

# **HYBRID TRANSFORMERS FOR MUSIC SOURCE SEPARATION**

**Simon Rouard, Francisco Massa,  
Alexandre D'efossez**

**Meta AI**

**Presentation by: Pamudu Ranasinghe**





# PRESENTATION OUTLINE

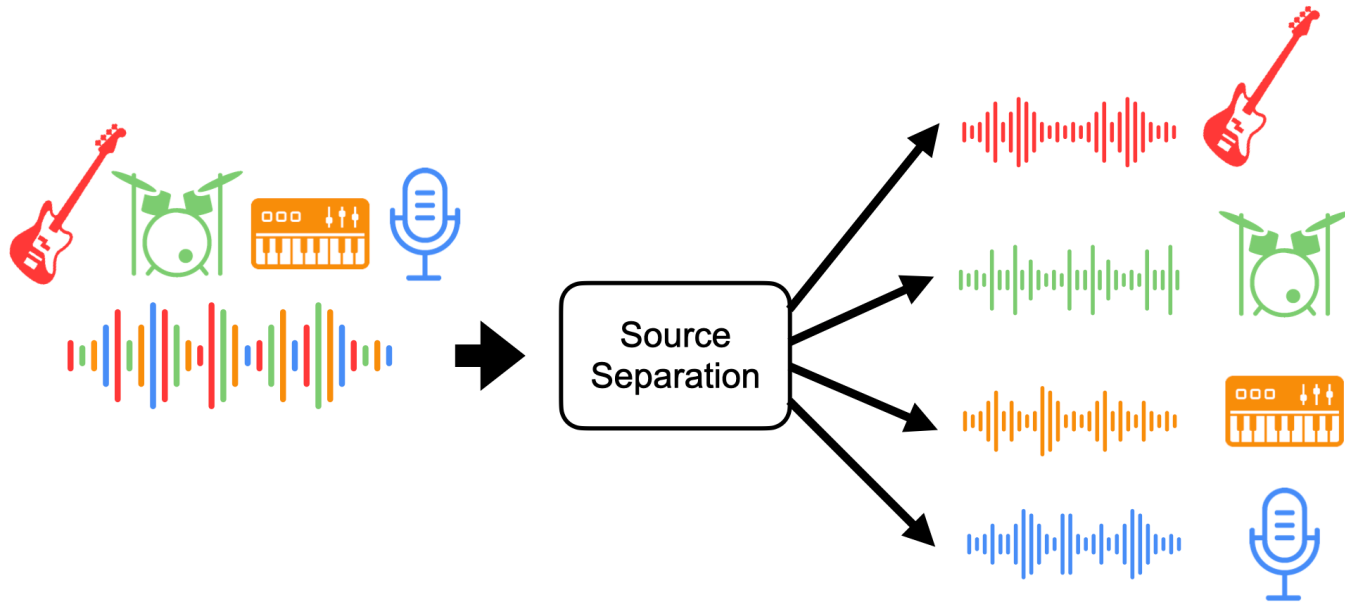
- **Introduction**
- **Related Work**
- **Architecture**
- **Demo**
- **Dataset Preparation**
- **Experimental Setup**
- **Results**
- **Conclusion**



# INTRODUCTION - MSS

## What is Music source separation (MSS) ?

- Music Source Separation (MSS) is the process of isolating individual sound sources from a mixed audio signal.



Mathematically

$$y(t) = \sum_{n=1}^N x_n(t)$$

- $y(t)$  represents the observed mixture signal
- $x_n(t)$  represents the  $n$ -th source signal
- $N$  is the total number of source signals

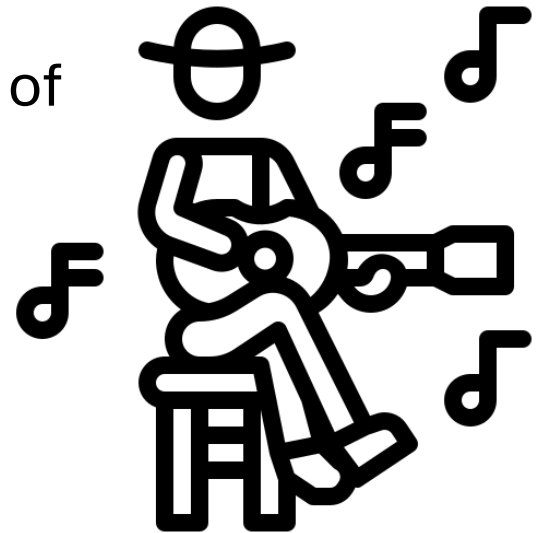
- In this research, researchers separated mixed signals into ***drums, bass, vocals, and other components.***

# INTRODUCTION - MSS

---

## Why is MSS important?

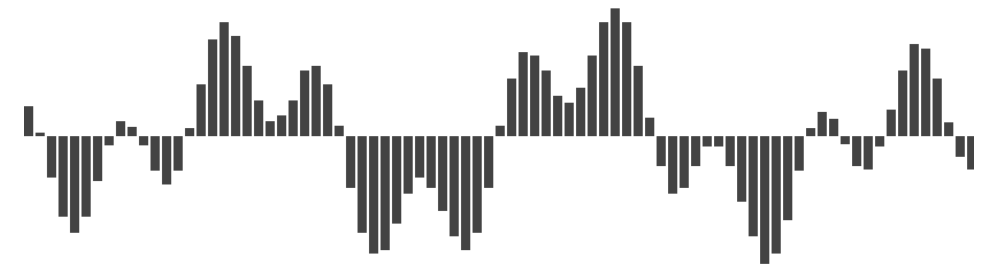
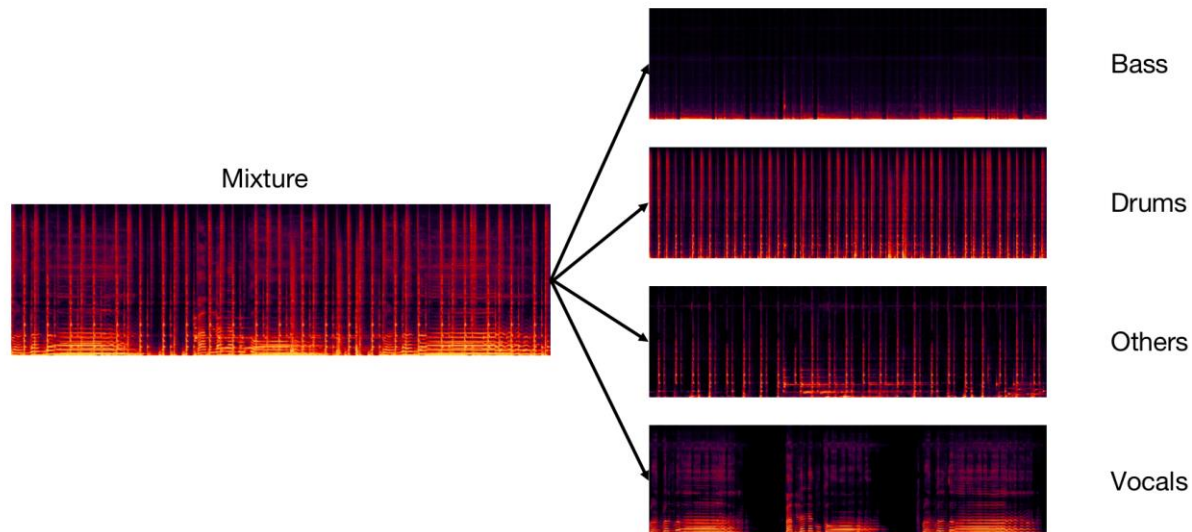
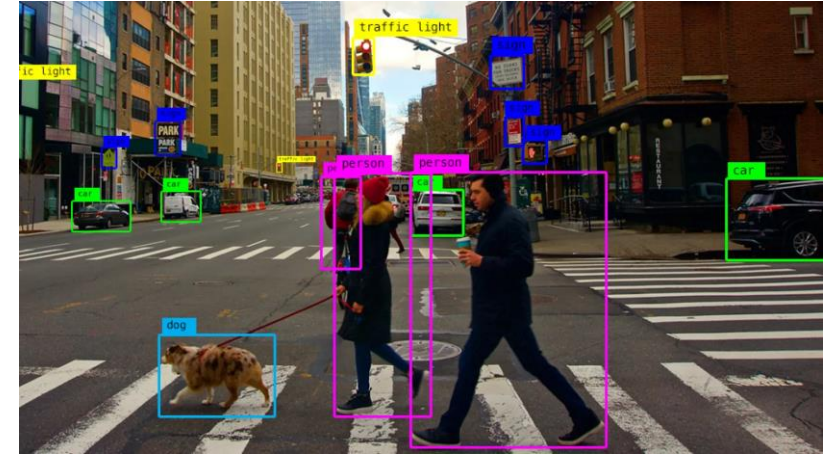
- Allows **musicians** to isolate instruments for practice.
- Helps **DJs and producers** remix music more easily.
- Enables **karaoke** by removing vocals.
- Improves **speech recognition** in noisy environments.
- Customizes the music experience by adjusting the balance of instruments and vocals.



# INTRODUCTION - MSS

## Why is it difficult?

- In a song, multiple instruments **overlap in time and frequency**.
- Some sounds, like background guitars and synths, are **blended together**.
- Unlike images, where objects are clearly separate, audio sources are **mixed into one waveform**.



# INTRODUCTION - OVERVIEW

## Methods of Music Source Separation

- **Spectrogram-based** - Work with frequency information like an image.
- **Waveform-based** - Process the raw sound directly.
- **Hybrid method** - Combine both Spectrograms and waveforms

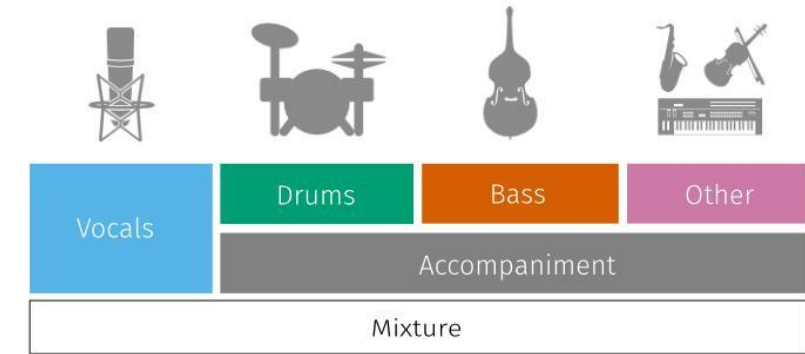
## Impact of Input Length in Music Source Separation

Short Context	Long Context
Captures local features & fast-changing sounds.	Captures broader structure & dependencies.
More computationally efficient.	Requires more memory & resources.
<b>Example:</b> Conv-Tasnet (1 sec).	<b>Example:</b> Demucs (up to 10 sec).

# INTRODUCTION - OVERVIEW

## MUSDB18 Dataset

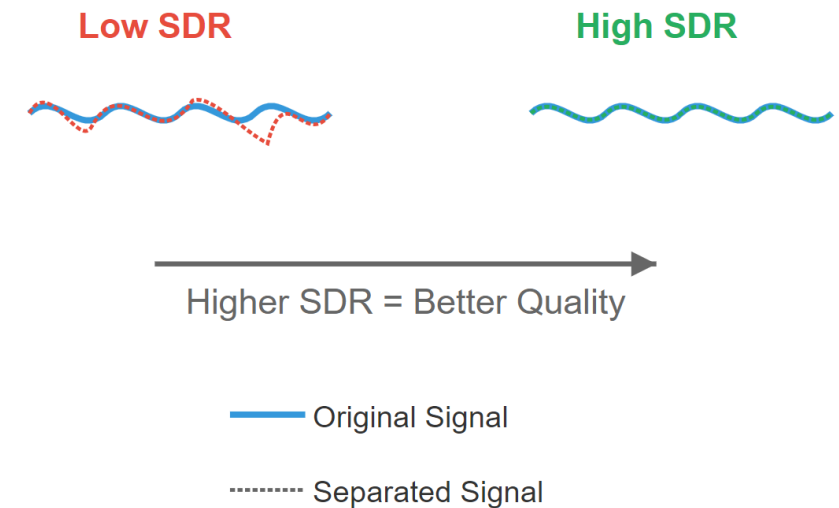
- MUSDB18 is a **reference dataset** for benchmarking Music Source Separation (MSS).
- 150 songs (87 for training)
- This dataset is relatively small compared to datasets used in other deep learning tasks.



## Evaluation in MSS

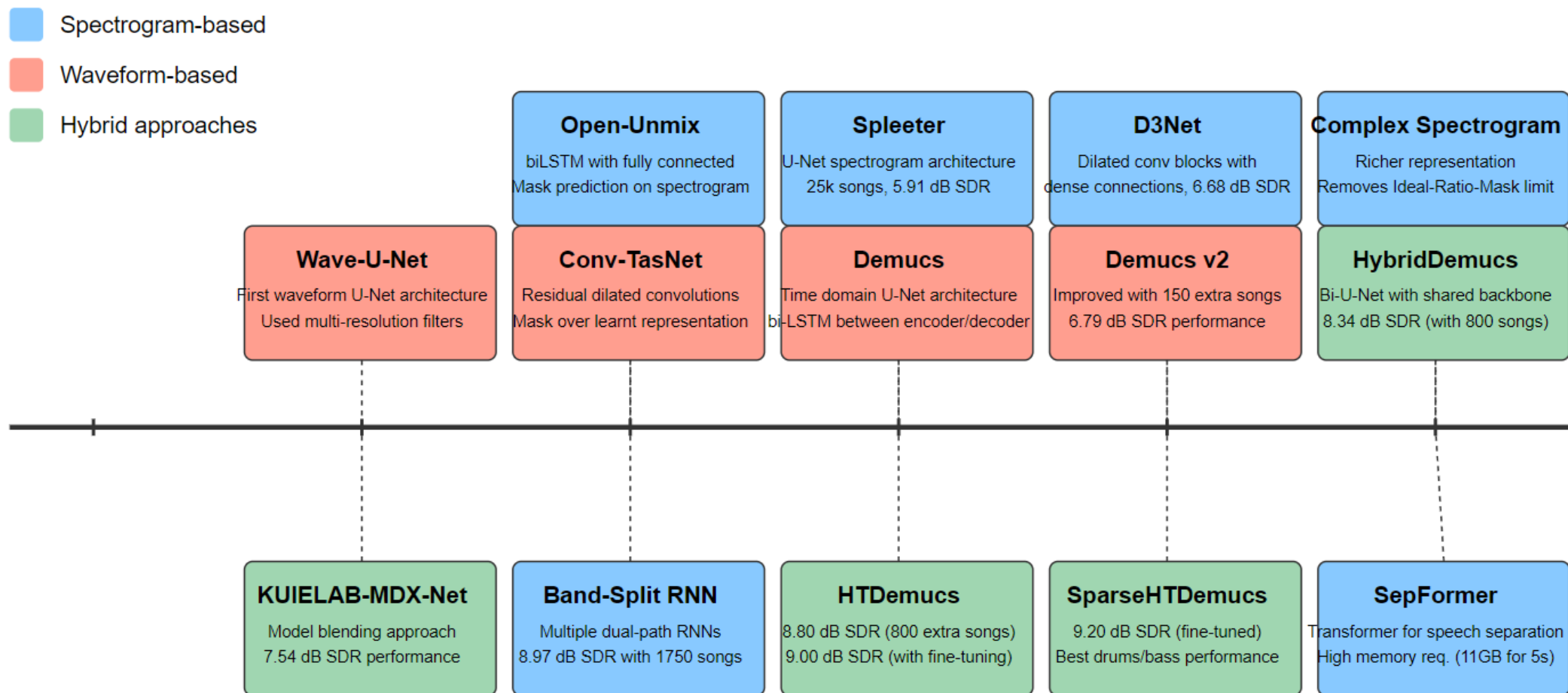
### Signal-to-Distortion Ratio (SDR)

- It measures how much-desired sound remains compared to noise or distortion.
- A higher number means the separated sound is closer to the original, with fewer distracting artifacts.





# RELATED WORK – MODEL ROADMAP



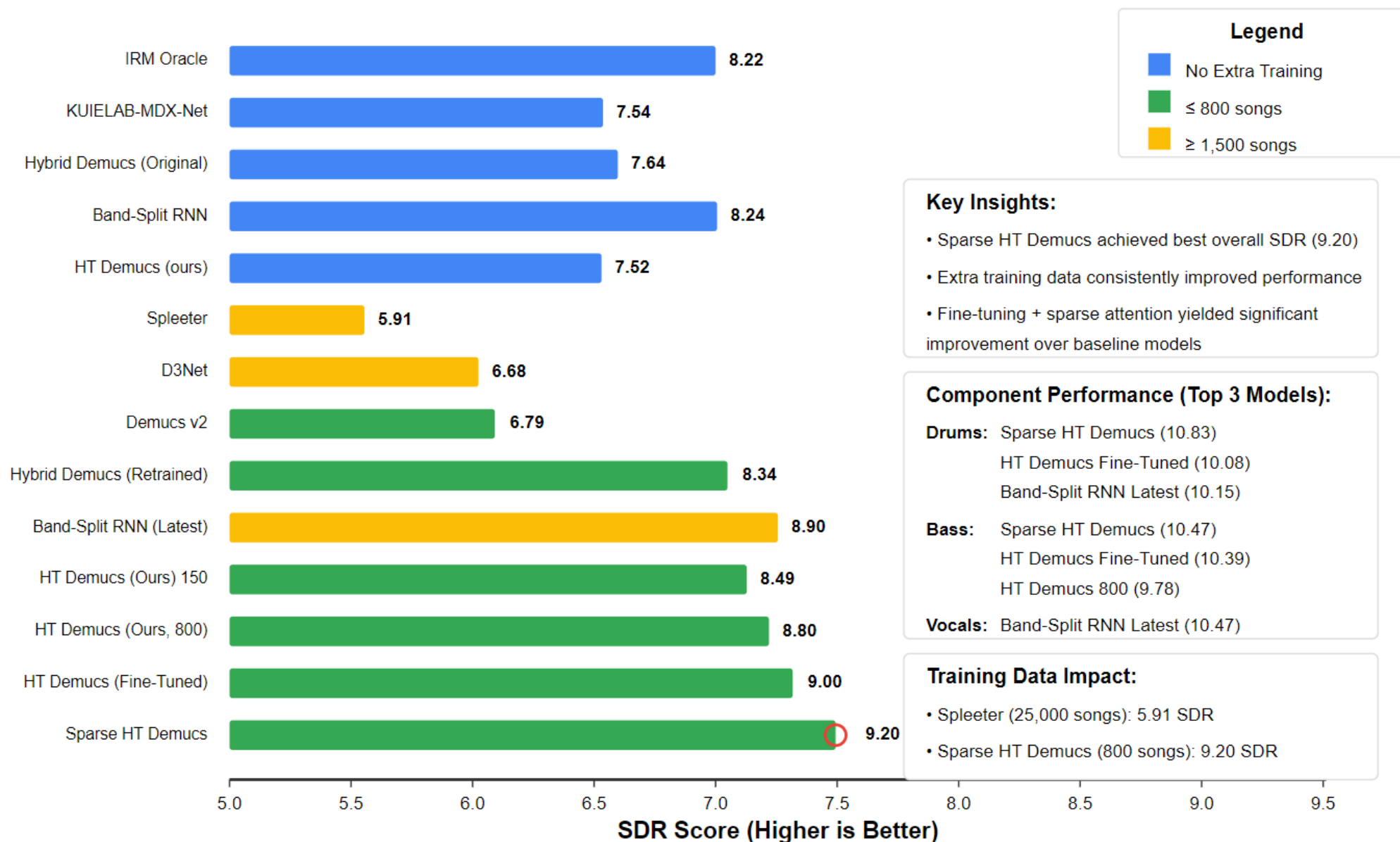
## Performance Metrics

SDR = Signal-to-Distortion Ratio (Higher is better)

MUSDB: Standard dataset for MSS evaluation

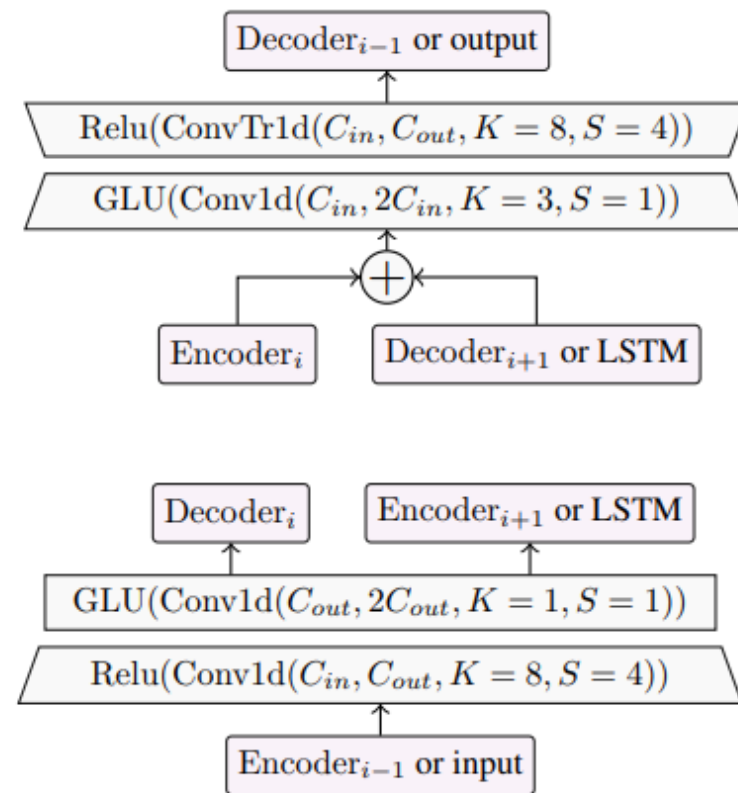
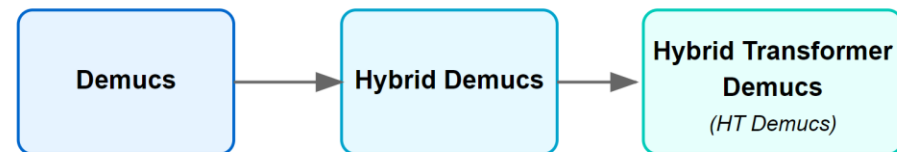
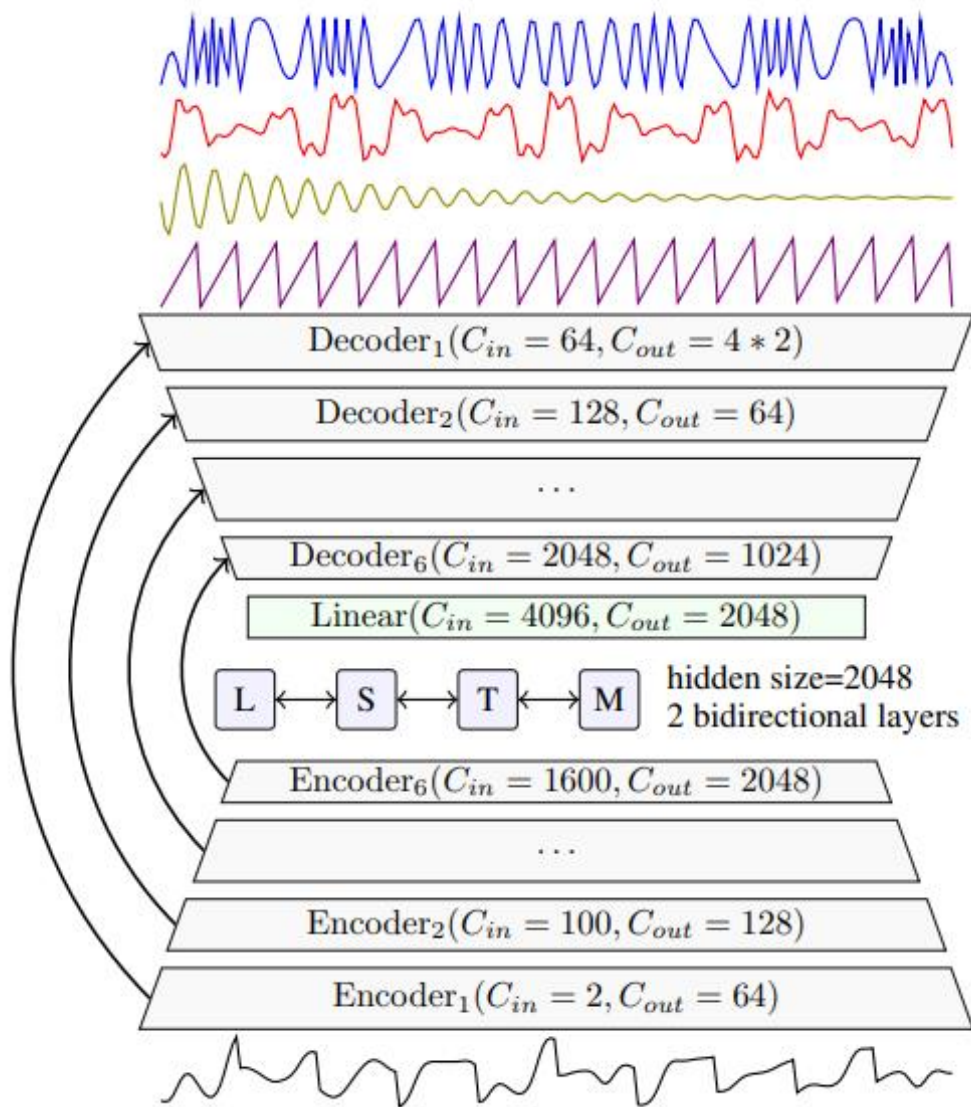


# RELATED WORK – REVIEW OF EXISTING MODELS

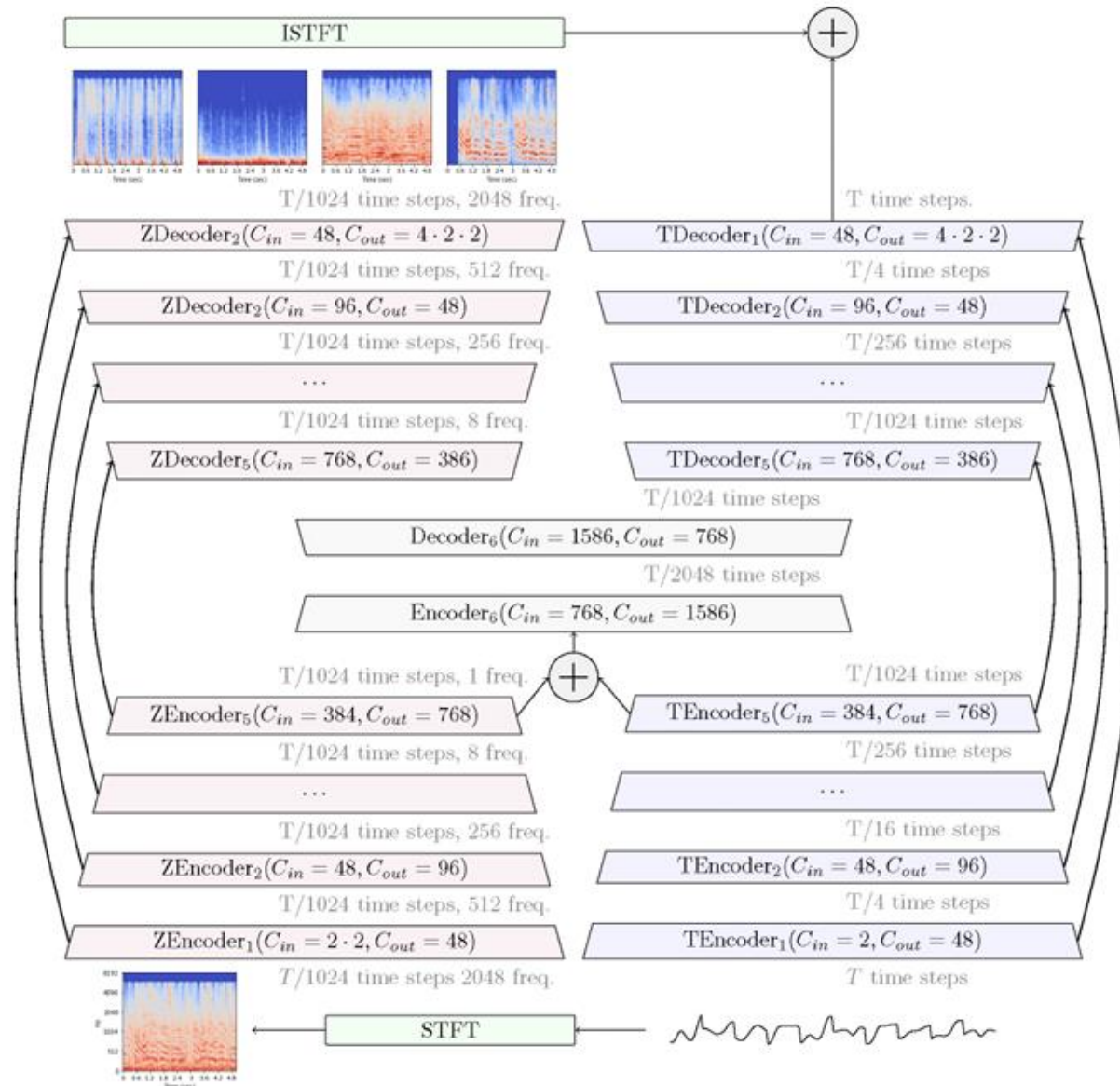


# ARCHITECTURE - DEMUCS

## Deep Extractor for Music Sources

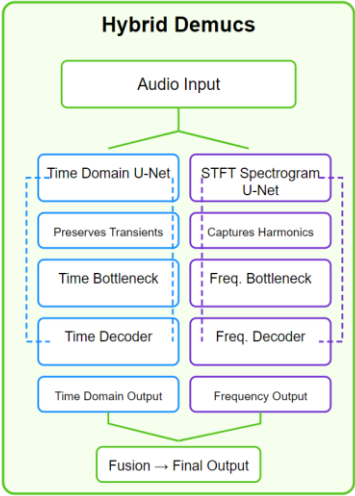
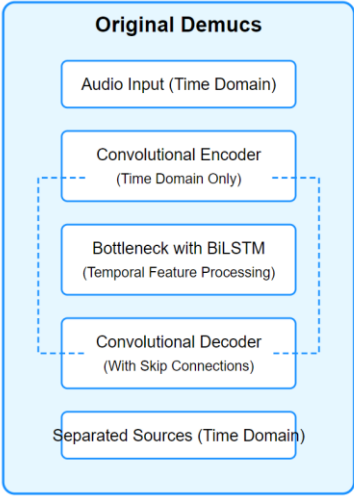


# ARCHITECTURE – HYBRID DEMUCS



- Hybrid Demucs extend the original Demucs by adding a **spectrogram-domain branch** alongside the time-domain branch
- By using a 2D U-Net in the frequency domain and merging its output with the time-domain branch

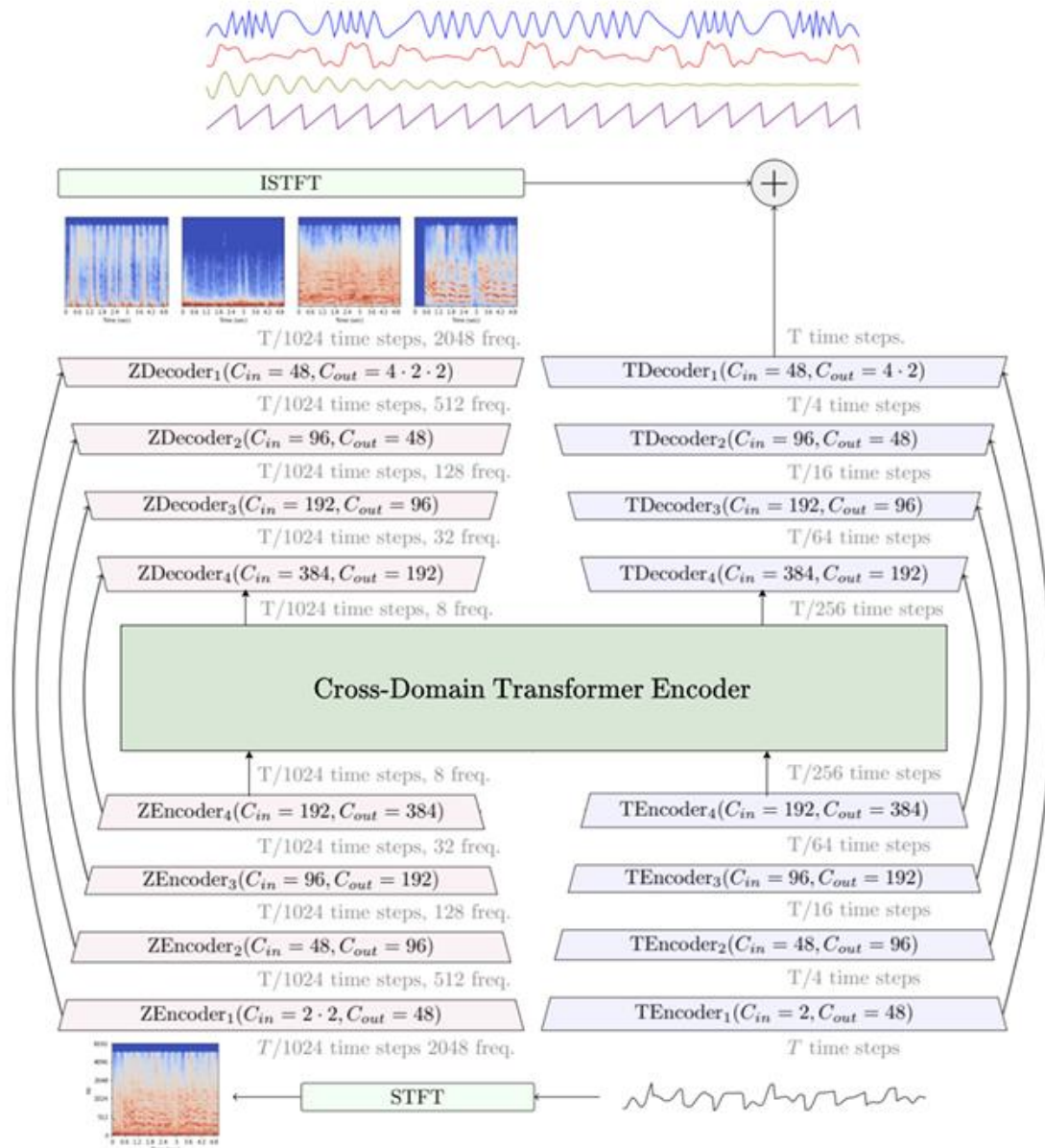
# ARCHITECTURE – DEMUCS VS HYBRID DEMUCS



Feature	Original Demucs	Hybrid Demucs
Domain of Operation	Operates entirely in the time domain	Uses a dual-domain approach: time domain and frequency domain
Architecture	Single U-Net encoder-decoder with BiLSTM bottleneck	Two parallel U-Nets: one for waveform, one for spectrogram
Bottleneck Processing	BiLSTM for long-range temporal features	Separate processing in each branch for complementary information
Skip Connections	Standard U-Net skip connections	Skip connections in both branches, outputs later merged
Strengths	Excellent at preserving phase details and handling transient-rich signals	Combines strengths of both domains for improved overall separation quality
Complexity and Efficiency	Simpler, less computationally demanding	More complex and resource-intensive due to dual-branch design
Output Formation	Directly synthesizes time-domain waveform per source	Produces two separate outputs that are then fused
Performance	Competitive separation quality; may struggle with harmonic context	Higher SDR, reduced artifacts, benefits from extra training data

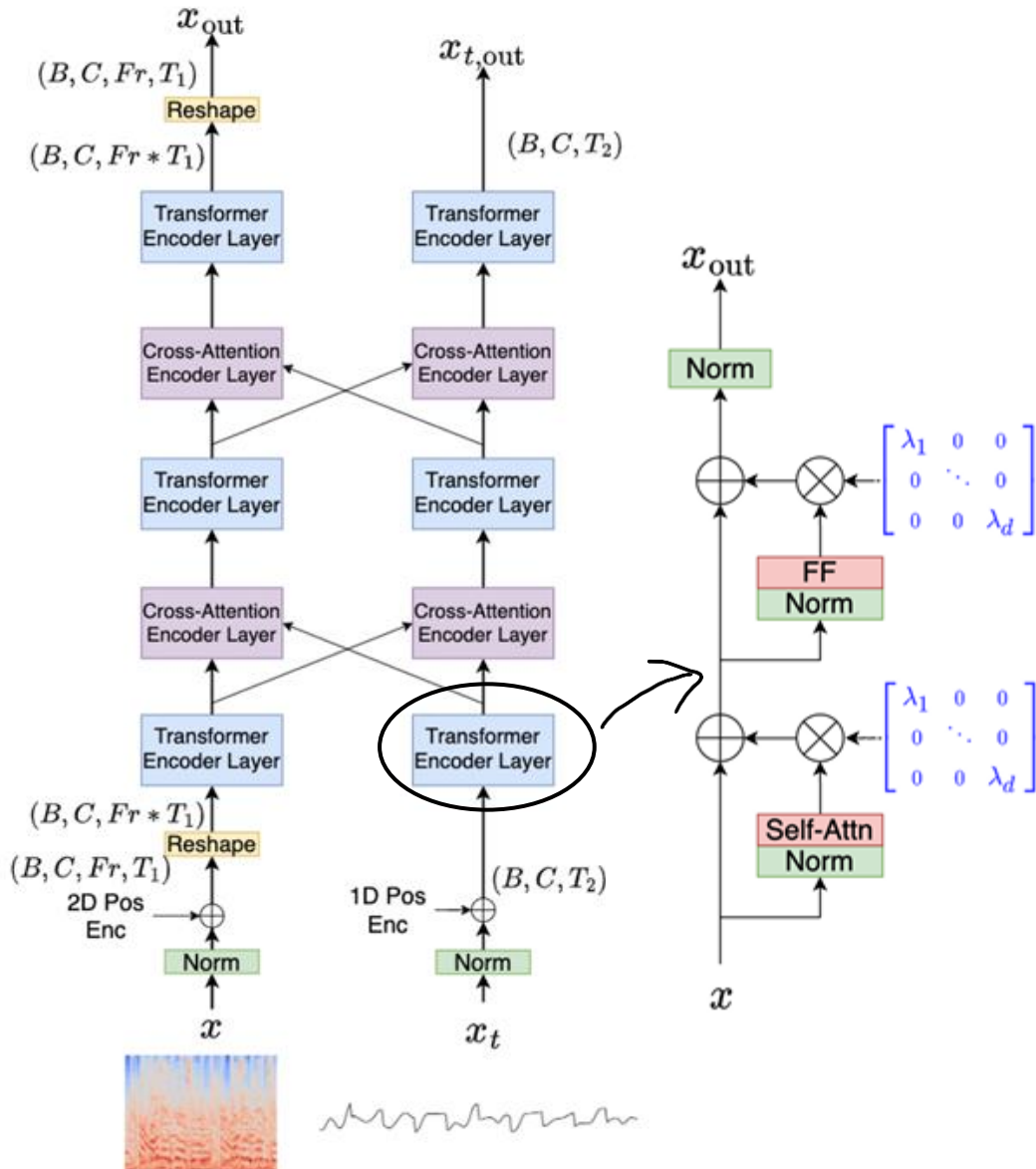


# ARCHITECTURE – HYBRID TRANSFORMER DEMUCS



- **Dual Branches:** Processes waveforms and spectrograms in parallel.
- **Cross-Domain Transformer:** Uses interleaved self- and cross-attention to fuse representations.
- **Output Fusion:** Sums the time-domain output with iSTFT-converted spectrogram output for separation.

# ARCHITECTURE – HYBRID TRANSFORMER DEMUCS



## Attention Blocks:

- Self-Attention: Operates within each domain independently.
- Cross-Attention: Allows features from one domain to attend to features from the other.

## Positional Encodings:

- 1D sinusoidal encodings for waveform features.
- 2D sinusoidal encodings (flattened) for spectrogram features.

## Model Dimensions:

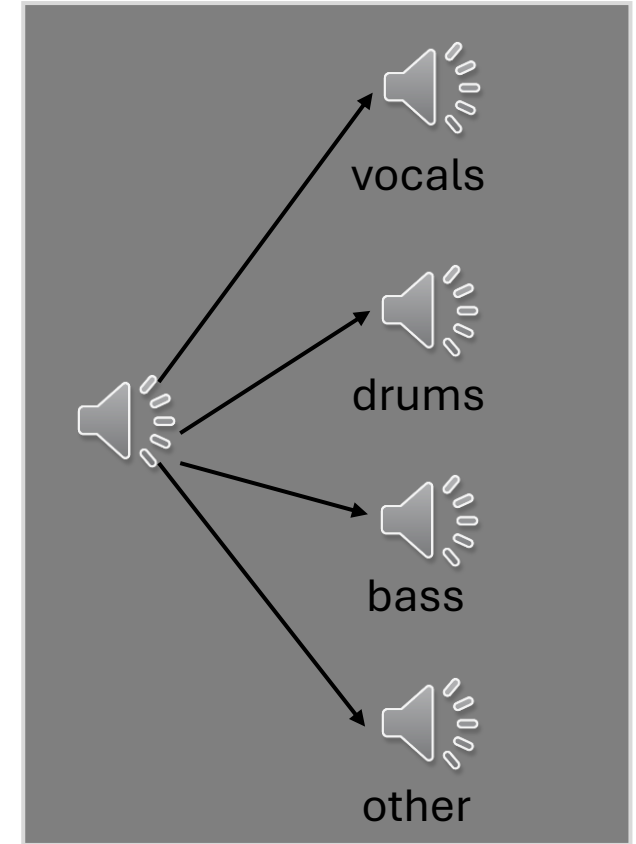
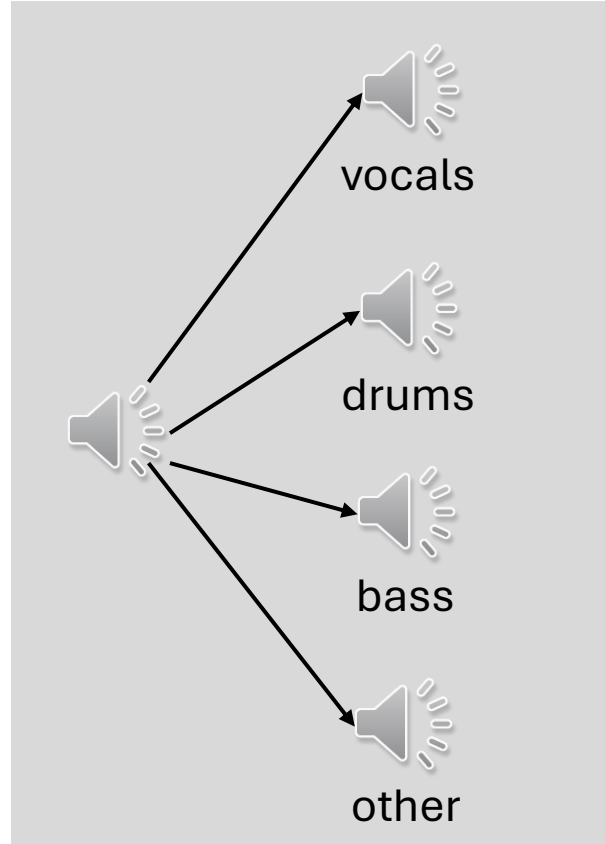
- Transformer dimension
- Feedforward network

## Normalization & Stability:

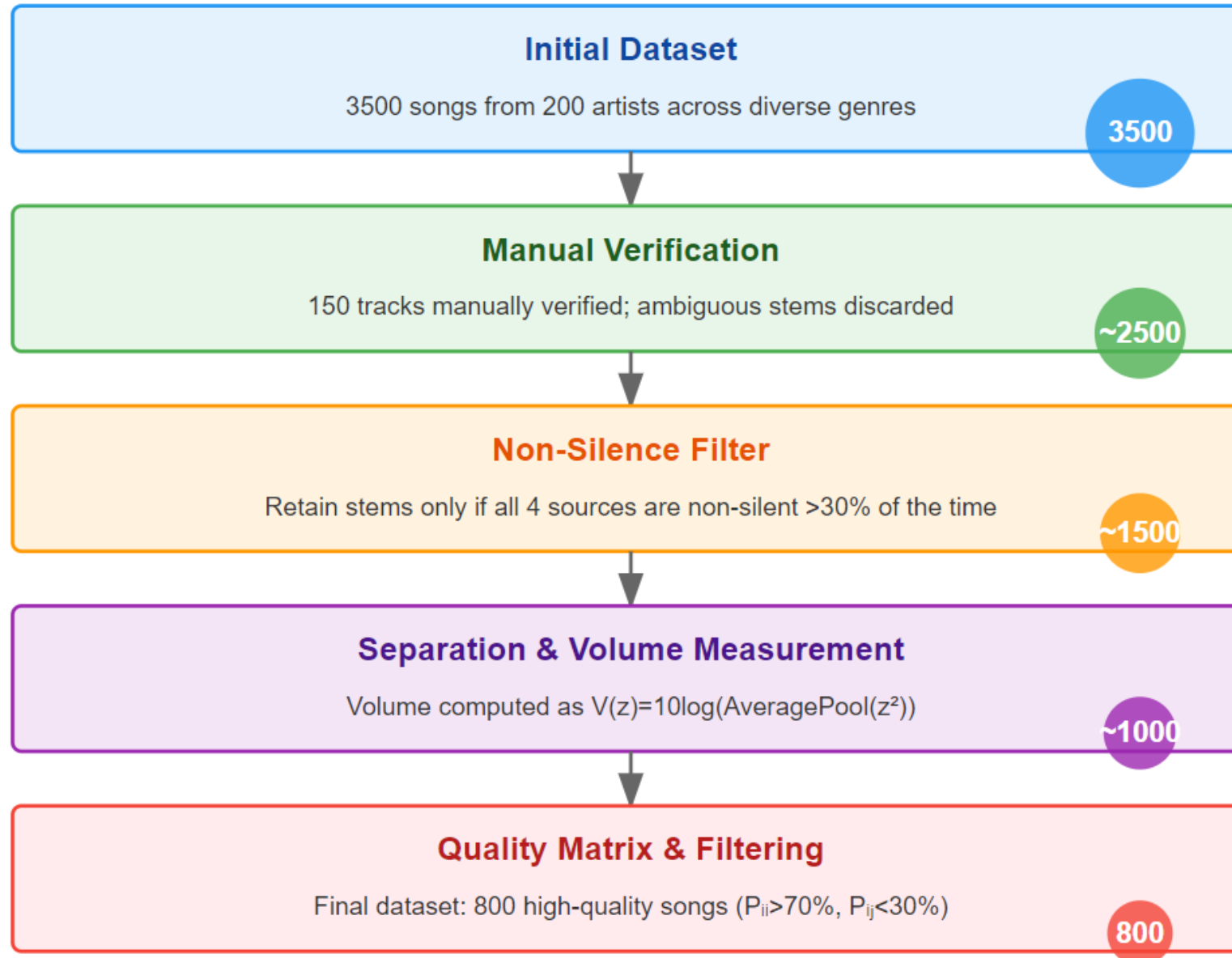
- Layer normalization is applied.
- LayerScale is used

# DEMO

- The code repository is **publicly available** with the trained weights.  
[\[CODE\]](#)
- I was able to reproduce the results on Google Colab.



# DATASET PREPARATION





# DATASET PREPARATION

## Quality Matrix Definition

$P_{(ij)}$ : Proportion of segments where

$$V(y_{(ij)}) - V(x_{(i)}) > 10 \text{ dB}$$

$$\text{where } y_{(ij)} = f(x_{(i)})_j$$

$$V(z) = 10\log_{10}(\text{AveragePool}(z^2))$$

## Ideal Matrix (Perfect Separation)

	vocals	drums	bass	other
vocals	100%	0%	0%	0%
drums	0%	100%	0%	0%
bass	0%	0%	100%	0%
other	0%	0%	0%	100%

## Acceptable Matrix

78%	24%	12%	25%
18%	75%	19%	28%
15%	17%	82%	21%
22%	26%	14%	71%

Retained:  $P_{ii} > 70\%$ ,  $P_{ij} < 30\%$

## Rejected Matrix

75%	22%	15%	42%
25%	62%	20%	27%
18%	29%	80%	19%
38%	26%	22%	58%

Rejected: Some  $P_{ii} < 70\%$  or some  $P_{ij} > 30\%$

# EXPERIMENTAL SETUP

## Hardware Configuration

32 GB

32 GB

32 GB

32 GB

32 GB

32 GB

32 GB

32 GB

8x NVIDIA V100 GPUs

## Training Parameters

L1 Loss

Adam Optimizer

LR:  $3 \times 10^{-4}$

Batch Size: 32

1200 Epochs  $\times$  800 Batches per Epoch

## Dataset Configuration

MUSDB18-HQ Dataset  
Professional Multitrack Recordings

+

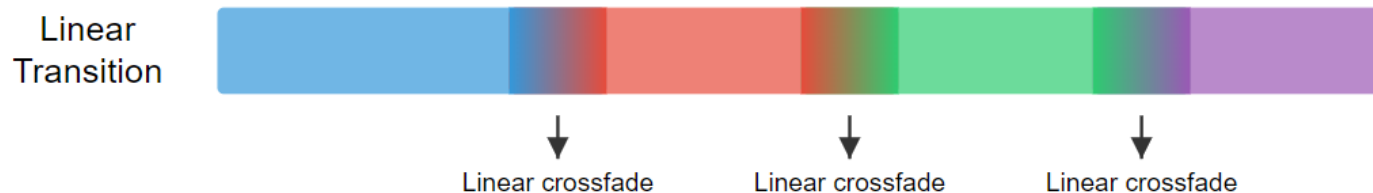
Custom 800 Songs Dataset  
Curated Collection

44.1kHz Sampling Rate, Stereo Audio

## Additionally

- 8 songs were used for validation.
- Researchers claim that using **one model per source can be beneficial, although it adds overhead during both training and evaluation.**

# EXPERIMENTAL SETUP



## Stage 1: Fixed-length Chunks

The audio is divided into equal-length.

## Stage 2: Overlap Applied

Each overlap region represents 25% of the chunk duration.

## Stage 3: Linear Transition

In the overlap regions, a linear crossfade is applied.

This means that as we progress through the overlap region:

- The first chunk's contribution gradually decreases from 100% to 0%
- The second chunk's contribution gradually increases from 0% to 100%
- This creates a smooth transition between chunks

# RESULTS - COMPARISON WITH THE BASELINES

## Color Legend:

- Baseline (Original Hybrid Demucs)
- Band-Split RNN (Best for Other/Vocals)
- Sparse HT Demucs (Best Overall)

Model	Extra Data	SDR (Overall)	Drums	Bass	Other	Vocals
Hybrid Demucs (Original)	No	7.64	8.12	8.43	5.65	8.35
Hybrid Demucs (Retrained)	800 songs	8.34	9.31	9.13	6.18	8.75
Band-Split RNN (Latest Version)	1,750 extra songs	8.97	10.15	8.16	7.08	10.47
HT Demucs (Ours)	800 songs	8.80	10.05	9.78	6.42	8.93
<b>Sparse HT Demucs</b> (Fine-Tuned + Sparse Attention)	800 songs	<b>9.20</b>	<b>10.83</b>	<b>10.47</b>	6.41	9.37

+1.56 dB  
+2.71 dB  
+2.04 dB  
-0.67 dB  
-1.10 dB

- **Sparse HT Demucs model achieves the best overall SDR (9.20 dB).**
- The **Band-Split RNN remains highly competitive** in Other (7.08 dB) and Vocals (10.47 dB) categories, despite using a different architectural approach.











# RESULTS - IMPACT OF HYPER-PARAMETERS

dur. (sec)	depth	dim.	nb. param.	RTF (cpu)	SDR (All)
3.4	5	384	26.9M	1.02	8.17
↓ 7.8	↓ 7	384	34.0M	1.23	8.26
	↓ 5	512	41.4M	1.30	8.12
	5	384	26.9M	1.49	8.70
↓ 12.2	↓ 7	384	34.0M	1.68	OOM
	↓ 5	512	41.4M	1.77	8.80
	5	384	26.9M	2.04	OOM

## Architectural hyperparameters on model performance




- **Duration** (in seconds) of the training excerpts.
- **Depth** of the transformer encoder.
- **Dimension** of the transformer.

\***RTF (Real Time Factor)** - Time to process fixed audio input (40 seconds of Gaussian noise) divided by the input duration.

 Duration: 3.4 sec	 Duration: 7.8 sec	 Duration: 12.2 sec	 Best SDR score
 Depth: 5	 Depth: 7	 Duration change	 Depth change

OOM = Out of Memory

# RESULTS - IMPACT OF DATA AUGMENTATION

dur. (sec)	depth	remixing	repitching	SDR (All)	
7.8	5	✓	✓	8.70	
7.8	5	✓	✗	8.65	
7.8	5	✗	✓	8.00	

## Configuration Performance:

 Enabled (✓)       Disabled (✗)

## Remixing

- Recombining stems (vocals, drums, bass) from different songs during training.
- Adjusting relative volumes of stems to create diverse audio mixes.

## Repitching

- Altering pitch and tempo of audio stems.
- Speeding up or slowing down songs during training.

# RESULTS - IMPACT SPARSE KERNELS AND FINE-TUNING



Model Parameters	
Parameter	Value
Depth	7 (increased from 5)
Train Segment Duration	12.2 seconds
Dimension	512

# CONCLUSION

---

## Potential Improvements

- **Knowledge Distillation** to create smaller, faster models for near real-time applications
- **Unsupervised or Weakly Supervised learning techniques** to improve generalization capability (Ex: wav2vec)

## Key Insights from HT Demucs Model

- Similar architectures in generative AI are used to fuse multiple modalities
- Vision Transformer (ViT) and Swin Transformer models in computer vision are inspired by cross-domain architectures used in audio processing.



