en tiempo corto nos permita recuperar la señal de audio a partir del espectrograma.

Pero el uso del enfoque en el dominio tiempo-frecuencia presenta ciertas limitaciones en la separación del habla. [Luo and Mesgarani, 2019]

1. Limitaciones de la STFT como método genérico: esta transformación de señal puede no ser capaz de capturar de manera óptima las variaciones que se encuentran dentro de la señal mezclada ya que no está diseñada específicamente para la captura de características distintas del habla.

2. Problema de la reconstrucción precisa de fase y suboptimalidad de los métodos de reconstrucción de fase. La estimación incorrecta de la fase puede provocar el empeoramiento de la calidad del audio separado, ya que la fase de una señal de audio contiene información sobre la ubicación temporal de las diferentes componentes de frecuencia presentes. Los métodos actuales no logran un rendimiento óptimo a la hora de la reconstrucción de la fase.

3. Requerimiento de alta resolución en frecuencia para separación exitosa. La separación efectiva en el dominio tiempo-frecuencia implica descomponerla señal de mezcla en una alta resolución en el dominio de la frecuencia lo cual provoca un aumento de la latencia del sistema.

Para solventar estos inconvenientes surgen diferentes algoritmos de aprendizaje profundo que presentan un enfoque más específico y adaptativo basado en los datos. La idea principal es reemplazar el uso de la STFT para la extracción de características. En su lugar, se basan en la capacidad del modelo para capturar directamente las propiedades acústicas y estructurales de los sonidos mediante un proceso de entrenamiento que ajusta las representaciones de acuerdo con la tarea de separación. Además, también se cuenta con su transformación inversa lo que posibilita la reconstrucción de las señales. Por tanto, se incorpora de forma implícita la extracción de características y la separación mediante arquitecturas de red, cada sistema se diferencia respecto a cómo se extraen las características y en el diseño de la parte de separación de hablantes [Luo and Mesgarani, 2019].

3.4.1. Arquitectura base

Los modelos estudiados durante el desarrollo de este trabajo se centran en modelos end-to-end. Estos modelos representan una avanzada aproximación en el campo del procesamiento de señales de audio y la separación de fuentes de voz en el dominio del tiempo.

Este enfoque end-to-end se destaca por su capacidad para abordar directamente el desafío de la separación de fuentes de voz en señales de audio complejas sin necesidad de dividir el proceso en etapas intermedias. En lugar de utilizar múltiples componentes o algoritmos independientes, estos modelos aprovechan redes neuronales profundas para aprender de manera conjunta las tareas de separación y mejora de la señal de audio.

Cuando se habla de un modelo end-to-end en el contexto de la separación de fuentes de audio, se refiere a la capacidad de ese modelo para tomar una señal de audio mixta (entrada) y separar las fuentes de audio deseadas (salida) sin depender de etapas intermedias específicas, como la extracción de características manuales o el procesamiento previo.

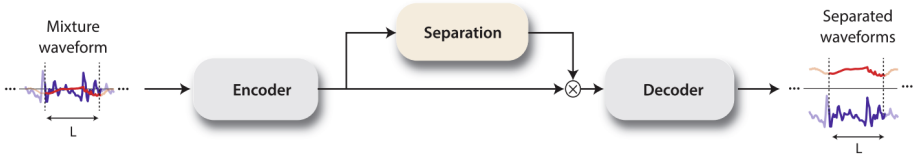

Figura 1: Diagrama de la arquitectura base de las redes de separación [Luo and Mesgarani, 2018]

Por otro lado, la arquitectura base de los modelos evaluados en este proyecto es idéntica. Ambos cuentan con una estructura de codificador/decodificador y una red de separación de fuentes. En este apartado, nos enfocaremos en explicar la parte correspondiente al codificador y decodificador, ya que esta es común a ambos modelos.

1. **Codificador**
   El codificador toma la señal de audio mezclada como entrada y calcula una representación en forma de vectores de pesos. Cada vector w_k representa un segmento k especifico de la señal mezclada y sus valores indican el grado de contribución de dicho segmento respecto del audio mezclado.

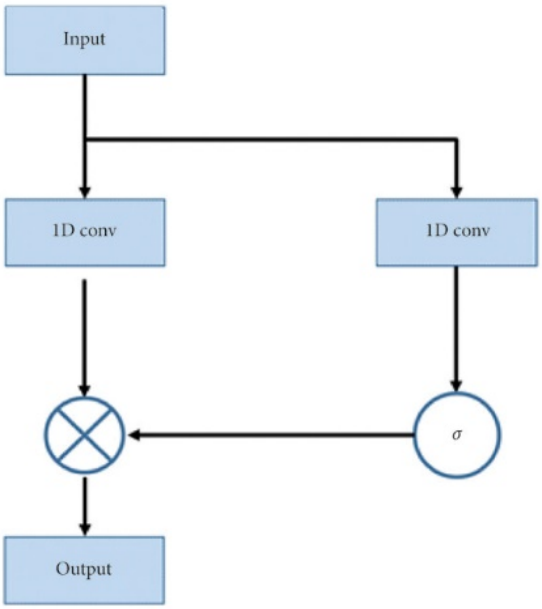

Figura 2: Diagrama de la unidad convolucional 1D [Zhang, 2021]

El cálculo del peso se realiza mediante una capa de convolución con compuertas en una dimensión (1-D) . Esta estructura se inspira en el concepto de unidades recurrentes con compuertas (GRUs) y celdas LSTM. Su objetivo es capturar dependencias tanto a corto plazo como a largo plazo en los datos.

$$w_k = \text{ReLU}\left(x_k \circledast U\right) \odot \sigma\left(x_k \circledast V\right), \quad k = 1, 2, \ldots, K \tag{6}$$

Cada segmento k está asociado con un valor de entrada x_k. El mecanismo de compuerta involucra la generación de señales de compuerta y señales de compuerta moduladas. La señal de compuerta se genera aplicando filtros de convolución al valor de entrada

$(x_k \circledast U)$, esto permitirá a la red a decidir que partes de la entrada son relevantes. Por otro lado, la modulación de compuerta determina en qué medida se debe retener la información de entrada mediante el uso de una función de activación sigmoide respecto a la señal de compuerta generando un valor entre 0 y 1, obteniendo una señal modulada.

El vector de pesos estimado final w_k para el segmento k se obtiene aplicando la función de activación ReLU a la señal de compuerta $(x_k \circledast U)$ y luego multiplicándola por la señal de compuerta modulada $\sigma\,(x_k \circledast V)$. Esta combinación asegura que el peso estimado sea no negativo. Cada w_k tendrá una dimensión de $1 \times N$ donde N representa el número de componentes subyacentes presente en la señal mezclada. [Luo and Mesgarani, 2018]

2. **Decodificador**

   El decodificador toma las máscaras generadas por la red de separación y junto con los vectores de pesos, reconstruirá las señales originales de cada fuente por separado.

   Para cada fuente i obtendremos una matriz M_i de dimensiones $K \times N$ donde K es el índice del fragmento de la señal y N es la longitud del vector señal. Cada elemento $m_{i,j} \in R^{K \times N}$ representa la máscara que se aplica a la fuente i en el fragmento de audio j.

$$M_i = [m_{i,1}, \ldots, m_{i,K}] \tag{7}$$

   La señal de cada fuente se reconstruirá multiplicando la matriz de mascaras M_i por la matriz de vector de pesos no normalizado $W = [w_1, \ldots, w_K]$ obteniendo una matriz de peso de fuentes D_i.

$$D_i = W \cdot M_i = [d_{i,1}, \ldots, d_{i,K}] \tag{8}$$

   Una vez obtenida la matriz de peso de fuentes D_i debemos recuperar las señales de las fuentes en el dominio del tiempo para ello multiplicaremos dicha matriz por la matriz de las señales de base $B = [b_1, \ldots, b_N]$ con $b_j \in R^{N \times L}$. Esta operación es la inversa de la capa convolucional utilizada en la etapa de separación, en donde cada fila de la matriz B corresponde a un filtro 1-D que se aprende de forma conjunta con otras partes de la red.

$$S_i = D_i \cdot B \tag{9}$$

Por último, se concatenan las señales recuperadas (10) de cada uno de los segmentos obteniéndose la señal completa para cada fuente por separado. [Luo and Mesgarani, 2018]

$$s_i(t) = [S_{i,K}], \quad k = 1, 2, \ldots, K \tag{10}$$

3.4.2. Conv-TasNet

Una de las arquitecturas que surgen es la Conv-TasNet la cual fue propuesta por Luo y Mesgarani en [Luo and Mesgarani, 2019]. Esta deriva de la arquitectura TasNet, una red de separación de audio en el dominio del tiempo [Luo and Mesgarani, 2018]. Se trata de una red neuronal con una arquitectura de tipo codificador-decodificador que realiza la separación tras la salida del codificador.

Una vez obtenidos los pesos de mezcla se procede a la estimación de las máscaras de separación que nos permitirán extraer las fuentes separadas de la mezcla original. La red de separación propuesta en la arquitectura TasNet es una red LSTM (Long Short-Term Memory) que permite captar la dependencia temporal en secuencias de datos. Dicha red se entrena para aprender las dependencias temporales entre segmentos de mezcla y generar máscaras que indiquen la contribución de cada fuente en diferentes momentos.

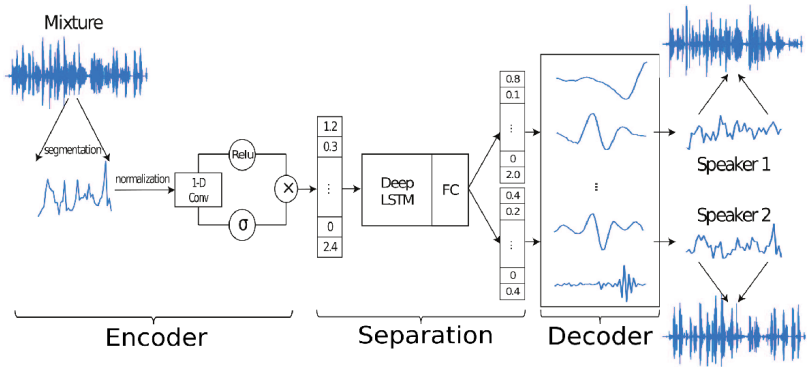

Figura 3: Arquitectura de la red de separación de audio en el dominio temporal TasNet (Luo & Mesgarani, 2018)

La celda LSTM se inicializa con un estado oculto h_0 y un estado interno C_0. El estado interno de la celda es la memoria interna que almacena la información relevante a lo largo del tiempo mientras que el estado oculto es la salida de la celda LSTM producto de procesar dicha información.

Cada celda LSTM consta de tres puertas: la puerta de entrada (input gate), la puerta de olvido (forget gate) y la puerta de salida (output gate). Estas puertas controlan la

información que fluye a través de la celda y como se actualiza su estado interno.

- Puerta de olvido (f_t) : establece que información del estado interno anterior C_{t-1} se debe mantener y por tanto cual se debe descartar olvidando la información irrelevante. Para ello se considera la entrada actual x_t y el estado oculto anterior h_{t-1} y se calcula f_t como

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{11}$$

  Donde W_f es la matriz de pesos para la puerta de olvido, b_f es un vector de sesgo y σ es la función de activación sigmoide.

- Puerta de entrada (i_t): determina que nueva información se debe añadir al estado actual C_t. Se considera la entrada actual x_t y el estado oculto anterior h_{t-1} y se calcula i_t como

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \tag{12}$$

  Donde W_i es la matriz de pesos para la puerta de entrada, b_i es un vector de sesgo y σ es la función de activación sigmoide.

- Puerta de salida (o_t): la función de esta puerta es regular la información que se extraerá del estado interno actual C_t para producir el estado oculto actual h_t. El valor de o_t se calcula como

$$o_t = \sigma\left(W_o \cdot [h_{t-1}, x_t] + b_o\right) \tag{13}$$

  Donde W_o es la matriz de pesos para la puerta de entrada, b_o es un vector de sesgo y σ es la función de activación sigmoide.

Una vez tenemos los valores de las respectivas puertas en el momento t , se calcula C'_t. Esta variable representa la información candidata que podría agregarse al nuevo estado interno.

$$C't = \tanh\left(WC \cdot [h_{t-1}, x_t] + b_C\right) \tag{14}$$

Donde W_c es la matriz de pesos para calcular la información candidata puerta , b_c es un vector de sesgo y σ es la función de activación sigmoide. La función tangente hiperbólica (tanh) asegura obtener un valor en un rango entre -1 y 1, lo que ayuda a controlar el flujo de información.

El estado interno actual se actualizará combinando la información de las puertas de olvido y entrada siguiendo la siguiente expresión:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot C'_t \tag{15}$$

Por último, se genera el nuevo estado oculto h_t combinando el estado interno actual C_t con la puerta de salida o_t

$$h_t = o_t \cdot \tanh(C_t) \tag{16}$$

Las redes LSTM están compuestas por tanto por múltiples celdas LSTM conectadas en serie, lo que les permite modelar dependencias a largo plazo en secuencias completas. Es importante recordar que este tipo de redes están diseñadas para trabajar secuencias de datos, como series temporales. En este caso se toma como entrada la secuencia de vector de pesos $w_1, \ldots, w_K$.

A medida que el vector de pesos se propaga a través de las capas LSTM, cada capa produce una representación intermedia basada en las entradas anteriores y las dependencias temporales aprendidas. En este caso, estas representaciones intermedias están siendo utilizadas para estimar las máscaras correspondientes a cada fuente.

La última capa de la red es una capa totalmente conectada, en la que se aplican transformación lineal a la representación intermedia generada en las capas anteriores y posteriormente se utiliza la función de activación Softmax para convertir la salida en valores de máscara normalizados que representarán la contribución de cada fuente a cada componente de frecuencia. Cada máscara toma valores en un rango de 0 a 1, donde un valor más alto indica una contribución mayor de una fuente en ese componente de frecuencia. La salida de la red para cada fuente i son K vectores de máscara $(m_{i,1}, \ldots, m_{i,K})$.

Para mejorar el funcionamiento de la red se llevó a cabo una normalización de los vectores de peso w_K con el objetivo de hacerlos más estables adecuados para el entrenamiento, lo que puede ayudar a acelerar la convergencia y mejorar el rendimiento general del modelo.

$$\widehat{w_k} = \frac{g}{\sigma} \odot (w_k - \mu) + b, \quad k = 1, 2, \ldots, K \tag{17}$$

Donde g es el vector de ganancia , σ es la desviación estándar del vector w_k, μ es el promedio de los valores del vector w_k y b es el vector de sesgo.

También se introdujeron conexiones de salto para cada dos capas consecutivas con el fin de

mejorar el flujo de gradientes durante el entrenamiento y acelerar así la convergencia de la red.

La arquitectura TasNet mostró una mejora en los resultados respecto a los métodos basados en la representación tiempo-frecuencia, sin embargo, este modelo se vió limitado debido a la dificultad de entrenar una red LSTM con entradas largas al optar por un tamaño de kernel pequeño en el codificador. Esto también conlleva a una disminución de la precisión al aumentar la longitud de la secuencia temporal. Además, se suma un coste computacional alto, ya que este tipo de redes cuenta con un gran número de parámetros.

Es por ello por lo que surge la arquitectura Conv-TasNet, con el objetivo de superar las limitaciones presentadas por el modelo TasNet. Manteniendo la arquitectura de tres bloques - codificador, red de separación y decodificador - se presenta un cambio respecto a la TasNet en el bloque de separación, optando por el uso de bloques de convolución 1-D dilatados apilados en lugar de una red LSTM. Este enfoque se basa en las redes temporales convolucionales (TCN), que buscan reemplazar a la red neuronales recurrentes (RNN) en tareas de modelado de secuencias. Para ello se utilizan bloques de convolución apilados con factor de dilatación, excepto en la primera capa de la red de separación donde se utilizará un bloque de convolución lineal 1×1 que actuará como un cuello de botella determinando el número de canales de entrada de lo bloques siguientes, permitiendo reducir la cantidad de canales y, por lo tanto, la dimensionalidad de los datos y combinar información de diferentes canales en una capa mejorando la eficiencia computacional.

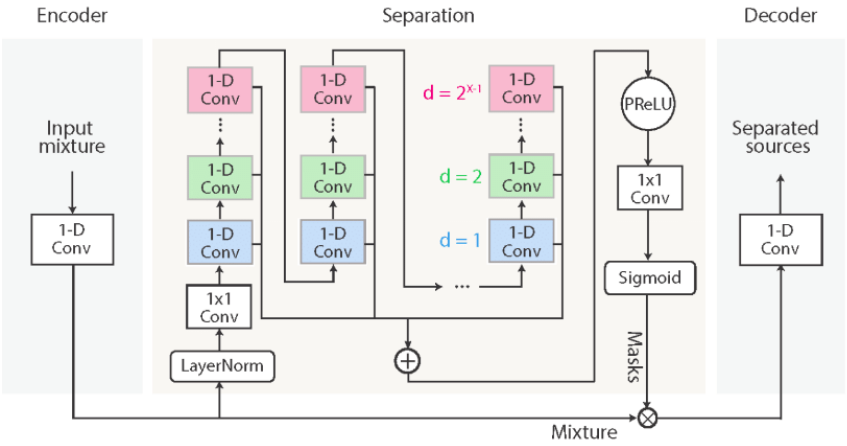

Figura 4: Diagrama de flujo de la arquitectura Conv-TasNet[Luo and Mesgarani, 2019]

Un bloque con factor de dilatación o también conocido como bloque de convolución dilatado

es similar a un bloque de convolución convencional, pero en el primero se introduce un factor de dilatación en el kernel, modificando la forma en el que el kernel se coloca en la señal de entrada. Si hablamos de una dimensión, dado un valor d el kernel omite d posiciones en cada paso. Esto permite aumentar la cantidad de información capturad sin aumentar la complejidad computacional del modelo. Cuando el factor de dilatación d es igual a 1, la convolución dilatada se limita a una convolución convencional. En este caso el factor de dilatación aumenta de forma exponencial lo que permite que la ventana empleada en el contexto temporal sea lo suficientemente grande para capturar las dependencias temporales de largo alcance. Se emplearon M conjuntos de bloques de convolución, donde cada conjunto utiliza factores de dilatación que van desde 1 hasta 2M-1, repitiendo esta secuencia R veces.

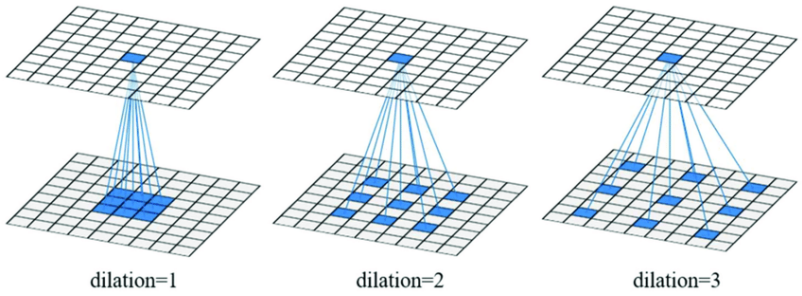

Figura 5: Representación gráfica del campo receptivo en una operación de convolución dilatada, con variados niveles de dilatación [Cui et al., 2019]

Cada bloque de convolución unidimensional sigue un patrón de arquitectura que incorpora una ruta residual, lo que implica que la salida del bloque anterior se utiliza como entrada del bloque actual y una ruta de conexión de salto en donde la salida de todos los bloques se suma para forma la salida final de la red.

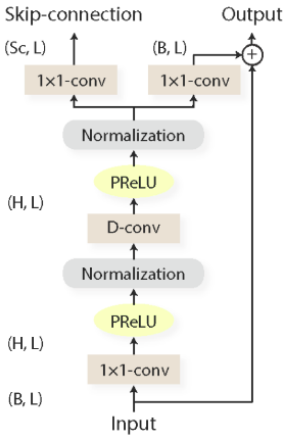

Figura 6: Estructura de un bloque convolucional (1-D Conv) Arquitectura ConvTasNet [Luo and Mesgarani, 2019]

Por otro lado, con el objetivo de disminuir la carga computacional de la red se sustituirá la convolución estándar por la convolución separable en profundidad (S-conv). Esta técnica divide la convolución en dos partes: una convolución en profundidad y una convolución punto a punto.

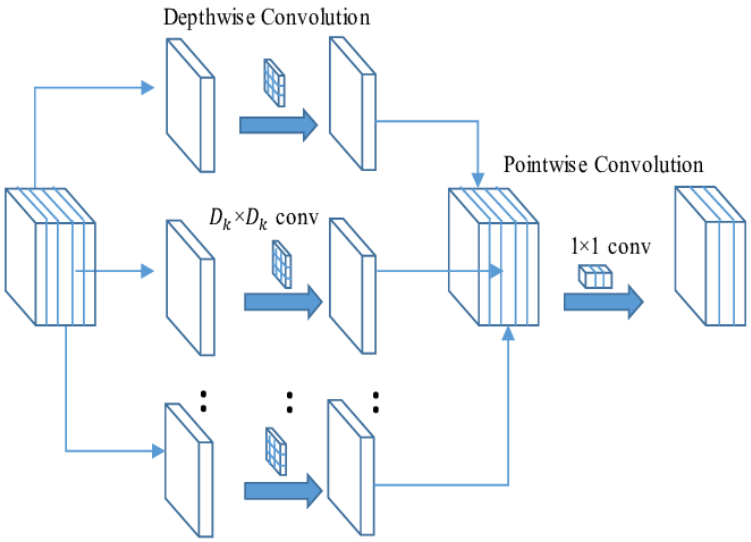

Figura 7: Bloque de convolución separable en profundidad S-conv [Hossain et al., 2020]

- **Convolución en profundidad (D-conv)**: en esta parte se aplica una convolución por cada canal de entrada. En lugar de tener un único filtro que atraviesa todos los canales tendremos un filtro por canal lo que permitirá capturar características específicas de cada canal reduciendo el número de cálculos.

$$D\text{-conv}(Y, , K) = \text{concat}\left(y_j, \circledast, k_j\right), \quad j = 1, \ldots, N \tag{18}$$

  Donde $Y \in R^{G \times M}$ es la entrada a la capa S-conv, $K \in R^{G \times P}$ el kernel de tamaño P

- **Convolución punto a punto (Conv 1x1)**: se aplica un filtro de tamaño 1x1 para combinar la información obtenida de los canales por separado.

$$S\text{-conv}(Y, K, L) = D\text{-conv}(Y, K) \circledast L \tag{19}$$

  Donde $L \in R^{G \times H \times 1}$ el kernel de tamaño 1.

Tras los bloques D-conv y 1x1 se añade una función de activación PReLU y una operación de normalización para evitar que la escala de entrad de los datos influya en la red de separación de fuentes.

3.4.3. Dual-Path RNN

Otro enfoque que se tomó en el problema de la separación de fuentes de audio es el aportado en [Luo et al., 2020] donde en vez de empelar una red basada en bloques de convolucionales (1-DCNN) se opta por una Red Neuronal de Camino Doble (DPRNN).

Esta arquitectura cuenta con dos tipos de capas RNN que funcionan de forma paralela dentro de la red : RNN intra-fragmento y la RNN inter-fragmento. La idea principal bajo este método es dividir la señal de audio de entrada en fragmentos, posteriormente la RNN intra-fragmento operara de forma independiente en cada uno de los fragmentos mientras que la RNN inter-fragmento opera de forma conjunta los fragmentos.

Podremos capturar características de forma local mediante la RNN intra-fragmento y de forma global mediante RNN inter-fragmento. Esto ayuda a capturar tanto la información detallada de corto alcance como las dependencias a largo plazo en la secuencia de entrada, y supone una ventaja respecto a a la arquitectura Conv-TasNet ya que a priori presentan dificultades para captar de forma efectiva las relaciones temporales a largo plazo.

La red de separación se organiza en 3 bloques: segmentación, procesamiento por bloques y solapamiento y suma como se muestra en la figura 8.

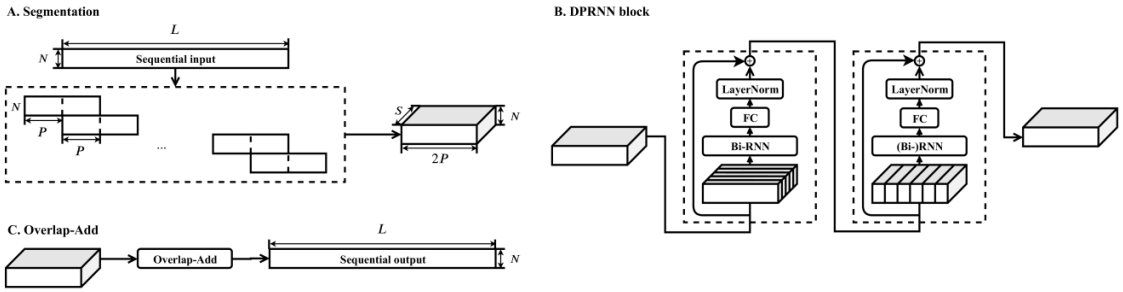

Figura 8: Diagrama de flujo de la arquitectura DPRNN[Luo et al., 2020]

1. Segmentación
   En esta etapa se divide la entrada de la señal de mezcla en fragmentos superpuestos para posteriormente concatenarlos en un tensor tridimensional. Dada una entrada $W \in R^{N \times L}$ siendo N la dimensión de la entrada y L el número de pasos de tiempo obtendremos K / P fragmento de longitud K con desplazamiento de P unidades entre fragmentos. Para evitar problemas de dimensionalidades y que todas las muestras W aparezcan en K / P fragmentos el ultimo fragmento se completará con ceros. Se obtienen S fragmentos $D_s \in R^{N \times K}$ que se concatenan obteniendo T.

$$T = [D_1, \ldots, D_s] \in \mathbb{R}^{N \times K \times S} \tag{20}$$

2. Procesamiento por bloques.
   Se emplea el vector tridimensional T obtenido de la segmentación se pasa como entrada de la RNN.

   Respecto a la RNN Intra-Fragmento. Cada bloque $b = 1, \ldots, B$ procesara la segunda dimensión del tensor T_b, es decir se aplica a cada uno de los S fragmentos para cada paso K a lo largo de la secuencia .

   Además, se aplica una RNN bidireccional, una extensión de las RNN tradicionales que permiten capturar información contextual tanto del pasado como del futuro. Esto se logra gracias a la combinación de dos RNN unidireccionales donde una procesa la información en orden cronológico y otra en orden inverso.

$$U_b = [h_b\left(T_b(:,:,i)\right), i = 1, \ldots, S] \tag{21}$$

   Como resultado obtendremos un tensor $U_b \in R^{H \times K \times S}$ siendo H la dimensión de características de la RNN bidireccional (dimensión del estado interno de la red). Para adaptar la dimensionalidad de U_b de regreso a la forma de T_b, se aplica una capa completamente conectada obteniendo $\widehat{U_b} \in R^{N \times K \times S}$.

$$\widehat{U_b} = [G\ U_b(:,:,i) + m,\ i = 1, \ldots, S] \tag{22}$$

   Donde $G \in R^{N \times H}$ es la matriz de pesos y $m \in R^{N \times 1}$ el vector de sesgo de la capa totalmente conectada. Por último, se aplica una normalización de capa a $\widehat{U_b}$ y el nuevo tensor $\widehat{T_b}$ se calcula como

$$\widehat{T_b} = T_b + \mathrm{LN}(\widehat{U_b}) \tag{23}$$

   En cuando a la RNN Inter-Fragmento, se utiliza el tensor tridimensional transformado por el módulo RNN inter-fragmento $\widehat{T_b}$ y se operara sobre la última dimensión por tanto en cada uno de los K pasos, la RNN procesa los S fragmentos obteniendo $V_b \in R^{H \times K \times S}$ como

$$V_b = \left[h_b\left(\widehat{T_b}\left[:, i, :\right]\right), , i = 1, \ldots, K\right] \tag{24}$$

   siendo H la dimensión de características de la RNN bidireccional (dimensión del estado interno de la red)

Al emplear un RNN bidireccional en este módulo intra-fragmento en cada paso de tiempo K en el tensor $\widehat{T}_b$, contiene información tanto de los pasos anteriores como de los pasos posteriores dentro del mismo fragmento por lo que permite cada fragmento se procesa de manera completa y coherente en el modulo inter-fragmento.

Como en el caso anterior se emplea una capa completamente conectada y un a normalización LN sobre V_b obteniendo $\widehat{V}_b \in R^{N \times K \times S}$ y el nuevo tensor $\widehat{T}_b$ que será la salida del bloque b de DPRNN

$$\widehat{T}_b = \widehat{T}_b + \mathrm{LN}(\widehat{V}_b) \tag{25}$$

Mientras $b < B$, la salida generada en el bloque $\widehat{T}_b$ se empleará como entrada del siguiente bloque T_{b+1}.

3. Solapamiento y suma.
   Se emplea la salida proporcionada por el ultimo capa de la red, $T_{B+1} \in R^{N \times K \times S}$, que se transforma de nuevo en una salida secuencial mediante el método de solapamiento y suma. Se aplica sobre los segmentos S y se obtiene un único segmento $Q \in R^{N \times L}$

4. Marco experimental

La biblioteca Asteroid se ha convertido en una herramienta esencial en nuestro proyecto, permitiéndonos realizar implementaciones de modelos de procesamiento de señales de audio de manera eficiente y efectiva. Para la implementación de modelos, Asteroid nos ha proporcionado una amplia gama de arquitecturas así como conjunto de métricas de evaluación de señales de audio que son esenciales para medir el rendimiento de nuestros modelos. [Pariente et al., 2020]

4.1. Datos

Los datos utilizados en este estudio fueron extraídos del conjunto de datos Bern Cocktail Party (BCP). Este conjunto consta de 55938 escenarios de fiestas con 20 personas grabadas utilizando un simulador de cabeza y torso con procesadores de audio diseñados para implantes cocleares.[Fischer et al., 2020]

Dado que los conjuntos de datos relacionados con el desafío de separación de fuentes en situaciones tipo cocktail party, obtenidos mediante grabaciones con micrófonos colocados en la cabeza humana, son escasos y la adquisición de nuevos datos para futuros análisis implica un alto costo, se optó por emplear las grabaciones generadas mediante el simulador

de cabeza y torso.

La elección de este conjunto de datos también permite enfrentarnos a una situación que se asemeja más a la vida real. En la vida cotidiana, las personas cuentan con solo dos oídos para percibir el sonido, y el simulador utilizado en el estudio reproduce de manera realista la forma en que el sonido viaja por el cuerpo humano y llega al oído interno, en este caso a los implantes cocleares.

El conjunto de datos cuenta con 4953 grabaciones (alrededor de 80h 25min) obtenidas mediante micrófonos colocado en el simulador. Se emplearon 15224 disposiciones diferente del escenario Cocktail Party. Cada uno de este escenario está formado por superposiciones acústicas de audio en inglés, música y ruido de murmullo.

Para cada grabación, contaremos con el audio mono resultante de la mezcla de todas las fuentes, así como el audio mono de cada fuente por separado. Utilizaremos conjuntos de datos que involucran 2 y 3 hablantes en diversas condiciones, incluyendo escenarios sin ruido de fondo, con música y con murmullos. Es importante resaltar que en este conjunto de datos también se nos proporcionan los audios monocanal tanto de la música como del sonido de murmullos presentes en el audio mezclado, tratándolos como fuentes de audio adicionales presentes en la mezcla.

Dado que la duración de las grabaciones abarca de 3 a 10 segundos, se llevó a cabo un proceso de preprocesamiento en los datos mediante el cual se recortaron tanto el audio mezclado como el de las fuentes individuales en fragmentos de 3 segundos. Esta adaptación fue necesaria, ya que los modelos utilizados requerían que todos los datos tuvieran las mismas dimensiones.

Además, se realizó un muestreo de los audios a una frecuencia de 8000 Hz ya que las grabaciones estaban recogidas a una frecuencia de 44100 Hz, esto permitió reducir la carga computacional y la demanda de memoria. También permitió acelerar el proceso de entrenamiento sin afectar significativamente el rendimiento final puesto que se emplearon datos de menor resolución.

También llevó a cabo la separación de cada conjunto de datos en tres subconjuntos distintos: entrenamiento, validación y prueba, con una proporción de 70 %, 20 % y 10 % respectivamente. Es importante destacar que fijamos una semilla durante este proceso de separación para garantizar la reproducibilidad de los resultados.

4.2. Métricas de evaluación

Para evaluar la capacidad de los modelos nos basaremos en métricas que evalúan la precisión proporcionada en la separación de fuentes de audio, comparando las señales de audio originales de las fuentes con las proporcionadas por el modelo.

Para poder calcular dichas métricas de vemos obtener entonces las señales originales sin mezclar $s \in R^{N \times T}$y las señales de fuente estimadas $\widehat{s} \in R^{N \times T}$. La señal estimada se puede descomponer en 4 componentes principales. [Vincent et al., 2006]

$$\widehat{s} = s_{\text{target}} + e_{\text{interf}} + e_{\text{noise}} + e_{\text{artif}} \tag{26}$$

s_{target} es una variante alterada de la fuente original en la cual pueden estar presentes ciertas distorsiones que se consideran permitida. e_{interf} representa la interferencia derivada de otras fuentes $s_{j'} \; \forall \; j' \neq j$, que podrían mezclarse con la señal de la fuente estudiada. e_{noise} representa el ruido , y e_{artif} representa las distorsiones prohibidas que no se encuentran en el conjunto de distorsiones permitidas. [Vincent et al., 2006]

Una vez tenemos nuestra señal descompuesta podemos proceder al cálculo de las métricas El uso conjunto de las métricas relación fuente-distorsión (SDR) y relación señal-ruido invariante de escala (SI-SNR) en la evaluación de la separación de fuentes de audio proporciona una evaluación más completa, robusta y confiable de la calidad del proceso de separación. Estas métricas evalúan diferentes aspectos en el proceso de separación, como la interferencia de otras fuentes de sonido (SDR), ya sea la relación entre la fuente original y la señal distorsionada, y la relación señal-ruido (SI-SNR), conservando la señal objetivo y reduciendo el ruido de manera efectiva.

Respecto a estas métricas también emplearemos SDRi y SN-SNRi. Se trata de la mejora SI-SNR (SI-SNRi) y la mejora SDR (SDRi). Estas métrica miden la calidad de la separación antes y después de aplicar un proceso de mejora de señal. Para calcular la mejora calculamos la métrica antes de la separación y después de ella y posteriormente restamos estos valores.

4.2.1. Relación fuente-distorsión (SDR)

La métrica SDR se enfoca en medir la calidad de la separación de fuentes considerando la relación entre la señal original y la señal distorsionada después de aplicar un algoritmo de separación. La SDR toma en cuenta tanto la interferencia de otras fuentes de sonido como los elementos introducidos por el proceso de separación. Se expresa en decibelios (dB) y un valor más alto indica una mejor calidad de la separación de fuentes.

$$\text{SDR} = 10 \log_{10} \frac{|s_{\text{target}}|^2}{|e_{\text{interf}} + e_{\text{noise}} + e_{\text{artif}}|^2} \tag{27}$$

Una descripción de SDR sería que constituye una métrica global que evalúa la calidad auditiva de una separación en su totalidad.[Gidlöf, 2023]

4.2.2. Relación señal-ruido invariante de escala (SI-SNR)

La métrica SI-SNR se centra en evaluar la calidad de la separación de fuentes considerando la relación señal-ruido. Se basa en la capacidad de un modelo para preservar la señal objetivo mientras reduce al mínimo la interferencia o el ruido en la señal separada. Se expresa en decibelios (dB) y un valor más alto indica una mejor calidad de la separación de fuentes.

El cálculo de esta métrica se define de la siguiente forma

$$\text{SI-SNR} = 10 \log_{10} \frac{|s_{\text{target}}|^2}{|e_{\text{noise}}|^2} \tag{28}$$

Donde $s_{\text{target}} \in \mathbb{R}^{1\times T}$ es una versión escalada y alineada a la fuente original de la fuente estimada (29), y e_{noise} se calcula siguiendo la fórmula (30) [Gidlöf, 2023].

$$s_{\text{target}} = \frac{\langle \widehat{s}, s \rangle}{|s|^2} s \tag{29}$$

$$e_{\text{noise}} = \widehat{s} - s_{\text{target}} \tag{30}$$

4.3. Problema de permutación de etiquetas

Como los modelos que estamos estudiando emplean máscaras para realizar a la separación de fuentes, existe un problema común denominado: problema de permutación de etiquetas.

Cuando existen múltiples fuentes de sonido en el audio de mezcla la red de separación genera una máscara por fuente de forma simultánea, el problema surge porque no se conoce de antemano el orden específico.[Kolbæk et al., 2017]

Supongamos que procesamos una señal de audio mezclada en la que hay presente dos hablantes, A y B, y se obtiene dos máscaras de salida $\widehat{m_{1,i}}$ y $\widehat{m_{2,i}}$. La salida del modelo será entonces un vector $\widehat{u_i}$ formado por las dos mascarás obtenidas. Pero existen dos formas de construir este vector

$$\widehat{u_i} = [\widehat{m_{1,i}}, \widehat{m_{2,i}}] \tag{31}$$

$$\widehat{u_i} = [\widehat{m_{2,i}}, \widehat{m_{1,i}}] \tag{32}$$

Si la máscara $\widehat{m}1, i$corresponde a la fuente A y $\widehat{m}2, i$ corresponde a la fuente B, entonces el orden correcto sería $\widehat{u_i} = [\widehat{m_{1,i}}, \widehat{m_{2,i}}]$Pero si, por alguna razón, el modelo produce las máscaras en el orden opuesto, no sería el caso. [Kolbæk et al., 2017]

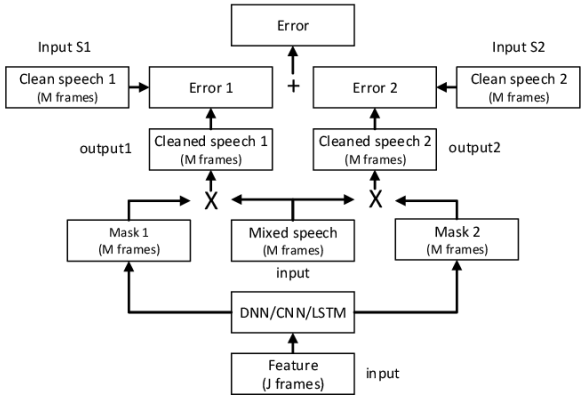

Figura 9: Modelo de separación de dos hablantes convencional

Esto se conoce como problema de permutación en el contexto de la separación de fuentes de audio se refiere a la incertidumbre sobre cómo asignar correctamente las máscaras de salida a las fuentes de origen a las que pertenecen.

Una solución planteada a este problema es el Entrenamiento Invariante a Permutaciones (PIT) [Yu et al., 2017]. La idea en la que se basa este método es entrenar el modelo de separación de tal forma que no se vea influenciado por la permutación de las fuentes.

Las señales de referencia se emplearán como un conjunto y no como una lista ordenada y el cálculo de la función de coste se hará para todas las posibles permutaciones de dicho conjunto. Se escogerá la permutación que obtenga un mejor valor para la función de coste del modelo. Estaríamos ajustando el modelo para incrementar el rendimiento en cuanto a la asignación de fuentes.

Este método tiene una limitación, ya que, durante la etapa de inferencia, el proceso de separación se maneja en términos de grupos de tiempo. Un grupo de tiempo es un conjunto de momentos de entrada que se procesan conjuntamente como una unidad, generando un

"grupo de tiempo de salida". Basándonos en la técnica PIT, la separación de todas las señales dentro de un mismo grupo de tiempo seguirá la misma permutación. No obstante, esta permutación puede cambiar entre distintos grupos de tiempo. En otras palabras, se permite que las señales de diferentes fuentes cambien su orden en diferentes momentos, pero se mantiene constante dentro de un mismo grupo de tiempo. La variación en la permutación de estas señales a lo largo del tiempo se considera válida, y el modelo se entrena para manejar todas las posibles permutaciones dentro de un grupo de tiempo. Esto posibilita abordar las distintas formas en que las señales pueden ordenarse en diferentes partes de la grabación de audio mezclada.

A partir del Entrenamiento Invariante a Permutaciones (PIT) se desarrolla el Entrenamiento Invariante a Permutaciones a nivel de enunciado (uPIT), que busca resolver tanto el problema de seguimiento de grupos de tiempo como el problema de permutación de etiquetas [Kolbæk et al., 2017].

El método uPIT utiliza la permutación que minimiza el error de separación global para todos los momentos en la señal de audio. Se trata de encontrar una manera de ordenar las señales de las diferentes fuentes en cada instante de tiempo para reducir al máximo el error de separación en toda la grabación. En lugar de calcular permutaciones distintas para cada momento, como se hace en el método PIT, el método uPIT busca una permutación que funcione bien para todos los instantes de la grabación en conjunto. De esta manera, se logra la mejor separación posible de las voces de los diferentes altavoces presentes en la grabación.

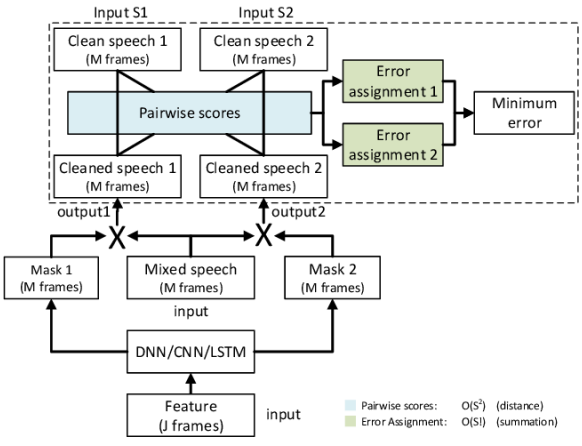

Figura 10: Modelo de separación de dos hablantes con entrenamiento invariante de permutaciones [Yu et al., 2017]

Llevando la idea de la técnica PIT a un nivel más alto de abstracción, donde en lugar de trabajar con segmentos individuales de audio, están considerando segmentos más largos

de discurso hablado, de separar diferentes hablantes en segmentos completos en lugar de partes individuales de audio.

4.4. Configuración de parámetros

Para garantizar la comparabilidad de los resultados y la imparcialidad en el funcionamiento de los modelos, se realizaron ajustes en los parámetros de entrenamiento utilizando como referencia los modelos originales [Luo and Mesgarani, 2019] [Luo and Mesgarani, 2018]. Se llevaron a cabo ajustes menores debido a limitaciones computacionales y con el propósito de lograr una mayor similitud en los parámetros entre ambos modelos. De esta forma, se aseguró la coherencia con la visión inicial y se facilitó una evaluación objetiva de los resultados.

Ambos modelos fueron sometidos a un entrenamiento a lo largo de 100 épocas, utilizando lotes de tamaño reducido compuestos por únicamente 2 unidades. Este proceso se vio limitado por la memoria del equipo empleado, que imponía restricciones al almacenamiento y procesamiento. Para mantener un equilibrio entre la precisión y la velocidad de entrenamiento, se optó por una tasa de aprendizaje de 1×10^{-3}, la cual demostró ser óptima. [Luo and Mesgarani, 2019]

Como algoritmo de optimización, se empleó Adam. Este algoritmo ha demostrado una efectividad considerable en variedad de problemas y arquitecturas. Desde redes neuronales convolucionales hasta redes recurrentes y modelos generativos. Adam permite la adaptabilidad de las tasas de aprendizaje, esto acelera el proceso de convergencia, reduciendo la necesidad de intervenciones manuales para ajustar la tasa de aprendizaje. Uno de los factores clave de Adam es su consideración de momentos, lo que permite manejar gradientes ruidosos de manera efectiva.

En el ámbito de los algoritmos de separación de fuentes, es común recurrir a métricas de desempeño para evaluar la calidad de la separación lograda. En este caso, la función de pérdida elegida es la relación fuente-distorsión (SDR), una métrica que evalúa la calidad de la separación. El objetivo fundamental del entrenamiento del algoritmo es minimizar el valor negativo de esta métrica, lo que se traduce en una mejora continua de la separación lograda.

En cuanto a la configuración de los hiperparámetros asociados a cada modelo, hemos optado por utilizar aquellos que fueron establecidos en los respectivos artículos donde se lograron los resultados óptimos. En lo que respecta a los parámetros particulares tanto del codificador como del decodificador, decidiremos aplicar valores establecidos en los artículos de referencia en ambas arquitecturas.

Símbolo	Descripción	
N	Número de filtros en el autoencoder	512
L	Longitud de los filtros (en muestras)	16
B	Número de canales en la capa cuello de botella y en los bloques 1×1-conv en las rutas residuales	128
Sc	Número de canales en los bloques 1×1-conv en las rutas de conexión de salto	128
H	Número de canales en los bloques convolucionales	512
P	Tamaño del kernel en los bloques convolucionales	3
X	Número de bloques convolucionales en cada repetición	8
R	Número de repeticiones	3

Tabla 1: Hiperparámetros arquitectura ConvTasNet

Símbolo	Descripción	
N	Número de filtros en el autoencoder	64
L	Longitud de los filtros (en muestras)	16
B	Número de canales en la capa cuello de botella	128
K	Tamaño de los fragmentos en que se divide la secuencia de entrada.	100
P	Tamaño de salto de los fragmentos	50
H	Unidades ocultas en RNNs de bloques internos y externos en cada dirección.	128
X	Número de bloques DPRNN en cada repetición	1
R	Número de repeticiones	6

Tabla 2: Hiperparámetros arquitectura DPRNN

4.5. Proceso de entrenamiento

Utilizamos 4 conjuntos de datos para el entrenamiento de ambos modelos, los cuales se diferencian en el número de locutores presentes en la señal de mezcla, así como en el tipo de ruido de fondo presente en la grabación. El primer conjunto de datos, denominado "Head_and_Torso_Simulator_Audio_2speakers_no_noise_chunks", consta de 2 oradores y carece de ruido de fondo. A continuación, el conjunto "Head_and_Torso_Simulator_Audio_3

speakers_no_noise_chunks"también se caracteriza por la ausencia de ruido de fondo, pero en esta ocasión el número de oradores se incrementa a 3.

Por otro lado, contamos con el conjunto denominado "Head_and_Torso_Simulator_Audio_2speakers_in_babble_chunks"donde se presentan 2 oradores y se introduce sonido de murmullo de fondo. Finalmente, el conjunto "Head_and_Torso_Simulator_Audio_2speakers_in_music_chunks"mantiene el número de oradores en dos, pero en esta ocasión se añade ruido de música de fondo.

Estos conjuntos de datos variados nos permiten entrenar y evaluar los modelos en una amplia gama de escenarios, abordando diversas condiciones de mezcla y ruido para mejorar su capacidad de separación y rendimiento general.

El proceso de entrenamiento varía en función del conjunto de datos utilizado en el modelo, específicamente dependiendo del tipo de ruido presente en las grabaciones. En el caso del primer y segundo conjunto de datos, se llevará a cabo un entrenamiento siguiendo la configuración de parámetros previamente explicada. El número de fuentes de sonido se establecerá en 2 y 3 respectivamente. En lo que respecta al tercer y cuarto conjunto de datos, se realizarán dos entrenamientos distintos, variando el número de fuentes de sonido presentes en la mezcla.

Cada configuración de entrenamiento se ajustará según las características particulares de cada conjunto de datos, permitiendo a los modelos adaptarse de manera efectiva a las condiciones específicas de mezcla y ruido. Esta estrategia garantiza una mayor robustez y versatilidad en la capacidad de separación de fuentes de los modelos en diferentes contextos y escenarios.

La razón detrás de la elección de llevar a cabo dos entrenamientos distintos radica en la intención de evaluar la capacidad de los modelos en situaciones más realistas del problema conocido como cocktail party. En primer lugar, llevaremos a cabo un primer entrenamiento que involucra la presencia de 3 fuentes de sonido en la mezcla de audio, ya que en este escenario contamos con la fuente asociada tanto a los murmullos de fondo como a la música ambiente. En un enfoque alternativo, procederemos a un segundo entrenamiento en el cual reduciremos el número de fuentes de sonido a 2.

Esta estrategia de entrenamiento dual presentara varias ventajas significativas:

- **Realismo de escenario**: al realizar dos entrenamientos distintos que reflejan situaciones reales de cocktail party, donde puede haber múltiples fuentes de sonido y

diferentes combinaciones de ruido de fondo, se logra una mejor adaptación de los modelos a escenarios complejos y variados.

- **Resistencia al ruido**: al exponer los modelos a diferentes tipos de ruido y condiciones de mezcla, se fortalece su capacidad para separar fuentes de manera precisa incluso cuando hay ruido de fondo presente. Esto mejora la capacidad de separación en escenarios ruidosos, como las fiestas de cóctel.
- **Evaluación integral**: la comparación de resultados entre los dos entrenamientos proporciona información sobre cómo se desempeñan los modelos en diferentes configuraciones. Esto ayuda a comprender la capacidad en diversas situaciones de separación de fuentes.

5. Resultados

En esta sección, procederemos a realizar un análisis de los resultados obtenidos a través de los modelos previamente entrenados. Se recopilaron las métricas SDRi (Índice de Mejora de la Relación Señal-Ruido) y SN-SNRi (Índice de Mejora de la Relación Señal-Ruido Espectral), además de presentar gráficos que muestran la evolución de la función de pérdida a lo largo del proceso.

Además de las métricas, que nos permiten evaluar la capacidad de separación de los modelos, los gráficos relacionados con la evolución de la función de pérdida nos permiten evaluar la velocidad de convergencia de nuestro modelo. Esto es importante para determinar si el modelo ha logrado estabilizarse y si se ha entrenado de manera efectiva.

Los resultados obtenidos para los modelos entrenados (ver tabla 3) en un escenario libre de ruido, donde solo están presentes los oradores, son muy favorables. En cuanto a los valores de SDRi obtenidos, podemos notar que tanto para los modelos ConvTasnet se alcanza un valor cercano a los 15 dB, lo que indica un excelente rendimiento del modelo, independientemente del número de fuentes.

Respecto a los valores de los conjuntos de validación y prueba, son bastante similares, excepto en el último caso, donde el SDR del modelo con dos oradores supera al de tres. Por otro lado, al analizar los modelos DPRNN, se observan valores ligeramente inferiores, especialmente en el caso del modelo con tres oradores, lo que podría sugerir que el número de oradores afecta al funcionamiento del algoritmo de separación. Este mismo patrón se refleja en los valores de validación y prueba.

Modelo	Train_SDRi	Val_SDRi	Test_SDRi	Train_SN-SNRi	Val_SN-SNRi	Test_SN-SNRi
ConvTasNet_2speakers	15.241335	14.462122	15.495666	-8.589516	2.689108	-1.561294
DPRNN_2speakers	10.083328	9.297959	10.055634	-8.035240	0.300695	-3.164351
ConvTasNet_3speakers	15.537989	14.738210	13.316523	-12.291194	-10.124516	-11.336721
DPRNN_3speakers	4.331401	3.497585	1.929250	-7.805893	-9.580969	-7.738111

Tabla 3: Resultados de los Modelos - Entorno no ruidoso

En cuanto a los valores del Índice de Mejora de la Relación Señal-Ruido Espectral (SN-SNRi), se observa que en la mayoría de los casos se obtienen valores negativos. Estos valores indican que las señales obtenidas tienen una calidad deficiente y están afectadas por la presencia de ruido. El mejor valor registrado es de 2.689108 dB, y se obtuvo en el conjunto de validación del modelo que maneja dos oradores.

En términos generales, se puede notar que los valores de SN-SNRi son más bajos en los modelos que involucran a más oradores, independientemente del conjunto de datos utilizado. Además, se puede concluir que el modelo ConvTasnet obtiene mejores resultados cuando se trabaja con dos oradores, mientras que el modelo DPRNN funciona de manera más eficiente cuando se trata de tres oradores. Estos hallazgos indican que la complejidad del modelo y la cantidad de oradores influyen en la calidad de las señales obtenidas y en la capacidad para reducir el ruido presente en ellas.

Acerca de la velocidad de convergencia del modelo, evaluada a través de la evolución de la función de pérdida, ninguno de los modelos logra alcanzar una convergencia completa o estabilizarse en torno a un valor específico de la función de pérdida. Tanto en los casos de dos oradores como en los de tres oradores, se observa que en los modelos ConvTasnet, la disminución de los valores de la función de pérdida es más pronunciada en comparación con los modelos DPRNN. También se pueden observar oscilaciones pronunciadas en los valores de la función de pérdida en la evolución por épocas.

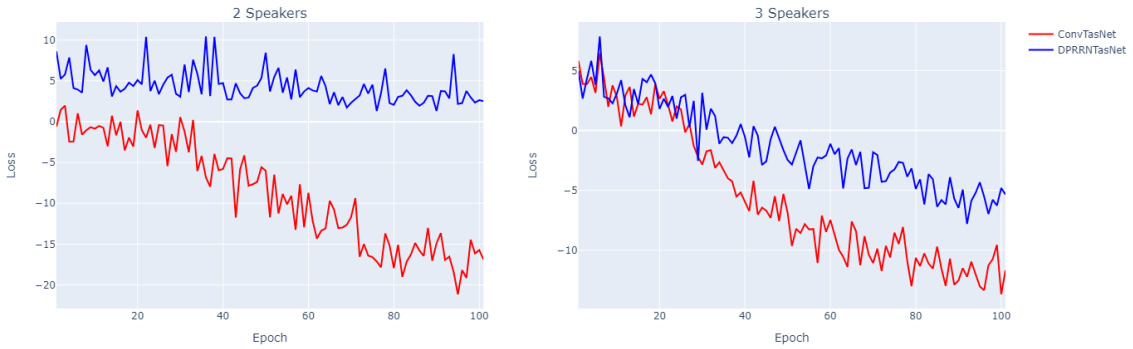

Figura 11: Evolución de los valores de la función de pérdida durante el entrenamiento - Entorno no ruidoso

En relación a los resultados obtenidos con una base de datos que incluye un escenario de murmullo, se pueden apreciar diferencias significativas entre los modelos de dos oradores y los modelos de tres oradores. En términos del SDRi, los modelos de dos oradores muestran un rendimiento superior en comparación con los modelos de tres oradores, y esta mejora es evidente en los conjuntos de entrenamiento, validación y prueba. En cuanto a los valores de SN-SNRi, la diferencia no es tan marcada, y se observa que los modelos de tres oradores logran valores ligeramente mejores, aunque no necesariamente óptimos.

Modelo	Train_SDRi	Val_SDRi	Test_SDRi	Train_SN-SNRi	Val_SN-SNRi	Test_SN-SNRi
ConvTasNet_2speakers	6.252124	5.898170	5.880346	-8.319438	-9.691925	-10.217670
DPRNN_2speakers	5.208428	4.923588	4.872973	-2.273670	-1.946310	-2.049056
ConvTasNet_3speakers	-12.549576	-13.093908	-13.768713	-8.034108	-7.877514	-8.308205
DPRNN_3speakers	-12.046008	-12.435085	-13.108159	-6.443164	-6.874418	-7.404101

Tabla 4: Resultados de los modelos - Entorno ruidoso

Al realizar una comparación en relación al tipo de modelo utilizado, se puede afirmar que el modelo ConvTasNet claramente arroja resultados superiores en comparación con el modelo DPRNN tanto en el caso de dos oradores como en el de tres, cuando se evalúa la métrica SDRi. Sin embargo, en lo que respecta a la métrica SN-SNRi, se observa que los modelos DPRNN presentan un desempeño superior.

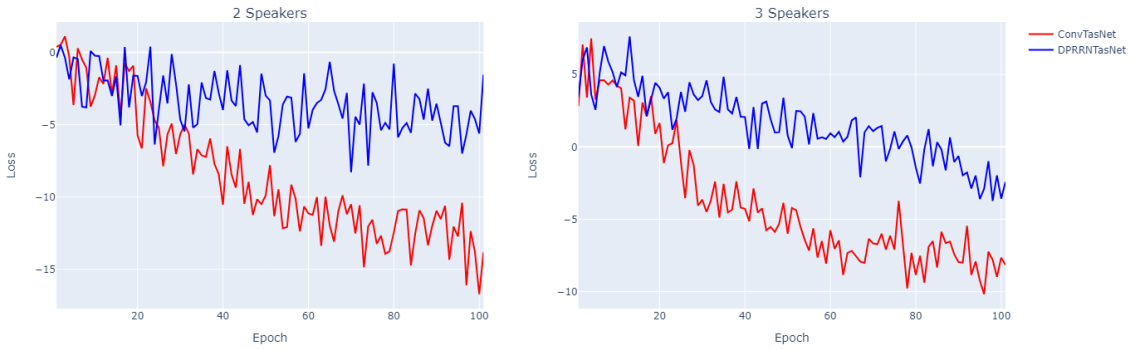

Figura 12: Evolución de los valores de la función de pérdida durante el entrenamiento - Entorno ruidoso

Respecto a los gráficos que representan la evolución de la función de pérdida, se aprecia un patrón similar al observado en los modelos sin ruido de fondo. Se destaca que la progresión de los valores de costo es más favorable en los modelos ConvTasNet en contraste con los modelos DPRNN, donde la reducción de dicho valor se produce de manera más gradual y prolongada. Además, el patrón de oscilaciones se mantiene constante, sin manifestar una convergencia clara hacia un valor de costo específico.

Por último, en lo que respecta a los resultados obtenidos con los modelos en un escenario de música, en líneas generales se observa que los resultados no son satisfactorios. Solo se obtiene un resultado destacado en el conjunto de entrenamiento del modelo ConvTasNet para dos oradores con un valor de 6.97 dB. Sin embargo, en el caso de los modelos DPRNN, se obtienen mejores valores en la métrica SI-SNRi que en la métrica SDRi, a diferencia de lo que ocurre en los modelos ConvTasNet, donde se obtienen mejores resultados en SDRi que en SI-SNRi. En relación a los gráficos que representan la función de pérdida, se observa

Modelo	Train_SDRi	Val_SDRi	Test_SDRi	Train_SN-SNRi	Val_SN-SNRi	Test_SN-SNRi
ConvTasNet_2speakers	6.976828	-2.619166	-2.644157	-10.799724	-5.751357	-6.190307
DPRNN_2speakers	-2.214626	-1.941923	-2.067427	-2.796564	-2.846970	-3.192492
ConvTasNet_3speakers	-11.804407	-11.272611	-11.611500	-7.452197	-7.135220	-7.296200
DPRNN_3speakers	-7.545327	-7.430502	-7.546956	-2.565378	-1.999737	-1.715504

Tabla 5: Resultados de los modelos - Entorno con música

que se mantiene el patrón previamente observado. Los modelos DPRNN muestran una disminución más lenta, llegando a un punto donde parecen estancarse en cierta medida. Por otro lado, en los modelos ConvTasNet, se aprecia una evolución más progresiva sin alcanzar una estabilización clara en torno a un valor específico.

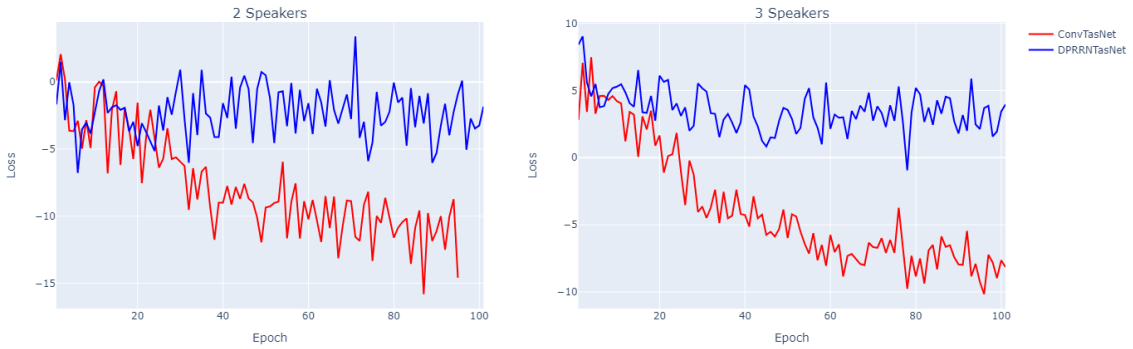

Figura 13: Evolución de los valores de la función de pérdida durante el entrenamiento - Entorno con música

En lineas generales podemos concluir que los modelos que trabajan con datos de audio sin ruido de fondo, los resultados obtenidos suelen ser buenos. Sin embargo, en escenarios ruidosos, como aquellos con música o susurros de fondo, los resultados no alcanzan un nivel considerablemente alto. Es decir, estos modelos enfrentan dificultades para separar las fuentes en presencia de ruido adicional.

Además, la técnica de entrenamiento que involucra una fuente adicional de ruido, como murmullo o música, en lugar de centrarse únicamente en las fuentes de los oradores que se desean separar, ha demostrado proporcionar peores resultados. Por lo tanto, podemos concluir que estos modelos son capaces de trabajar con audio mezclado con ruido, pero son menos eficientes cuando se considera dicho ruido como una fuente adicional en el proceso de separación de fuentes.

Sobre a la convergencia de los algoritmos, podemos afirmar que esta ha sido influenciada significativamente por el tamaño de lote utilizado. Cuando se emplea un tamaño de lote tan pequeño, los modelos realizan actualizaciones de peso después de procesar solamente un pequeño subconjunto de ejemplos de entrenamiento. Este comportamiento ha dado lugar a oscilaciones pronunciadas en la función de pérdida a lo largo de las épocas, ya que el modelo tiende a ajustar sus pesos de manera extremadamente sensible a las señales de audio individuales. Además, esta configuración conlleva a una convergencia más lenta en el entrenamiento del modelo, pudiendo afectar además a la capacidad de separación de fuentes.

Igualmente se ha observado, que los resultados obtenidos en relación a la métrica SN-SNRi han sido peores. Esto podría atribuirse al hecho de que se empleó la SDR como función de pérdida durante el entrenamiento de los modelos. Esto influye en que los modelos se han

centrado en mejorar la relación entre la señal de interés (fuente de audio) y la distorsión o error generados por el procesamiento de señales, en lugar de enfocarse en otros aspectos de la calidad del audio.

También podríamos llegar a la conclusión de que, en términos generales, los modelos Conv-TasNet han mostrado un rendimiento superior a los modelos DPRNN en lo que respecta a la métrica SDRi. Sin embargo, se observa una tendencia opuesta en la métrica SN-SNRi, aunque esta diferencia no es tan clara. Como se ha mencionado antes esto podría haber sido influenciado por la elección de la métrica SDR como función de pérdida. No obstante en nuestro caso podría afirmarse que los modelos ConvTasNet han superado a los modelos DPRNN. Esta afirmación podría reforzarse con la comparación de estos modelos en otros contextos, utilizando diferentes funciones de pérdida o evaluando métricas alternativas.

6. Conclusiones

Durante este trabajo, se ha logrado la implementación de dos arquitecturas de modelos de separación de fuentes, ConvTasNet y DPRNN. Estos modelos fueron entrenados y evaluados en diversos conjuntos de datos de efecto cóctel. Se ha observado que en escenarios donde solo se encuentran los oradores, se logra una separación óptima de las fuentes. Sin embargo, en entornos con ruido de fondo, los modelos no han mostrado una eficiencia considerable, a pesar de haber probado diferentes métodos de entrenamiento para abordar el ruido presente en ellos.

Además, se ha comprobado que la limitación de la memoria asociada a la GPU utilizada para el desarrollo de este trabajo ha sido un factor que ha condicionado los resultados obtenidos influyendo en la convergencia de los modelos así como en su evaluación.

También se ha constatado que los modelos ConvTasNet muestran un desempeño ligeramente superior en comparación con los modelos DPRNN. No obstante, sería aconsejable realizar un análisis más detallado de estos modelos mediante la utilización de un tamaño de lote mayor, así como empelando distintas funciones de pérdida. Esta estrategia podría facilitar una evaluación más precisa de su eficacia en una variedad de condiciones y escenarios.

7. Lineas futuras

Las futuras mejoras en los métodos de superación e fuentes de la separación de fuentes de audio podrían incluir dos características: la consideración del contexto espacial y la adaptación a diversos escenarios. Estas líneas de investigación permitirían transformar la manera

en que abordamos la separación de fuentes en entornos complejos y dinámicos.

La consideración del Contexto Espacial implica la integración de información sobre la ubicación de micrófonos y altavoces en el proceso de separación. En un entorno de cocktail party con múltiples fuentes sonoras, el conocimiento de la disposición de estos dispositivos puede suponer una ventaja a la hora de llevar a cabo la separación de fuentes. Por ejemplo, un micrófono más cercano a una fuente de interés debería tener un peso mayor en la separación. Además, la acústica de la sala y la reflexión del sonido pueden ser consideradas para mejorar la reverberación. Al combinar la información espacial con técnicas de procesamiento de señales y redes neuronales, podemos crear sistemas más robustos y precisos para separar fuentes en condiciones de ruido y alboroto acústico.

También podría considerarse el desarrollo de algoritmos que permitan la adaptación automática a diferentes escenarios y condiciones de grabación. La realidad es que no todos los entornos son iguales, y las condiciones pueden variar drásticamente. Esto incluye cambios en el número de fuentes sonoras, variaciones en la acústica de la sala, o la presencia de ruido ambiental.
Para abordar esto, se podrían desarrollar algoritmos capaces de identificar y ajustarse a las condiciones en tiempo real. Esto podría incluir técnicas de aprendizaje automático que supervisen y ajusten constantemente los parámetros de separación, o la creación de modelos que sean lo suficientemente flexibles como para adaptarse a distintas situaciones.

También sería recomendable evaluar la capacidad de los modelos en contextos del mundo real, como dispositivos auditivos. Además, la latencia de los resultados también es un factor clave, ya que cualquier retraso en la señal procesada puede afectar negativamente la experiencia del usuario. Por lo tanto, la investigación futura debe abordar la capacidad de los modelos en tiempo real y la minimización de la latencia en dispositivos auditivos.

Referencias

[Agrawal et al., 2023] Agrawal, J., Gupta, M., and Garg, H. (2023). A review on speech separation in cocktail party environment: challenges and approaches. *Multimedia Tools and Applications*, 82(20):31035–31067.

[Bronkhorst, 2015] Bronkhorst, A. W. (2015). The cocktail-party problem revisited: early processing and selection of multi-talker speech. *Attention, Perception & Psychophysics*, 77(5):1465–1487.

[Chandna, 2016] Chandna, P. (2016). Audio Source Separation Using Deep Neural Networks.

[Cherry, 2005] Cherry, E. C. (2005). Some Experiments on the Recognition of Speech, with One and with Two Ears. *The Journal of the Acoustical Society of America*, 25(5):975–979.

[Cui et al., 2019] Cui, X., Zheng, K., Gao, L., Yang, D., and Ren, J. (2019). Multiscale spatial-spectral convolutional network with image-based framework for hyperspectral imagery classification. *Remote Sensing*, 11:2220.

[Fischer et al., 2020] Fischer, T., Caversaccio, M., and Wimmer, W. (2020). Multichannel acoustic source and image dataset for the cocktail party effect in hearing aid and implant users. *Scientific Data*, 7(1):440. Number: 1 Publisher: Nature Publishing Group.

[Gidlöf, 2023] Gidlöf, A. (2023). *Evaluation of Methods for Sound Source Separation in Audio Recordings Using Machine Learning*.

[Hossain et al., 2020] Hossain, D., Imtiaz, M., Ghosh, T., Raju, V., and Sazonov, E. (2020). Real-time food intake monitoring using wearable egocnetric camera. volume 2020, pages 4191–4195.

[Isik et al., 2016] Isik, Y., Roux, J. L., Chen, Z., Watanabe, S., and Hershey, J. R. (2016). Single-Channel Multi-Speaker Separation using Deep Clustering. arXiv:1607.02173 [cs, stat].

[Jiménez Bohmer, 2022] Jiménez Bohmer, N. A. (2022). Separación de fuentes de audio con Deep Learning para la interacción humano-robot. Accepted: 2022-05-18T15:38:12Z Publisher: Universidad de Chile.

[Kolbæk et al., 2017] Kolbæk, M., Yu, D., Tan, Z.-H., and Jensen, J. (2017). Multi-talker Speech Separation with Utterance-level Permutation Invariant Training of Deep Recurrent Neural Networks. arXiv:1703.06284 [cs, eess].

[Luo et al., 2020] Luo, Y., Chen, Z., and Yoshioka, T. (2020). Dual-path RNN: efficient long sequence modeling for time-domain single-channel speech separation. arXiv:1910.06379 [cs, eess].

[Luo and Mesgarani, 2018] Luo, Y. and Mesgarani, N. (2018). TaSNet: Time-Domain Audio Separation Network for Real-Time, Single-Channel Speech Separation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 696–700. ISSN: 2379-190X.

[Luo and Mesgarani, 2019] Luo, Y. and Mesgarani, N. (2019). Conv-TasNet: Surpassing Ideal Time-Frequency Magnitude Masking for Speech Separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(8):1256–1266. arXiv:1809.07454 [cs, eess].

[Makino, 2018] Makino, S., editor (2018). *Audio Source Separation*. Signals and Communication Technology. Springer International Publishing, Cham.

[McDermott, 2009] McDermott, J. H. (2009). The cocktail party problem. *Current Biology*, 19(22):R1024–R1027. Publisher: Elsevier.

[Pariente et al., 2020] Pariente, M., Cornell, S., Cosentino, J., Sivasankaran, S., Tzinis, E., Heitkaemper, J., Olvera, M., Stöter, F.-R., Hu, M., Martín-Doñas, J. M., Ditter, D., Frank, A., Deleforge, A., and Vincent, E. (2020). Asteroid: the PyTorch-based audio source separation toolkit for researchers. arXiv:2005.04132 [cs, eess].

[Rebala et al., 2019] Rebala, G., Ravi, A., and Churiwala, S. (2019). Deep Learning. In Rebala, G., Ravi, A., and Churiwala, S., editors, *An Introduction to Machine Learning*, pages 127–140. Springer International Publishing, Cham.

[Rodíguez Serrano, 2014] Rodíguez Serrano, F. (2014). *Separación de fuentes sonoras en señales acústicas*. http://purl.org/dc/dcmitype/Text, Universidad de Jaén.

[Sonning et al., 2020] Sonning, S., Schüldt, C., Erdogan, H., and Wisdom, S. (2020). Performance Study of a Convolutional Time-Domain Audio Separation Network for Real-Time Speech Denoising. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 831–835. ISSN: 2379-190X.

[Szabó et al., 2016] Szabó, B. T., Denham, S. L., and Winkler, I. (2016). Computational Models of Auditory Scene Analysis: A Review. *Frontiers in Neuroscience*, 10.

[Vincent et al., 2006] Vincent, E., Gribonval, R., and Fevotte, C. (2006). Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech,*

and Language Processing, 14(4):1462–1469. Conference Name: IEEE Transactions on Audio, Speech, and Language Processing.

[Yamashita et al., 2018] Yamashita, R., Nishio, M., Do, R. K. G., and Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*, 9(4):611–629. Number: 4 Publisher: SpringerOpen.

[Yu et al., 2017] Yu, D., Kolbæk, M., Tan, Z.-H., and Jensen, J. (2017). Permutation Invariant Training of Deep Models for Speaker-Independent Multi-talker Speech Separation. arXiv:1607.00325 [cs, eess].

[Zhang, 2021] Zhang, J. (2021). Music feature extraction and classification algorithm based on deep learning. *Scientific Programming*, 2021:1–9.