Université de BLIDA 1
Faculté des Sciences
Département d'Informatique

MASTER THESIS

Option: Ingénierie des Logiciels

LANGUAGE-QUERIED AUDIO SOURCE SEPARATION

By:
BENLAOUBI Chaima Nour el Houda & KHETTAL Mounia

In front of a jury composed of :

Mr. KAMECHE Abdallah Hicham	President
Mr. CHETTAT Merouane	Examiner
Ms. YKHLEF Hadjer	Supervisor

2024/2025

Abstract

Language-queried audio source separation (LASS) enables on-demand sound extraction of sound sources using natural language queries overcoming limitations in traditional audio source separation systems. In this work, we propose a language-queried audio source separation architecture integrating two major innovations: **a cross attention driven ResUNet++** with multi scale receptive fields (via Atrous Spatial Pyramid Pooling), channel wise attention(Squeeze and Excitation block) and residual connections to integrate FLAN-T5 text embedding with audio features; **Cosine similarity filtering** to suppress overly similar mixture target pairs that might hinder the training. We trained our model on Clotho dataset derived mixtures and evaluated on its test set using state of the art metrics. Our system achieves good separation quality with an SDR of 2.41 and SDRi of 8.37. This work presents a lightweight, efficient framework for language-queried audio source separation compared to current state of the art models.

Keywords: Language-queried audio source separation, Cross-Modal Attention, ResUNet++, Cosine similarity filtering, Phase-aware reconstruction, computational efficiency.

Résumé

La séparation de sources audio avec requête en langage naturel (LASS) permet l’extraction de sources sonores à l’aide de requêtes textuelles, surmontant ainsi les limitations des systèmes de séparation traditionnels. Cependant, les modèles existants souffrent toujours d'inefficacité de calcul. Nous proposons une architecture de séparation de sources audio par requête en langage naturel, intégrant deux innovations majeurs : un **ResUNet++** avec **attention croisée intégrée**, des champs réceptifs multi échelle (Atrous Spatial Pyramid Pooling), une attention par canal (block squeeze et excitation) et des connexions résiduelles pour intégrer l’embedding, obtenu en utilisant FLAN-T5, aux caractéristiques audio; ainsi qu’un filtrage par **similarité cosinus** pour supprimer les paires mélange cible trop similaires. Nous avons entraîné notre modèle sur des mélanges dérivés de Clotho et évalué sur un jeu de données de test en utilisant des métriques d'évaluation modernes. Notre système a atteint des performances notables en séparation avec un SDR de 2.41 et SDRi de 8.37. Ce travail présente un cadre léger et efficace pour la séparation de sources audio par requête en langage naturel, comparé avec des modèles de l'état de l’art.

Mots-clés: Séparation des sources audio par requête en langage, attention croisée inter-modale, ResUNet++, filtrage par similarité cosinus, reconstruction sensible à la phase, efficacité computationnelle.

ملخص

فصل المصادر الصوتية بناءً على استعلام لغوي (LASS) يمكنك من استخراج مصدر صوتي باستخدام لغة طبيعية متجاوزا التحديات المتواجدة في الأنظمة التقليدية المستعملة لفصل الأصوات ومع ذلك فإن النماذج الحالية تعاني من ضعف الكفاءة الحسابية. نقترح نموذجا لفصل المصادر الصوتية يدمج ابتكارين رئيسيين, استعمال الشبكة **ResUNet++** مدمج معه آليه الانتباه المتقطع **cross attention** مع مجالات تقبل متعددة القياس multi scale receptive fields وذلك خلال Atrous Spatial Pyramid Pooling و انتباه موجه باستعمال Squeeze and and Excitation block و الروابط المتبقية Residual connections لتدريج تمثيل النص الناتج عن FLAN-T5 مع الخصائص الصوتية. كما استعملنا ال **Cosine similarity filtering** للتخلص من المزيجات الصوتية المشابهة لصوت المراد و التي قد تؤثر على تدريب النموذج. دربنا نموذجنا على مزيجات مشتقة من Clotho و قمنا بتقييمه على مجموعة الاختبار الخاصة به و ذلك باستعمال مقاييس معيارية حديثة. حقق نظامنا متوسط SDR قدره 2.41 د.ب ومتوسط SDRi ببقيمة 8.37 د.ب. هذا العمل يمثل نظام خفيف وفعال لفصل المصادر الصوتية بناءً على استعلام لغوي, مقارنة مع النماذج الحالية.

Acknowledgment

First and foremost, thanks and praises to Allah the Almighty, whose guidance has carried us through every step of this journey, and eased every hardship that we faced.

We would like to thank ourselves, for our constant dedication, perseverance and countless sleepless nights spent iterating on experiments and refining ideas. Our shared commitment and complementary strengths were the key to make this work possible.

We owe an immeasurable debt to our parents, whose support extended financial and emotional boundaries. Their tireless encouragement and countless sacrifices laid the very groundwork for our academic encouragement. When setbacks tested our confidence, they were there to encourage us and get us back on track. No words can fully capture the magnitude of their contributions throughout our academic journey.

We also extend our heartfelt thanks to our friends Zineb, Liza, Karima, Dallel, Aya, and Nada. Their unwavering moral support, thoughtful check-ins, and the countless moments we've shared have brought immense joy and balance to our lives.

Last but not least, we would like to express our deepest and most sincere gratitude to our research supervisor, Ms YKHLEF, for her insightful feedback and patient mentorship which have been valuable in shaping this thesis.

Table of Content

List of Figures :

List of Tables :

Introduction

Problem definition :

Audio source separation refers to the process of isolating individual sound sources from a mixture of overlapping sounds. Its primary purpose is to automate the cocktail party effect [1], the human ability to focus on a single auditory stream in a noisy environment. Audio source separation has long relied on the use of statistical methods [2] [3]. However, over the past decade, deep learning has rapidly taken over the field. Many approaches developed, each addressing specific characteristics of the source signals. For example, in music separation, models are designed to disentangle musical instruments and vocals [4] [5] [6] [7] whereas in speech separation, the focus is on extracting a speaker's voice from other speech or from noise [8] [9]. A broader category, known as universal sound separation, aims to handle a variety of sources, separating any type of source from a mixture [10].

Building on these foundational methods, recent research introduced query based audio source separation, which leverages additional modalities such as an image, an audio or a text to guide the separation of a target audio source from a mixture.

Studies have shown that employing text as the guiding modality, a process commonly known as language-queried audio source separation (LASS), showed better results to this day [11], and prove to be more efficient and practical over the other modalities [12]. Unlike earlier methods that attempt to separate multiple sources simultaneously, LASS focuses on extracting a single target source conditioned on its corresponding query, reducing the difficulty of the task considering the increasing number of sounds in real world mixtures. While previous approaches utilized categorical labels as queries, they showed poor results as the information they provided were limited and could not fully describe the real world audio [13]. In contrast, natural language queries offer rich and detailed descriptions that help the system learn fine grained audio features, ultimately surpassing the performance of traditional audio source separation methods [14] [15].

A typical language-queried audio source separation undergoes three major steps. First, audio mixture processing and query encoding, where the audio input is transformed to a time frequency representation and a query is encoded into text embeddings. Second, the fusion of text modalities with audio features into the neural separator to predict a time frequency mask. Finally, the mask is applied on the mixture spectrogram to get the predicted spectrogram, we apply an audio reconstruction technique to recover the original waveform.

However, several challenges remain, one major issue is the subjectivity of language queries, the same sound can be interpreted differently from one individual to another, resulting in different predictions [14]. Additionally, the model requires a fusion technique to link the modalities of different nature. The choice of this technique plays a big part for good separation results.

Contribution :

Although state of the art language-queried audio source separations systems achieve impressive performance [18][88][89], they require significant computational resources and specialized hardware. We propose a computationally efficient and optimized approach adaptable to consumer grade PCs while maintaining good separation results. Our main contributions to the language-queried audio source separation task consists of :

Architectural innovation: We extend the ResUNet++ [16] backbone to the language-queried separation task by inserting cross-attention blocks that tightly fuse audio and text features.

Data preparation for efficient training: We apply cosine-similarity thresholding to remove mixture audios where the mixture and target spectrograms are overly similar, improving robustness.

Thesis organisation :

This master thesis is categorized into five chapters to establish a comprehensive foundation, propose our solution, and evaluate its efficacy for language-queried audio source separation. Chapter 1 introduces the evolution of audio source separation into language guided audio source separation, then it delves into core audio representation concepts, preprocessing techniques and feature extraction. Chapter 2 details deep learning fundamentals we relied on in our model, covering artificial neural networks, convolutional architecture (UNet, ResUNet++), transformer based natural language processors, and the cross modal fusion technique. In chapter 3, we conduct a thorough study over the previous models for the task and datasets and evaluation metrics adapted. Chapter 4 presents the proposed solution, integrating FLAN-T5 for text encoding, a cross attention enhanced ResUNet++ for query conditioned audio separation and a phase aware iSTFT for audio waveform reconstruction. Finally chapter 5 dives into the practical details, the choice of dataset for audio mixtures, the libraries and tools we leveraged and training setups as well as the evaluation results obtained.
We conclude our work with summarizing the results of our contributions, limitations and future work.

Chapter 1 : Introduction to Audio Representation and Separation

1.1. Introduction:

Audio source separation has long been a fundamental challenge in audio signal processing, evolving from purely statistical models to today's powerful deep learning architectures [17]. While state of the art neural separators have shown significant progress, they still face challenges with separating an unknown number of sound sources from a mixture [18]. Language-conditioned guidance alleviates these difficulties by fusing related text-audio pairs and enabling on-demand extraction of any specific sound source [14]. While language-conditioned models have demonstrated improved performance in audio separation tasks, it is important to walk through the audio preprocessing pipeline to understand how the audio is prepared through different steps.

In this chapter, we start by tracing the evolution of audio source separation and introduce language-queried audio separation; then we will explore the audio preprocessing pipeline starting from fundamental acoustic properties, waveform digitization methods, to preprocessing techniques, and feature extraction.

1.2. Audio source separation

Audio source separation is the process of isolating an individual sound from a mixture of sounds, with the goal of recovering the original separate sources as accurately as possible. Audio source separation covers a wide scope of applications, from music remixing, forensics analysis, to speech enhancement [19].

Earlier approaches relied on statistical methods, popular methods we include Independent Component Analysis (ICA) [2], that unmixes audio sources by maximizing the statistical independence between the estimated sources and Non-negative Matrix Factorization (NMF) [3], which works with a spectrogram and decomposes it into patterns and activations.

However, due to their minimal training [20], these models struggle when applied to real world data [21], and with the advent of deep learning, new models relying on neural networks have seen the light of the day such as Conv-TasNet [9] for speech separation, and Demucs [22] for music separation.

Although these models outperform previous traditional methods, they struggle with separation precision due to the increase of overlapping sound sources in their mixture. Some solutions attempted to solve the issue by using sound classes to guide the separation. However, the information they provided was limited [14].

1.3. Language-queried audio source separation

Language-queried audio source separation (LASS) is the task of extracting a specific audio from a given mixture based on a provided query e.g. “focus on the man’s voice” [14]. LASS aims to solve the limitations faced by traditional source separation systems, such as the difficulty in handling a vast number of sound classes in the real-world applications when multiple sources are presented at the same time, making it harder to obtain accurate separation results [23]. Another limitation is the scope of the labels or categories in traditional methods, as generally they fall within one domain, making it even harder to extract. In contrast, Language-queried audio source separation offers full freedom when it comes to the queries and can be seamlessly generalized to open domain scenarios due to the unrestricted nature of its language descriptions [15]. Moreover, it enables a more accurate separation process because the model focuses specifically on extracting the sound described in the query [18].

Language-queried audio source separation models are useful in many applications, such as automatic audio editing, multimedia content retrieval, controllable hearable devices, and audio-augmented listening [14].

For language-queried audio separation models to properly function, the relationship between the query and the relevant components of the input audio needs to be captured [3]. To achieve that, the inserted audio must be pre-processed to enhance feature extraction, allowing the model to focus on features that correspond to the query.

1.4. Audio preprocessing pipeline

1.4.1. Fundamental sound properties

Sound is the energy produced from a vibrating object. Its movements cause the surrounding particles to bump into each other, creating a wave. The overall waves generated support what is commonly known as a **sound waveform**. To distinguish different waveforms, we need to introduce some characteristics to define how they behave [24]:

Frequency is defined as the number of complete oscillations per second; it is measured in hertz. Humans perceive frequency as pitch, which is logarithmic in nature: doubling the frequency raises the perceived pitch by one octave. In audio source separation frequency is important to distinguish between overlapping sounds in mixtures [25].

Amplitude refers to the maximum displacement of a sound wave from its mean position. Humans perceive this as loudness, the larger the amplitude the higher the energy that we transfer [26].

1.4.2. Analog to digital conversion

Sound, as we perceive it, is an **analog signal** with infinite resolution on both time and amplitude. Due to storage limits, it must be converted to a digital signal via sampling and quantization:

Sampling involves taking data points from the sound wave at equidistant time intervals. The sampling rate, defined as the number of data points per second, determines the accuracy of the captured sound : the higher the sampling rate, the more of the sound resolution we're preserving, whereas too low of a sampling rate can lead to sampling errors [27].

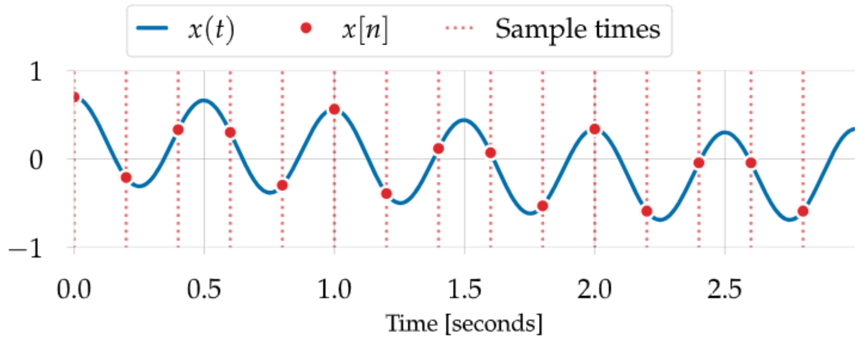


Figure 1.1 The process of sampling [28].

Quantization maps continuous amplitude values to a finite set of predetermined levels. The precision of this process is determined by the number of bits used. A higher bit depth increases the number of quantization levels which allows for capturing finer amplitude resolution; a lower bit depth can introduce quantization errors [27].

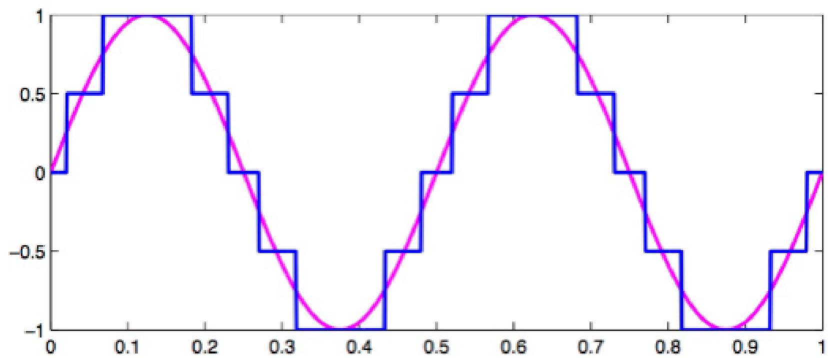


Figure 1.2 The process of quantization [29].

1.4.3. Preprocessing techniques

Before feeding the audio to the model, it is essential to pass it through a preprocessing pipeline to ensure stable training. This includes standardizing all audio clips to the same format, handling variable lengths by padding or truncating techniques to achieve a uniform duration, resampling every clip to a consistent sample rate, and normalizing the amplitude so that all signals are within the same range [30].
These steps ensure compatibility of the data characteristics and the consistency of both spectral and temporal features across all instances making it ready for feature extraction that we will explore for the remainder of this section.

1.4.4. Feature extraction

Raw audio waveforms are time-domain representations providing only temporal features However for a complex task like audio source separation which requires detailed frequency information, this representation is insufficient. This is where the Fourier transform becomes necessary:
The Fourier transform is a transition of an audio signal from its time representation into its frequency representation. Mathematically it is a model that decomposes the signal into its sinusoids (frequency) components. For each frequency, the transform multiplies the original signal by the sine wave of the frequency component, effectively computing the net area of the combined signals. This process outputs two values: a **magnitude,** which indicates how strong the sine wave appears in the signal; and a **phase** that indicates the shift of the wave within the original signal [31].

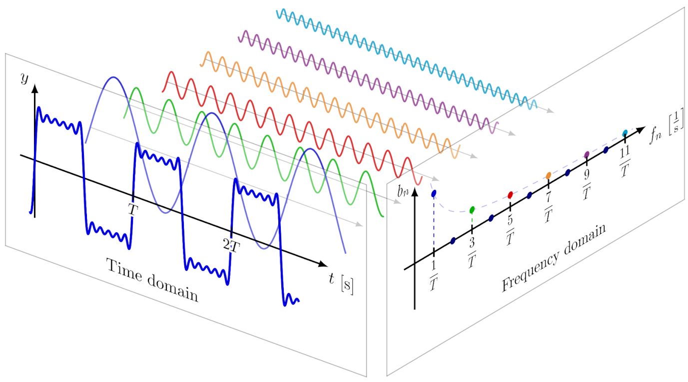


Figure 1.3 Fourier Transform process [32].

Although the Fourier transform reveals the signal's frequency components, **it does not retain when each of these frequencies occurs** which is a major limitation in real world audio. Audio data is dynamic by nature, it relies on the evolution of frequency components over time and therefore we need a representation that encompasses both temporal and frequency information.

Short-time fourier transform (STFT) addresses this problem by dividing the signal into overlapping frames, where a window function slides across these frames applying the fourier transform to each segment. Overlapping frames minimize the spectral leakage, which are distortions caused by the abrupt cut of oscillations. To avoid redundancy introduced by these fixes, a hop size parameter is added. However, the frame size involves a trade off : larger frames improve frequency resolution but blur temporal details, while smaller frames sharpen time resolution at the expense of spectral details. The STFT results in a spectrogram, a time frequency domain representation of the signal [33].

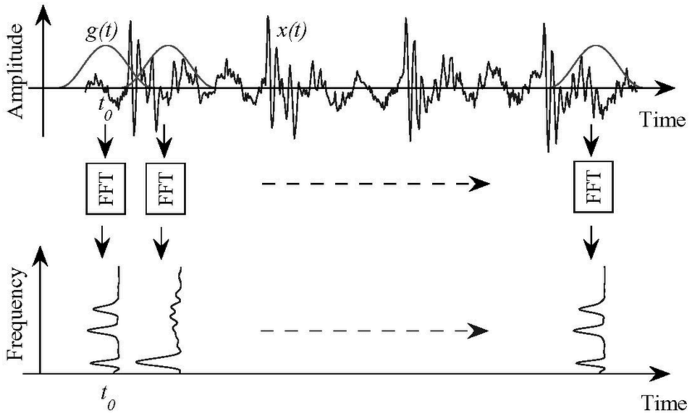


Figure 1.4 Short time fourier transform process [34].

Log spectrograms compress the dynamic range of linear spectrograms into a decibel scaled amplitude to match the human perception of the loudness. Log spectrograms preserve original frequency resolution and map extreme volumes into a perceptual range preventing the drowning out of low energy features. **Figure 1.5** below shows the impact of log magnitude:

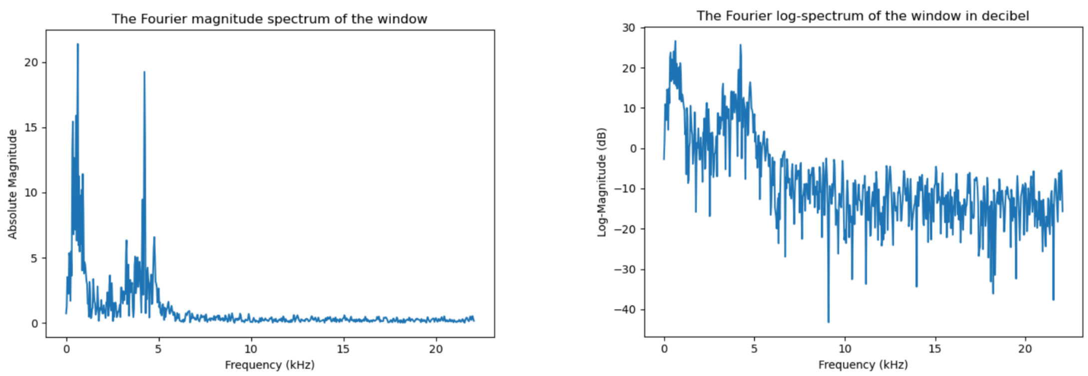


Figure 1.5 Comparison of linear and Log-Magnitude Spectra [35].

1.5. Conclusion

To summarize, Language-queried audio source separation emerged as a solution to fix the issues that traditional source separation methods faced, through the incorporation of these textual queries. To properly extract the desired sound, the audio must be properly pre-processed. In addition to that, queries need to map to the correct target audio through deep learning which is the focus of our next chapter.

Chapter 2 : Deep learning for Language-Queried Audio Source Separation

2.1. Introduction

Language-queried audio source separation is enabled through foundational deep learning techniques, which are presented in this chapter. We first present artificial neural networks and their training mechanisms, such as backpropagation and gradient descent. Next, we explore convolutional neural networks, focusing specifically on the U-Net architecture in audio source separation. We then resume with natural language processing for Language-queried audio source separation tasks, emphasizing on transformer models. Finally, we explore fusion techniques, most specifically cross attention mechanisms, which ensure that the query aligns with the corresponding audio components. These components form the foundation of our architecture, which will be described later on in chapter 4 in detail.

2.2. Artificial neural networks

A neural network consists of vertically stacked layers, which are made of multiple neurons. Within these neurons, there are adjustable parameters whose values evolve as the network learns [36].They calculate the weighted sum of their input along with the weights associated with them, then add a bias to it. Based on the result, an activation function will be applied or not, depending on a defined threshold [37].

Activation functions determine how the weighted sum of the input is transformed into an output in the neurons of a layer [38]. The most common ones are rectified linear activation (ReLU), sigmoid activation, and hyperbolic tangent (Tanh).

Artificial neural networks can contain different types of layers, which perform different transformations on their inputs [39]. Most of them are divided into three categories: the input layers, the output layers, and the hidden layers.

Artificial neural networks can be used for different tasks in multiple fields. However, when it comes to image processing, they reach their limitations; they cannot compete with convolutional neural networks, which are renowned in this field [40].

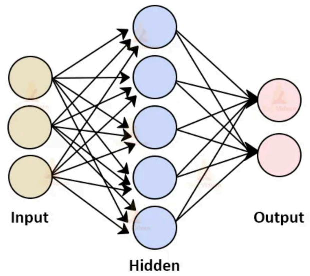


Figure 2.1. Architecture of an artificial neural network [41].

2.2.1. Training an ANN : Backpropagation, Gradient descent

Gradient descent is a popular optimization technique used to train a neural network. It minimizes the loss function by iteratively adjusting the model's parameters, guided by the negative gradient. The gradient is a partial derivative of the loss with respect to each parameter, it indicates the direction of the steepest ascent of the loss and therefore by moving against the gradient, the algorithm moves closer to the loss. The size of the descent is determined by the learning rate hyper parameter, a too small of a learning rate can slow the convergence of the model while with a much larger step we risk to skip the minimum. Gradient descent does not guarantee the convergence to the global minimum, it may settle at a local one if the loss function was non convex as shown in **Figure 2.2** below [42].

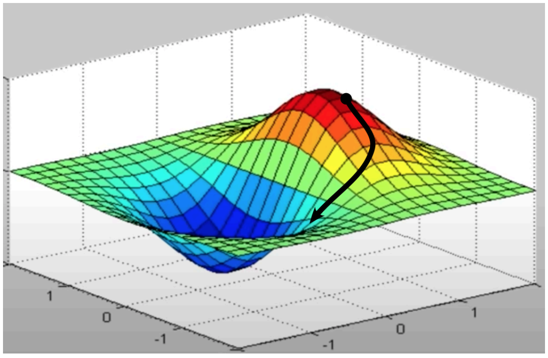


Figure 2.2 The gradient descent process [43].

Backpropagation requires gradient descent; it is an algorithm that computes errors starting at the output units and goes backward through the network layer by layer, while using the chain rule of calculus. In each layer, it calculates the gradient descent for the weights of the previous layer and repeats the process until the input layer is reached [44].

2.3. Convolutional based deep learning architectures

2.3.1. Convolutional neural network:

Convolutional neural networks (CNNs) deliver state-of-the-art performance across various fields such as medicine, gaming, agriculture [45] , and for tasks such speech and music source separation[46]. They are designed to automatically and adaptively learn spatial hierarchies of features through the backpropagation algorithm [47].The architecture consists of convolutional layers, pooling layers and fully-connected

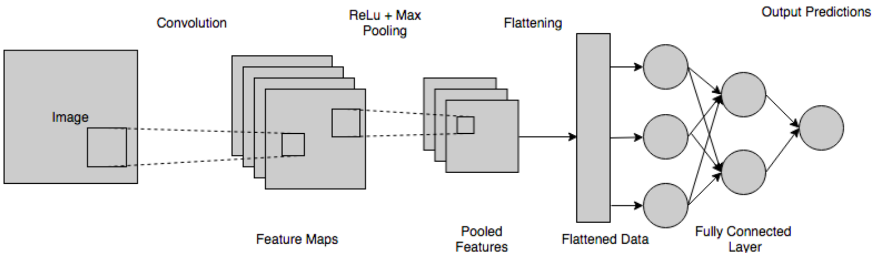


Figure 2.3 Architecture of a convolutional neural network [48].

CNN will take as input a spectrogram. This time frequency domain representation will go through an operation called convolution, where a learnable kernel, with weights that will evolve throughout training, will scan the spectrogram vertically and horizontally, computing dot products between the kernel and the input [49].

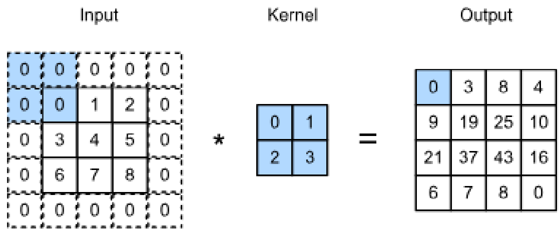


Figure 2.4 The convolution operation [50].

The stride and padding of a kernel control the overlap and the size of the output feature map [51].

The resultant feature map will pass through a pooling layer. This latter aims to reduce its size while keeping important features [52], through aggregation functions such as max pooling, where the maximum value is chosen from our feature map. This operation also has a stride parameter.

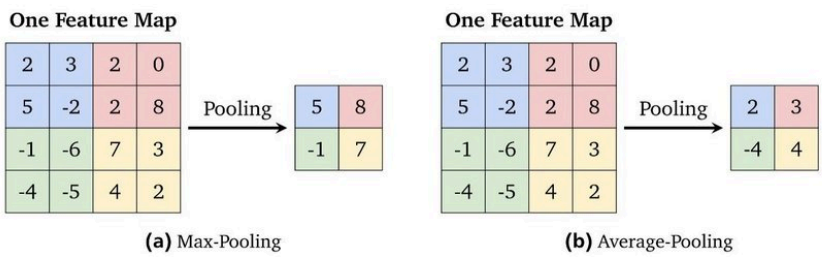


Figure 2.5 Downsampling of the feature map [53].

The flattened result is then inserted into the fully connected layers to get the final output.

2.3.2. UNet architecture for audio source separation

Built on the foundation of CNN, the UNet architecture was first proposed for biomedical image segmentation [54], and since then has known wide adaptation to audio source separation, specifically with time frequency domain features [55]. UNet as shown in **Figure 2.6**, has an encoder decoder structure, the encoder captures relevant information through several blocks into a compact representation in the bottleneck of the architecture, while the decoder reconstructs that abstract representation into the original input.

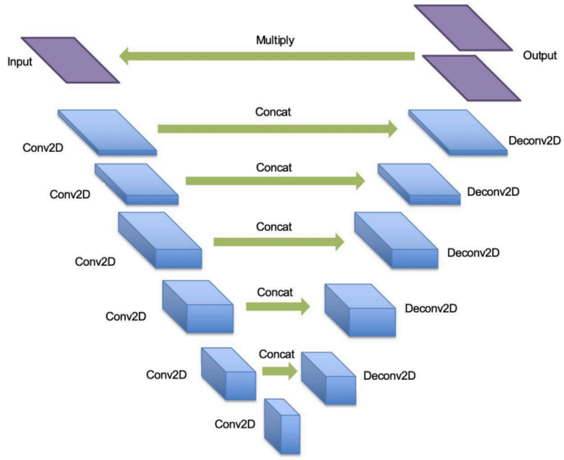


Figure 2.6 The UNet architecture for audio separation [56].

2.4. ResUNet plus plus

ResUNet++ as described in the original paper [16], is a neural network architecture that is based on a deep residual UNet. It builds upon residual blocks, attention blocks, squeeze and excitation as well as Atrous Spatial Pyramid Pooling [16].

2.4.1. Deep residual networks

Deep residual networks [57] address the problem of **vanishing gradients** introduced by very deep UNets. It learns the residuals instead of the entire desired output by adding a **skip connection** to connect a layer to further layers, facilitating the flow of gradient computations during backpropagation. **Figure 2.7** zooms inside a block of this architecture

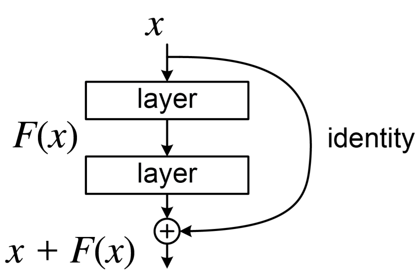


Figure 2.7 The Residual block [57].

2.4.2. UNet Attention block

Unet attention blocks are inspired by the attention mechanism in natural language processing . It guides the network to focus on task relevant regions, integrated into each skip connection. It takes the encoder feature map and the decoder gating signal, combines them to compute a spatial mask that is applied on the encoder features, letting only the lost important time frequency pixels to pass through to the decoder [58].

2.4.3. Squeeze and Excitation block

The Squeeze and Excitation Block (SE) is a channel wise attention mechanism introduced in 2018 [59], to improve the performance of CNN and focus on channel interdependencies. It consists of two steps. First, the squeeze phase where each channel is collapsed into a scalar value by performing average mean pooling over its spatial dimensions, obtaining a vector of squeezed channels, this step helps capture global context. Second, the excitation phase, which takes the vector of channels as input and passes it to a small two-layer network. The output is a list of channel importance weights which we apply on the original channels to highlight the most relevant features [60]. The process is illustrated detail in **Figure 2.8**:

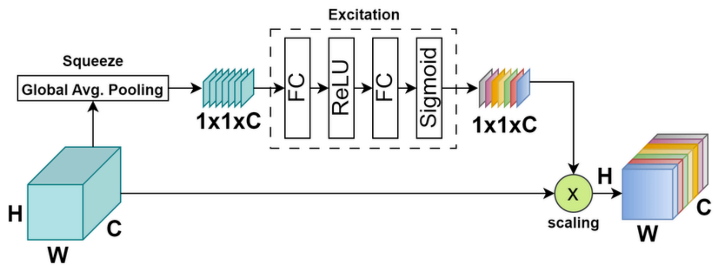


Figure 2.8 The architecture of SE block [61].

2.4.4. Atrous spatial pyramidal pooling

Atrous Spatial Pyramidal Pooling (ASPP) is used to extract features on different scales [62], it mainly relies on atrous convolutions and employs them in parallel, with different dilation rates on our input [63]. Atrous convolutions are a variant of standard convolution that introduce dilation rates, which expand the receptive field of the kernel, by inserting holes in it, through the addition of gaps between the kernel values. This allows it to cover a large area of the input without increasing the number of parameters and computational capacity. They have been used in multiple fields such as audio processing and semantic segmentation [64].

2.5. Natural language processing in language-queried audio source separation

Natural language processing enables computers to understand, process and interact with human language. Some applications of NLP include text translation, voice recognition, text summarization, and chatbots [65].

In the context of language-queried audio source separation, NLP models encode our textual queries into textual embeddings, which are later used to guide the extraction process in our separation network [66]. Embeddings represent our text in a numerical format, in which words with similar meaning have similar representations [67].

Transformers in NLP :

Transformers are state-of-the-art models that redefined the standard for NLP models, due to their ability to pay attention to specific words, and their superior performance compared to traditional models [68]. Well-known text transformer models include FLAN-T5 [69], BERT [70], and GPT [71].

The core components of transformers are:

Positional encodings : define the word order in a token embedding sequence, helping the model in recognizing the structure of the sequence [72].

Multi-head attention mechanisms : enables focusing on different aspects of a sequence in parallel. It allows the model to determine the relationships present in the sequence along with weighting the importance of different tokens that are present [73].

2.6. Fusion techniques for language audio alignment

After building two powerful architectures, one extracting high level audio feature maps and the other producing dense text embeddings from the user query in the previous sections, the last piece of the puzzle to establish a functional language-guided audio separation pipeline is a reliable fusion technique that captures the complex relationship between the two modalities.

We use cross attention as a fusion technique to dynamically align language queries with their corresponding audio components. It follows the concept of self attention with a few changes; it considers the text encoder input as a query and the UNet's feature maps as keys and values. As shown in **Figure 2.9**, cross attention computes similarity scores between each query and key, the resultant values are assigned weights depending on their level of contribution into a feature vector that effectively acts as a conditioning signal to the target [74].

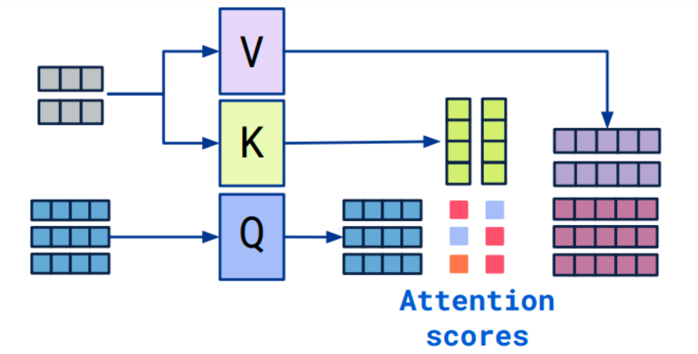


Figure 2.9 Cross attention mechanism [75].

2.7. Conclusion

To conclude, chapter 2 has built the foundational blocks of our language-queried audio source separation pipeline. We began by reviewing artificial neural networks and their training via backpropagation and gradient descent, then moved on to convolutional architectures. Next, we covered transformer based embeddings. Finally, we talked about cross attention as a fusion technique to merge the audio features with text embeddings and guide the separation.

In the next chapter, we will survey how these different architectures and techniques are assembled in previous language-queried audio separation models.

Chapter 3 : State of the art benchmarks for Language-Queried Audio Source Separation

3.1. Introduction

Since its introduction in 2022 [14], language-queried audio source separation has attracted the attention of many researchers. Since then, many models appeared in favor of this task [15][18][88][89], leveraging multiple techniques and architectures aiming to boost performance and increase separation quality.

In this chapter, we start by establishing the evaluation framework, identifying the key datasets and evaluation metrics that define how to measure the progress. Then, we list the most popular approaches that proved state of the art performance. Finally, we analyse the different models by breaking them down into their core components, strengths and limitations while establishing common ground between them in order to select and pick the components to use for our model.

3.2. Language-queried audio separation benchmarks

The rapid progress in language-queried audio source separation relies on two key pillars : large-scale audio datasets with rich language descriptions and robust evaluation metrics that measure the separation quality. This section outlines the most widely adopted datasets and metrics in the field.

3.2.1. Datasets

Table 3.1 lists the benchmark datasets in language-queried audio separation and highlights their most relevant aspects to the field, showing the progression from classification only datasets to to language separation specific datasets:

Table 3.1 Overview of audio datasets [76].

	Year	Total hours	Audio source	Annotatio n type	Audio duration	Separation advantages	Separation limitations	refer ence
AudioSet	2017	4971	Youtube videos.	Class Labels.	10 sec.	Diversity in sound events.	Weak annotations (labels).	[77]

	Year	Total hours	Audio source	Annotation type	Audio duration	Separation advantages	Separation limitations	reference
VGGSound	2020	550	Youtube videos.	Class labels.	10 sec.	Audio visual corresponda nce.	Lacks language annotations.	[78]
AudioCaps	2019	127	AudioSet subset.	Human captions.	10 sec.	The use of human annotation captions	Simple caption.	[79]
WavCaps	2023	5500	Multiples sources.	Mixed (human + AI generated) .	Variable length.	Diverse audio categories. and rich captions.	The quality of captions vary. (different datasets).	[80]
Clotho v2	2019	36	Freesound	Human captions	15-30 seconds	Rich human descriptions for audio clips.	Small training data size.	[81]
Auto-ACD	2023	1700	Multiple sources	AI generated	Variable length	Large scale dataset.	Captions may lack the nuance of human annotations.	[82]
DCASE-2024-T9Dev	2024	21	Synthetic and real mixtures	Task specific (source separation)	10 sec	Specifically designed for the task.	Limited sound classes	[83]

3.2.2. Evaluation metrics

Evaluation metrics are crucial in determining the performance and effectiveness of a model [84]. In language-queried audio source separation, the performance of models is evaluated through:

SDR measures of the quality of a signal, through comparing it with its distorted or noisy version , higher SDR indicates a superior quality of separation [85] . It is calculated though this formula :

$$\mathrm{SDR} = 10 \times \log_{10}\left(\frac{||\mathbf{s}||^2}{||\mathbf{s} - \hat{\mathbf{s}}||^2}\right)$$

where : s is the ground truth audio.

$\hat{s}$ is the estimated one.

SDRi is an improvement metric that measures the difference in SDR before and after applying an audio source separation algorithm [86].

$$\mathrm{SDR_i} = \mathrm{SDR_{after}} - \mathrm{SDR_{before}}$$

CLAPScoreA: measures the similarity between the output audio and the target audio using CLAP, Contrastive Language-Audio Pretraining, CLAP learns to connect natural language and audio via two separate encoders and a contrastive learning objective that brings audio and text descriptions into a joint multimodel space [87]. CLAPScoreA is an innovative metric adapted to evaluate recent language queried audio source separation models like FlowSep [88], whereas the higher the value, the more accurate the separated sound is to the ground truth.

3.3. Previous works in language-queried audio source separation

Table 3.2 provides the previous works in language queried audio source separation. It includes each model's foundational component, the datasets used for training, the text encoder applied, the separation network integrated and the fusion technique used to align the different modalities. The models were evaluated on a set of evaluation metrics some different from others depending on the nature of the model. The main purpose of this table is to highlight the evolution of the models for the LASS problem in the recent years:

Table 3.2 A comparison of language-queried audio source separation models.

Model	Dataset (training)	Text encoder	Separation model	Fusion technique	Audio reconstruction	Evaluation metrics	Results
LASSNet	Dataset created from AudioCaps	BERT	ResUNet	FiLM	iSTFT	SDR SDRi (Clotho)	6.35 2.36
AudioSep	AudioSet, VGGSound AudioCaps, Clotho v2, WavCaps,	CLAP	ResUNet	FILM	iSTFT	SDRi (Clotho)	6.85

Model	Dataset (training)	Text encoder	Separation model	Fusion technique	Audio reconstruction	Evaluation metrics	Results
AudioSep DP	DCASE-2024 - T9Dev, Audiocaps, Auto-ACD, Wavcaps	CLAP text encoder	ResUNet+ DPRNN	FILM	iSTFT	SDR SDRi	8.764 (32 khz sr) 8.658 (32 khz sr)
CLAPSep	AudioCaps	CLAP text encoder	CLAP audio encoder	FiLM + feature aggregator	iSTFT	SDRi	20.17±12.43
FlowSep	AudioCaps, VGGSound WavCaps	FLAN-T5	Rectified Flow Matching	Through Rectified Flow Matching	Vocoder BigVGan	CLAPScoreA	78.9

3.4. Comparative analysis

LASSNet [14], was the first model to ever integrate natural language queries for audio source separation. It used BERT to encode the query and a ResUNet separation network merged via FiLM layers. LASSNet achieved an SDR of 6.35 and demonstrated a good adaptability to real world scenarios. However, this model being the first attempt to solve the LASS task, it faced many limitations. It lacked training and did not perform well on open-domain scenarios. Additionally, its predicted mask was either excessively or insufficiently selective, leading to artifacts such as spectral holes or incomplete separation [88].

AudioSep [15], was introduced to overcome the shortcomings of LASSNet. It replaced BERT with CLAP, a contrastive audio text model that embeds both modalities in a shared latent space; this change strengthened semantic alignment between audio-text pairs and allowed the decoupling of query and separation training. Additionally, AudioSep was trained on a variety of sound sources achieving remarkable zero shot performance. Unlike LASSNet, it tackles the phase reconstruction challenge by learning phase residuals, ensuring a proper reconstruction of the target audio. Nonetheless, it still faces issues with the predicted mask [88].

AudioSep DP [89], was introduced to address the limitations of AudioSep, through inserting a dual path RNN block between the encoder and decoder stacks of the ResUnet, which allowed the model to independently capture both temporal and spectral information. Furthermore, target sound event detection branches are used to identify the events present in the predicted sound. These improvements led to a higher SDRi of 8.658, surpassing AudioSep.

CLAPSep [18], compared to previous works, leverages the CLAP model for both query encoding and separation. The separation network consists of a feature aggregator which follows the UNet architecture. It is responsible for producing an aggregated feature rich in information about the target sound conditioned on the text embedding. This aggregated feature, along with the linear magnitude of the mixture, is then passed to a MaskNet, producing a spectrogram mask that will later be concatenated with the spectrogram of the mixture to get the target spectrogram. CLAPSep uses low-rank adaptation lightweight which is a fine tuning technique that allows it to adapt with minimal training. CLAPSep outperforms both LASSNet and AudioSep by establishing a semantic relationship between the query and separation networks.

FlowSep [88], is an innovative approach that discarded discriminative architectures and relied on generative models. It adopts rectified flow matching RFM to gradually transform a gaussian distribution into the target audio representation. Unlike previous models that are mask based, FlowSep samples from a learned distribution. Additionally, it introduces gaussian corrupted data in training to adapt the model to real world data, which is often recorded in noisy environments. For text encoding, FlowSep leverages FLAN-T5, a model fine-tuned on a variety of tasks enhancing its adaptability to our task and outperforming CLAP. On top of that, the model trains a vocoder for a high fidelity audio reconstruction. This model has achieved the best separation quality.

3.5. Conclusion

In this chapter, we presented commonly used datasets for training, along with evaluation metrics used to assess model performance. We then conducted a comparative study on various state of the art models based on different key components such as the text encoder, evaluation metrics and the architecture. In the next chapter, we will be presenting the components of our proposed model.

Chapter 4 : Proposed model architecture

4.1. Introduction

Over the past few years, language-queried audio source separation has gained in popularity. Many models have seen the light of the day with the majority of them focusing on discriminative approaches for their straightforward architecture and suboptimal results [14][15][18]. Inspired by the efforts to adapt ResUNet [16] for language-conditioned separation, we instead adopt a more efficient mask-based approach using ResUNet++ which was yet to be used for language-queried audio source separation tasks; we found potential in its simplified [16] and extendable design [90] to perform our task.

In this chapter, we will explain the role of each component in our model from the FLAN-T5 text encoding capabilities to the separating ResUNet++ architecture and finally a phase aware inverse STFT for audio reconstruction.

4.2. Overall system architecture

Our system is made of several key components that operate together in order to make the process of language queried audio source separation happen. From a high level perspective, it includes an audio pre-processing pipeline, a text encoder, a query conditioned separation model and a waveform reconstruction module. Together these cited components ensure that the extracted sound matches the given query.

First, we preprocess our audio and text: we make mixtures out of our audio, and turn the mixture audio and the target audio into their respective time frequency domain representation and we encode the text query into a text embedding using FLAN-T5 text encoder. Second, our cross attention infused ResUNet++ takes as input the mixture spectrogram with its the query embedding, layer by layer, the model learns to predict a mask from the given mixture conditioned by the query embedding. The predicted mask is concatenated to the inserted mixture spectrogram to obtain the predicted spectrogram. Finally, the isolated spectrogram is passed through inverse STFT with mixture phase to obtain the corresponding raw waveform. **Figure 4.1** illustrates the approach we adopted in more details

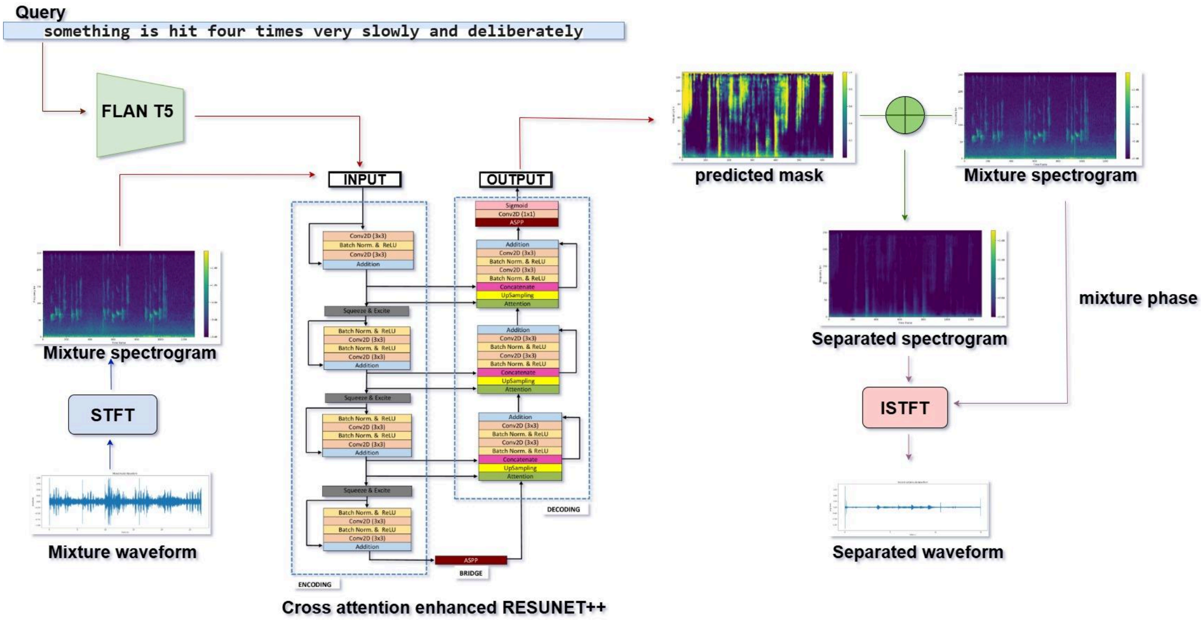


Figure 4.1 An overall view of our model's architecture with the different components.

4.2.1. FLAN-T5 component

FLAN-T5 represents an enhanced version of T5. It has been fine tuned on more than a thousand diverse tasks, enabling it to generalize well on unseen data and generate high-quality, instruction aware text embeddings. It also achieves strong semantic understanding and captures nuanced user intent. FLAN-T5 outperforms alternative text encoders such as CLAP [88] and BERT which have been previously used for this task [91]. In our approach, we utilize only the encoder part of the model :

We tokenize the input using SentencePiece to create a vocabulary of subword units by splitting input words into subword tokens, convert these units into token IDs, which are later transformed into token embeddings that contain the positional encoding of each token in the sequence. These resultant embeddings are finally inserted into the encoder, where they go through self attention mechanisms, feed forward neural networks, layer normalization and residual connections to get contextualized embeddings [92] as shown in **Figure 4.2** below.

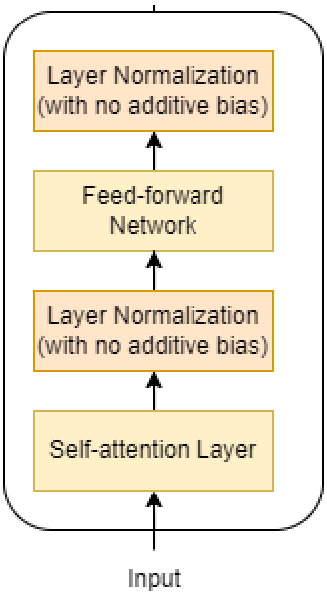


Figure 4.2. Architecture of the encoder stack of FLAN-T5 model [93].

4.2.2. Separation component

Our source separation network builds upon ResUNet++, incorporating additional cross-attention mechanisms to guide the separation process using language queries. ResUNet++ proved state of the art performance in medical imaging tasks, outperforming both UNet [54] and ResUNet [16]. Motivated by its performance, we chose this architecture to investigate its effectiveness in language-queried audio source separation tasks. Its lightweight and efficient design displayed in **Figure 4.3** further encouraged us to adopt it.

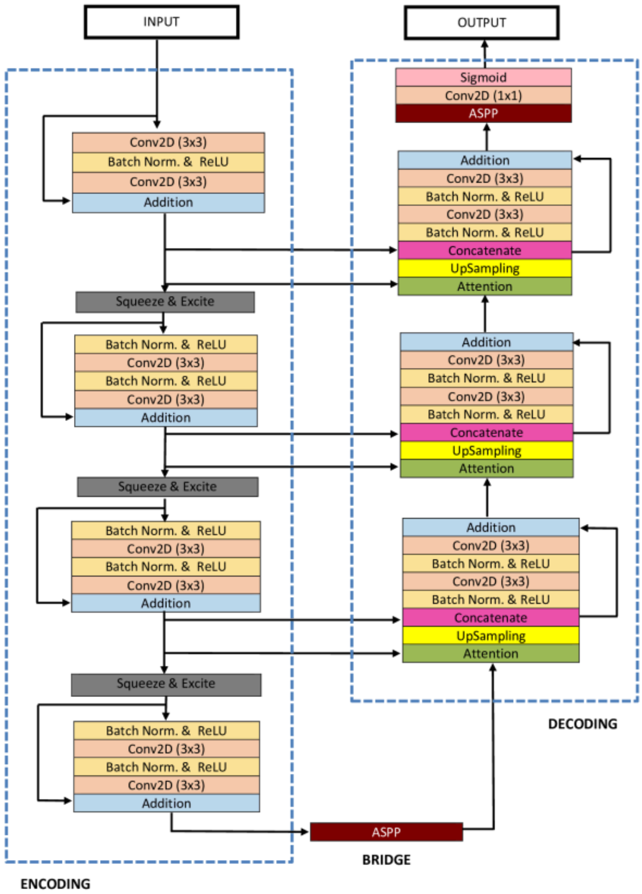


Figure 4.3. Architecture of the ResUNet++ [16].

The architecture follows an encoder-decoder fashion, where the input mixture spectrogram is first passed to a stem block, which consists of a preliminary step for the encoder that only allows relevant features to continue. The encoder consists of 3 residual blocks each followed by a cross attention layer. Each residual unit downsamples the channels and filters them out using an SE block. The cross attention layer is applied immediately after to fuse the text embedding into the feature map to prioritize the audio features associated with the query early on. After passing all the layers in the encoder, the output is **a compressed abstract representation.**

In the bottleneck, this latter is passed to an ASPP module which expands the receptive field using multiple atrous convolutions at different rates enabling the model to capture fine, medium, and long-range temporal structures simultaneously, a cross attention layer follows to refine the multiscale representation to tend to query embeddings details.

In the decoder, the attention block applies a gating mechanism in skip connection to allow only relevant high resolution details from the encoder to the decoder upsampled feature, these combined features are again processed through a cross-attention layer and an SE block to ensure that only the most query-relevant and high-resolution features influence the final prediction. A final cross-attention layer and lightweight ASPP further refine the output before it is passed through a sigmoid-activated 1×1 convolution to generate the time-frequency mask for source reconstruction.

By concatenating the mask to our input mixture, we get our **final separated spectrogram**, which is finally tuned into waveform using the audio reconstruction method explained in the next section.

4.2.3. Phase aware inverse short time fourier transform

We use a phase aware iSTFT for audio reconstruction. This method [94] builds on the standard inverse short time fourier transform by reusing the mixtures's phase [95] instead of estimating it. Given the predicted magnitude spectrogram and the mixture phase, we first reconstruct the complex windowed frame using **inverse fourier transform** to their corresponding sinusoids, we shift them by their hop size and overlap and add them respecting the COLA condition [96]. This approach avoids the computational cost and convergence issues of iterative phase estimators such as Griffin-Lim. Integrating phase with iSTFT has shown a stronger performance compared to the standard iterative phase estimator, as observed both in literature [95] and in our own experiments. This is due to the fact that phase is significantly more difficult to model than magnitude due to its stochastic nature [97].

4.3. Conclusion

Throughout this chapter, we provided an overview and explained in detail the role of each component of our language-queried audio source separation model. In the next chapter, we will present our experimentations and discuss the results we obtained.

Chapter 5 : Experimentation & Results

5.1. Introduction

This chapter presents the end results of all the trial and error that went into building our language-conditioned audio source separation. First we will walk through the open source libraries and tools we relied on as well as the online environments that we used for implementation and training. Then, we will cover how we picked and processed our audio text dataset. After that we will enumerate our training set up and hyperparameter tuning. Next, we will share the numerical results of the training as well as a demonstration application to display listening examples. We conclude the chapter with the challenges and limitations we encountered along our journey.

5.2. Libraries and tools

To build our language-queried audio source separation pipeline, we combine state of the art deep learning frameworks, specialised text and audio processing packages and other utilities on scalable cloud platforms. In the section below we will delve into each category.

5.2.1. Libraries

Core deep learning framework :
We build our model on Pytorch [98], an open source python library, derived from the original Torch framework. PyTorch offers high-level abstractions for tensor operations, automatic differentiation, and seamless GPU acceleration, making it ideal for developing and training complex neural networks like language-queried audio source separation.

NLP embeddings and tokenization :
For converting user queries into rich vector representations, we leverage Hugging Face Transformers [99]. Its suite of pretrained tokenizers and models, including the FLAN-T5 text encoder, allows us to efficiently tokenize text and generate embeddings that guide the audio-separation process.

Audio processing :
Librosa [100] provides fundamental audio-analysis routines from loading files, computing STFTs, Mel spectrograms, and other spectral features, enabling fine-grained feature extraction.

Pydub [101] offers high-level audio manipulation, which we use to synthesize mixtures before passing them to our spectrogram extractor.

For Data manipulation and Visualisation: we use NumPy [102] to perform numerical operations on spectrograms arrays, Pandas [103] for dataset management and preprocessing in DataFrame structures, and Sklearn [104] to split data for cross validation. Matplotlib [105] is used to plot spectrogram representation for comparisons.

5.2.2. Tools

Deep learning models are computationally heavy, thus training them with local hardware is almost impossible, as it would lead more often than not to the instance crashing, or the model needing an extremely long time to train, which is not ideal. Moreover, audio data requires a lot of storage.

To address these issues, we mainly relied on the tools presented below to store our data and train our model:

Google drive: is an online storage service that enables synchronization and sharing [106]. Different plans of varying storage capacities are proposed, in which the 15 GB option is free. For our project, to store all the mixtures we created, we chose the 2TB offer.

Colab: is a cloud-based platform that allows users to write and execute python. By providing this virtual environment it facilitates machine learning tasks [107]. Colab offers different subscription plans, each providing a certain amount of computing units and features such as the interruptibility of the kernel and background executions. Along with each subscription type, different GPUs are offered such as T4, L4 or A100. To train our model, we chose the google colab pro+ subscription to minimize training interruption and gain access to faster GPUs, most specifically A100.

Runpod: along with Google Colab, we also used Runpod, a cloud solution made for the development and training of AI models [108]. It offers a wide range of GPUs, from H100 to RTX 3090, whereas they can be rented by the hour. We primarily used RTX 6000 ADA with a storage disk space of 400GB so it can withhold all our data.

5.3. Dataset Overview and Preprocessing Pipeline

5.3.1. Dataset overview

For the dataset, we chose Clotho [81] due to its convenient size, making it easy to use across platforms. In addition, it offers detailed captions in which each sound is associated with five unique captions each containing eight to twenty words. The dataset holds over 4981 audio clips in total, whereas the length of the audio samples ranges from 15 to 31 seconds.
For training purposes, we choose to keep only one caption per audio and that is to ensure that our model trains on all available samples. Following this, we preprocess the audio in order to use it.

5.3.2. Mixture making

We use Pydub AudioSegment to make audio mixtures. It reads each audio file and automatically decodes it. After, to ensure that neither source dominates the other, because of differing recording levels, we normalize both audio clips to the same loudness to create a fair mixing scenario where separation is not biased to one clip over the other.
Before mixing, we check the durations of both clips, if one is shorter, we append silence to the shorter clip to match the duration of the other one. Finally, we align both sounds so their start times coincide. At each millisecond, Pydub adds the sounds' samples together resulting in an overlapping sound with two sources [109].

5.3.3. Preprocessing Pipeline

Our pipeline consists of loading, padding, generating the spectrogram, and normalizing it.

The loader processes the audio with a fixed length of 15 seconds. In cases where the audio is too short padding is applied. Padding consists of adding silence in the beginning or the end of the audio [110], and is mostly used to ensure consistent audio lengths.

Next, we extract log-spectrograms using Short Time Fourier Transform with a frame size of 512 and hop length of 256. The resulting spectrograms are then normalized. Normalization is used to scale the decibel values of our spectrogram to a fixed range going from [0,1], it ensures training stability and prevents vanishing/exploding gradients [111].

5.3.4. Cosine similarity filtering for mixtures

After training our separation network for multiple epochs, we observed that the model consistently predicted the mixture spectrogram instead of separating the target source. Upon closer inspection, we discovered that the spectrograms of many of the mixtures and their corresponding target were very similar. These pairs constituted the majority of our training data, indicating the sounds of the mixture shared similar spectral content.

We hypothesised that training our model with a dataset dominant of these high similar pairs was the reason it returned the mixture as output. Lazily learning the identity mapping from it rather than distinguishing and separating sources.

To address this issue, we applied cosine similarity filter [112] to each mixture-target pair. We obtain its value by computing the dot product between the spectrograms of the pair divided by the product of their magnitude; we get a value between 0 and 1 reflecting how aligned the vectors are. We discard pairs above varied thresholds (0.8, 0.75, 0.7 and 0.65) to maintain a balanced mix of highly overlapping and more distinct training examples.

After retraining with this filtered dataset, we observed notable improvements: throughout training, as we periodically saved predicted, target and mixture spectrograms, we could see the model beginning to learn characteristic patterns of the target source rather than defaulting to the mixture.

Of course, we want to point out that this remains our working hypothesis, one that may not generalise universally and other factors such as architecture choices, learning rate schedules and loss functions likely contribute to the improved performance. Furthermore, while filtering proved useful during initial training to encourage the model to learn distinct patterns, for later fine-tuning, we plan to reintroduce the unfiltered data so the model can capture more nuanced source characteristics.

5.4. Implementation details

5.4.1. Data generation

To ensure a high diversity in our training data, despite the time cost, we assembled 17000 Clotho mixtures via an iterative cosine similarity filtering process. First we randomly generated oversized batches of mixtures, computed their magnitude spectrograms. Next, for each mixture batch we discarded samples with a high target mixture cosine similarity. We repeated the process “generate → extract → filter” until we obtained a decent number of training samples.

5.4.2. Training

The filtered mixtures are then passed onto the training pipeline, they are trained for 200 epochs with a batch size of 16 using **Adam optimizer** and **Cosineannealinglr** and a set of **Losses**.

Adam optimizer or commonly known as ADAM, is an optimization algorithm designed to improve training speed and reach convergence quickly. It relies on two other robust optimization techniques, momentum and RMS prop, in which both methods are used to adjust the learning rates for each parameter during training [113]. We use ADAM with an initial learning rate of 1e-4.

CosineAnnealingLR (Tmax, eta_min) is a learning rate scheduler used for a variety of models, such as Restormer for image restoration and ResNet++ in image classification, which is ideal for our case [114]. It helps models escape local minima and often achieves better final performance than linear decay schedules [115].This scheduler relies on two parameters Tmax which is the number of max iterations , eta_min the minimum learning rate achievable. It follows a cosine curve whereas the learning rate starts at a high value then is periodically adjusted over a number of specified epochs [116].
For our training setup, we set the minimal learning rate at a standard value of 1e-6 and we set Tmax at 200 which represents the duration of our training.

Losses: To train our model to isolate a target source given a text query, we combined two complementary losses, each term addressing a different aspect of the separation task.

<u>Spectral convergence loss (SC)</u>: is used to measure how closely the predicted spectrogram matches the target in shape. SC operates especially well in early training stages [117] guiding the model to its correct spectral shape first.

$$\mathcal{L}_{SC}(Y, \hat{Y}) = \frac{\| \, |Y| - |\hat{Y}| \, \|_F}{\| \, |Y| \, \|_F}$$

Concretely this loss computes the difference between the linear magnitude of the target and predicted in the frobenius norm and then divides it by the frobenius norm of the target to make it relative to the latter.

Log scale STFT magnitude loss: unlike SC, this loss captures small amplitude components [117] that are more perceptually relevant to humans. By computing the mean absolute difference between the logarithms of the predicted and the target.

$$\mathcal{L}_{\text{LogMag}}(Y, \hat{Y}) = \frac{1}{N} \sum_{i,j} \left| \log\left(|Y_{i,j}| + \epsilon\right) - \log\left(|\hat{Y}_{i,j}| + \epsilon\right) \right|$$

The two losses combined help capture the overall energy distribution and fine grained details.

5.4.3. Evaluation set

We reserved 2500 mixtures to form our test set, these clips underwent the identical spectrogram extraction and cosine similarity based filtering procedure to ensure fair evaluation.

5.4.4. Reconstruction

For each predicted magnitude spectrogram, we first denormalized it using the original clip's stored min/max values. We then reconstructed the audio waveform via inverse STFT reusing the phase information from the original mixture.

5.5. Results

5.5.1. Quantitative results

We evaluate separation performance using three complementary metrics: SDR , SDRi and CLAPScore A.

Table 5.1 Objective evaluation results.

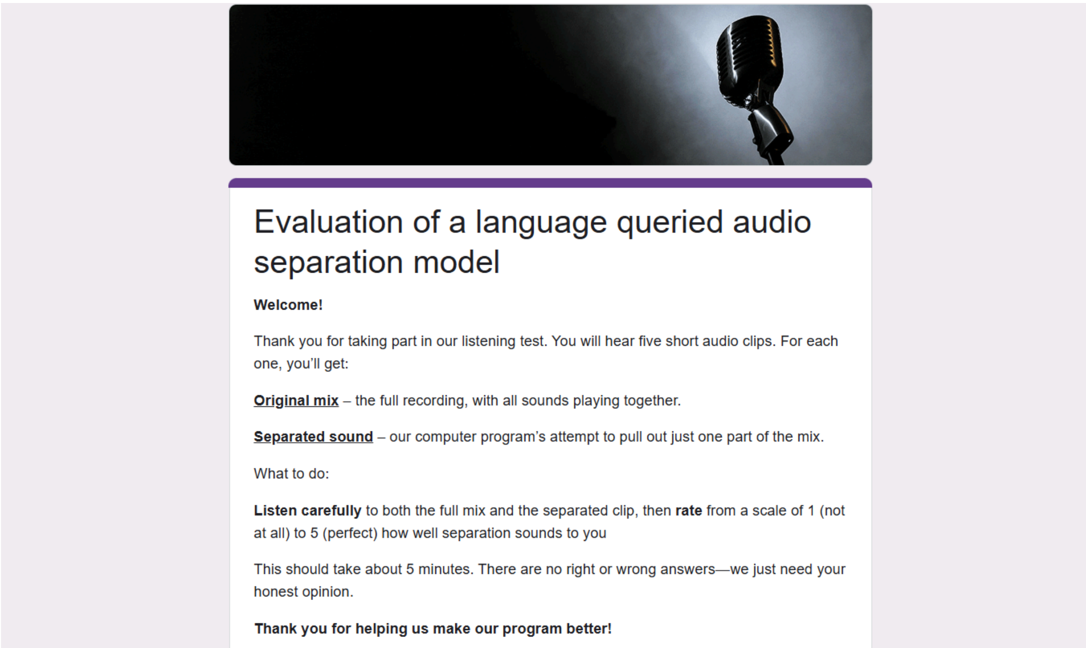
dataset	SDR Mean ± Std	SDRi Mean ± Std	CLAPScore A Mean ± Std
Clotho	2.41 ± 2.55	8.37± 5.24	40 ± 11

Table 5.1 shows that our model achieves a respectable mean SDR of 2.41 dB with a standard deviation of 2.55 dB which indicates that while many clips are separated cleanly some samples remain challenging. In terms of SDRi, we observe an average improvement of 8.37 dB over the mixtures. However with a large spread (std 5.24 dB), this reflects the diversity of Clotho dataset from very simple isolated sounds (e.g. bird calls) to complex,

overlapping urban recordings. We achieved a mean CLAPScore A of 40 (± 11) demonstrating that our model consistently captures a decent amount of semantic information in the separated audio.

5.5.2. Qualitative results

We conducted a blind listening test with over 70 participants from different fields, asking them to rate five mixture-separation pairs on a scale of 1 to 5, with 1 being "unsuccessful separation” and 5 being “perfect separation”. **Figure 5.1** below provides further details on the form:



Evaluation of a language queried audio separation model

Welcome!

Thank you for taking part in our listening test. You will hear five short audio clips. For each one, you'll get:

Original mix – the full recording, with all sounds playing together.

Separated sound – our computer program's attempt to pull out just one part of the mix.

What to do:

Listen carefully to both the full mix and the separated clip, then **rate** from a scale of 1 (not at all) to 5 (perfect) how well separation sounds to you

This should take about 5 minutes. There are no right or wrong answers—we just need your honest opinion.

Thank you for helping us make our program better!

Figure 5.1 The evaluation form.

Description of our samples and the queries we proposed for each one:

Sample 1: contains ambient forest sounds with an owl hooting in the background. We associated this sample with the query “resonant hoot pierces the night’s steady chorus of crickets”.

Sample 2: contains ambient party sounds with multiple people talking in the background when someone suddenly knocks on the door. We associated this sample with the query “remove any other sound except knocking on the door”.

Sample 3: contains street ambient noise with the sound of someone skating, along with sirens blaring in the background. We associated this sample with the query “a siren blaring and it continues over and over again”.

Sample 4: contains ambient sounds of a forest accompanied by faint bird calls and the sound of someone walking. We associated this sample with the query "the birds sing louder and louder in nature".

Sample 5: contains multiple whales making sounds under the rain. We associated this sample with the query "multiple whales making sounds".

We note the average and top scores for each sample into **Table 5.2** below:

Table 5.2 Subjective evaluation results

Sample	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5
Average rating	3,75	4,01	3,60	4,09	3,77
Highest rating	4	4	3,5	5	4

Sample 1: with an average rating of 3.75, this sample had a highest rating of 4 with 36.8% votes, 29.4% for 3 and 25% for 5 proving a beyond average separation was achieved.

Sample 2: scoring an average of 4.01, this example had 36.8% of listeners rate it a 4 and 35.3% rate it a 5. Those high marks in this example confirm the network's ability to cleanly extract a clear target signal from noisy party background mixture.

Sample 3: with an average of 3.60, participants' votes were more dispersed: 29.2% chose 3, and another 29.2% chose 5, 22.2% chose 4, and 15.3% chose 2. This spread reflects the difficulty posed by the sample's overlapping sources where some residual artifacts remain unavoidable.

Sample 4: achieving an average rating of 4.09, 42.9% of raters awarded it a five and 32% gave it a 4. Despite the target being faint in this example, the model isolated it almost perfectly, as proved by the results.

Sample 5: averaging 3.77, this sample received 43.1% 4-ratings and 23.6% 5-ratings while occasional remnants of the background persist even so, the outcome remains well above average.

To summarize, for the most part and considering all samples the separation was achieved beyond average, however some interferences still remain from the mixture.

5.6. Application Demo

Along with the abstract numbers, we present a Flask powered front end to demonstrate our model performance in real time. This web interface allows the user to upload a mixed audio file along with a description of the sound to isolate from a list of samples as shown in **Figure 5.2**

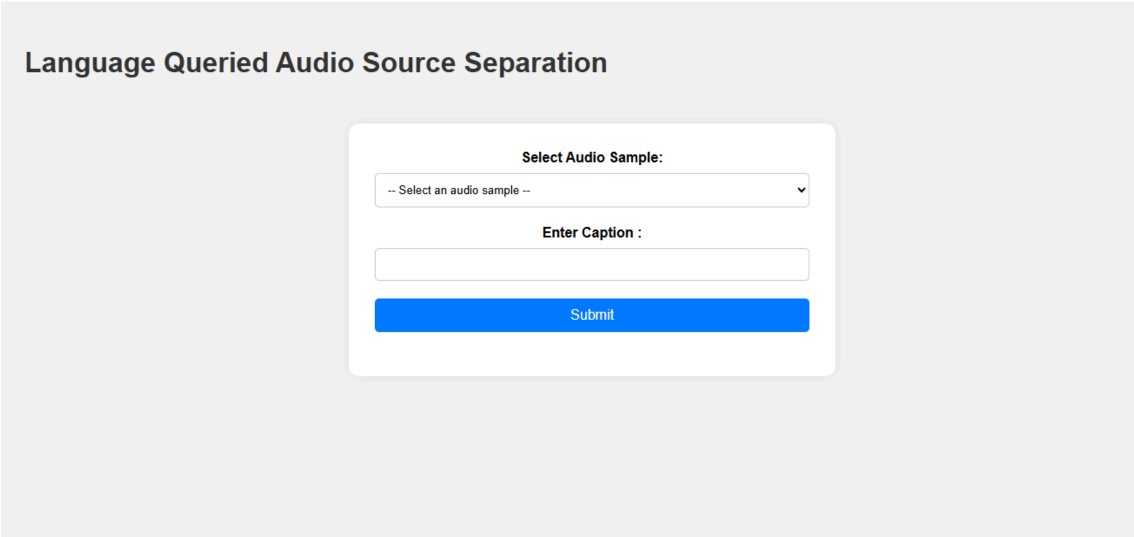


Figure 5.2 Main interface for separation.

After submitting a mixture-caption pair, the app returns the isolated audio along with the original mixture audio to allow the comparison of the two files side by side, illustrated in **Figure 5.3**

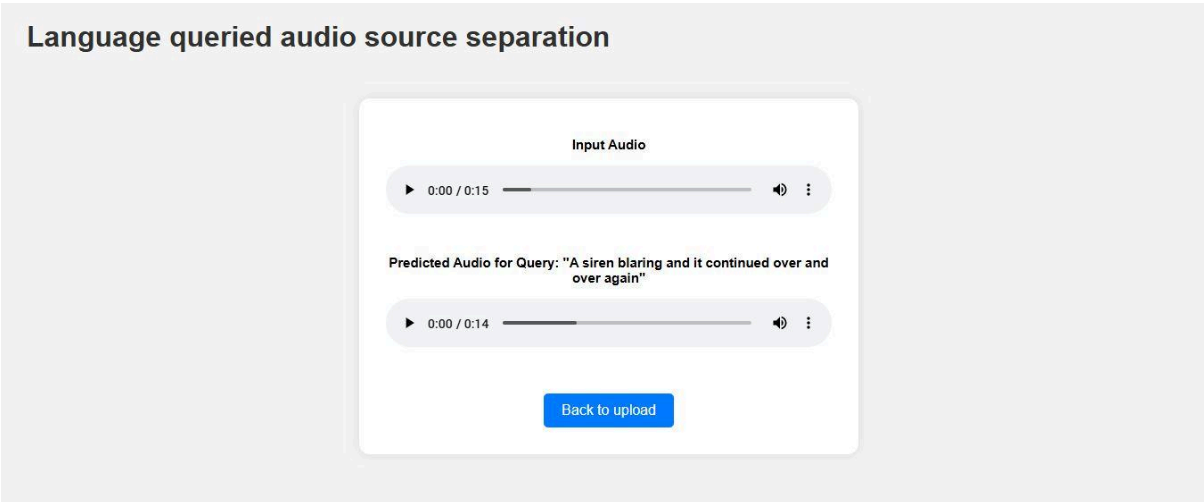


Figure 5.3 The results interface.

5.7. Challenges and limitations :

The first limitation we faced relates to the accessibility of the audio mixtures. Most datasets either provided the clean target audio or YouTube links to download it. As a result, we had to generate our own sound mixtures, which took a lot of time, as properly training our model requires a large number of mixtures and that is to ensure that it gets exposed to the biggest number of captions.

Another issue we faced is with loading and pre-processing them. Many cloud-based environments have limited disk storage making it difficult to handle large amounts of data. In cases where we managed to do it, the environment would heavily slow down.

After making mixtures and pre-processing them, we faced another issue, regarding overlapping mixtures, whereas our resultant and target spectrogram would look nearly identical. This highly affected the training, as when the model was trained on such samples, it would just wind up reproducing the mixture rather than separating during inference.

5.8. Conclusion

In this chapter, we presented the tools and libraries used in our implementation. We then described the selected dataset and presented the pre-processing steps used on its audio. We then presented our training setup and the results we obtained. We concluded by highlighting key challenges and limitations that we faced.

Conclusion

In this thesis, we tackled the task of language-queried audio source separation, which involves separating a target source from a mixture based on a natural language query. For training, we relied on the Clotho dataset due to its size and availability, compared to other datasets such as AudioCaps, which only provide youtube links to download.

To build our model, we first reviewed existing implementations. Based on this investigation, we built our own approach, aiming to extend them and improve them.

In our proposed approach, we adapted a ResUNet++ architecture, originally developed for the medical field, into a language-queried audio source separation based one.

Our architecture extends the standard ResUNet by incorporating components such as a Stem block, Squeeze And Excitation blocks, Attention mechanisms and Atrous Spatial Pyramid Pooling in the bottleneck. This architecture has demonstrated better results, and an enhanced separating mask compared to the ResUNet architecture in the medical field, and to our knowledge, has not been explored yet for LASS tasks.

Additionally, we incorporated cross attention mechanisms along with ResUNet++ which constitutes a novel approach as prior LASS-related work mostly relied on Film conditioning. We also adopted FLAN-T5 as the text encoder, as it wields better results compared to other text encoders.

Compared to established baselines, our approach based on resunet++ provides a lighter technique with its three encoder-decoder architecture and remains efficient as it focuses on relevant information with attention blocks to enhance skip connections, squeeze and excitation blocks to highlight relevant channels, and atrous spatial pyramid pooling blocks to capture information from feature maps at different scales. With a smaller number of training epochs it achieved an SDR value of 2.41 dB and an SDRi of 8.37 dB, which both represent decent values seeing the limitations that we faced.

For future work, adapting our model to handle more diverse and naturally phrased queries would represent a valuable improvement. Additionally, rather than relying on the mixture's phase as the estimate for the target sound during Inverse Short Time Fourier Transform (ISTFT), a more effective approach could involve using a Neural network to estimate the phase.

Another key area of improvement concerns the issue of overlapping mixtures. As when used during training, the model tended to replicate the mixture without separating due to the target being too similar to the mixture. Moreover, the process of sorting and filtering such mixtures was time consuming, ultimately preventing us from training our model to its full potential and limiting its exposure to a wider variety of sounds.

References

[1] Cherry, E. C. (1953). Some experiments on the recognition of speech, with one and with two ears. *Journal of the acoustical society of America*, *25*, 975-979.

[2] *Daichi Kitamura*. (n.d.). http://d-kitamura.net/demo-ILRMA_en.html

[3] Benslimane, Z. (2023, February 1). Audio source separation using Non-Negative Matrix Factorization (NMF). *Medium*.

[4] Kim, M., Lee, J. H., & Jung, S. (2023, June 15). *Sound Demixing Challenge 2023 Music Demixing Track Technical Report: TFC-TDF-UNet v3*. arXiv.org. https://arxiv.org/abs/2306.09382

[5] Défossez, A., Bottou, L., & Bach, F. (2021, April 28). *Music source separation in the waveform domain*. https://hal.science/hal-02379796v2

[6] : *Google Colab*. (n.d.). https://colab.research.google.com/github/pytorch/pytorch.github.io/blob/master/assets/hub/sigsep_open-unmix-pytorch_umx.ipynb

[7] : Stöter, F., Uhlich, S., Liutkus, A., & Mitsufuji, Y. (2019). Open-Unmix - a reference implementation for music source separation. *The Journal of Open Source Software*, *4*(41), 1667. https://doi.org/10.21105/joss.01667

[8] : Luo, Y., Chen, Z., & Yoshioka, T. (2019, October 14). *Dual-path RNN: efficient long sequence modeling for time-domain single-channel speech separation*. arXiv.org. https://arxiv.org/abs/1910.06379v2

[9] : Luo, Y., & 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. https://doi.org/10.1109/taslp.2019.2915167

[10] : Tzinis, E., Wang, Z., & Smaragdis, P. (2020, September). Sudo rm-rf: Efficient networks for universal audio source separation. In *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)* (pp. 1-6). IEEE.

[11] Yin, X., Peng, X., Jiang, X., Xiong, Z., & Lu, Y. (2025, May 27). *Text-Queried audio source separation via hierarchical modeling*. arXiv.org. https://arxiv.org/abs/2505.21025

[12] Dong, H., Takahashi, N., Mitsufuji, Y., McAuley, J., & Berg-Kirkpatrick, T. (2022, December 14). *CLIPSep: Learning Text-queried Sound Separation with Noisy Unlabeled Videos*. arXiv.org. https://arxiv.org/abs/2212.07065

[13]*OmniSep: Unified Omni-Modality Sound Separation with Query-Mixup*. (n.d.). https://arxiv.org/html/2410.21269v1

[14] Liu, X., Liu, H., Kong, Q., Mei, X., Zhao, J., Huang, Q., Plumbley, M. D., & Wang, W. (2022, March 28). *Separate what you describe: Language-Queried Audio source separation*. arXiv.org. https://arxiv.org/abs/2203.15147

[15] Liu, X., Kong, Q., Zhao, Y., Liu, H., Yuan, Y., Liu, Y., Xia, R., Wang, Y., Plumbley, M. D., & Wang, W. (2023, August 9). *Separate anything you describe*. arXiv.org. https://arxiv.org/abs/2308.05037

[16] Jha, D., Smedsrud, P. H., Riegler, M. A., Johansen, D., Thomas, D. L., Halvorsen, P., & Johansen, H. D. (2019b, November 16). *ResUNet++: an advanced architecture for medical image segmentation*. arXiv.org. https://arxiv.org/abs/1911.07067

[17] Makino, S. (Ed.). (2018). *Audio source separation* (Vol. 433). Berlin, Germany: Springer

[18] Ma, H., Peng, Z., Li, X., Shao, M., Wu, X., & Liu, J. (2024). CLAPSep: Leveraging Contrastive Pre-trained Model for Multi-Modal Query-Conditioned Target Sound Extraction. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

[19] Girin, L., Gannot, S., & Li, X. (2018). Audio source separation into the wild. In *Elsevier eBooks* (pp. 53–78). https://doi.org/10.1016/b978-0-12-814601-9.00022-5

[20] Pons, J., Liu, X., Pascual, S., & Serrà, J. (2023, September 29). *GASS: Generalizing Audio Source Separation with Large-scale Data*. arXiv.org. https://arxiv.org/abs/2310.00140

[21] Hilsdorf, M. (2023, September 22). AI Music Source Separation: How it Works and Why It Is So Hard. *Medium*. https://medium.com/data-science/ai-music-source-separation-how-it-works-and-why-it-is-so-hard-187852e54752

[22] Défossez, A., Usunier, N., Bottou, L., & Bach, F. (2019). Demucs: Deep extractor for music sources with extra unlabeled data remixed. *arXiv preprint arXiv:1909.01174*.

[23] Tzinis, E., Wisdom, S., Hershey, J. R., Jansen, A., & Ellis, D. P. W. (2020). Improving universal sound separation using sound classification. *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 96–100. https://doi.org/10.1109/icassp40776.2020.9053921

[24] *Khan Academy*. (n.d.). https://www.khanacademy.org/science/physics/mechanical-waves-and-sound/sound-topic/v/production-of-sound

[25] *Physics tutorial: Pitch and frequency*. (n.d.). https://www.physicsclassroom.com/class/sound/Lesson-2/Pitch-and-Frequency

[26] The Editors of Encyclopaedia Britannica. (1998, July 20). *Amplitude | Definition & Facts*. Encyclopedia Britannica. https://www.britannica.com/science/amplitude-physics

[27] Testa, M. (2024, September 2). *Understanding digital audio: sampling, quantization, and more — TheAudioPod.Com®*. TheAudioPod.Com®. https://www.theaudiopod.com/blog/xank7qi4qjlgxsltcwz3wdkzjdnagy

[28] *2.1. Sampling period and rate — Digital Signals Theory*. (n.d.). https://brianmcfee.net/dstbook-site/content/ch02-sampling/Sampling.html

[29] Gribble, P. (n.d.). *9. Signals, sampling & filtering*. https://gribblelab.org/teaching/scicomp2014/09_Signals_sampling_filtering.html

[30] Zilliz. (2025, February 28). Getting Started with Audio Data: Processing Techniques and Key Challenges. *Medium*. https://medium.com/%40zilliz_learn/getting-started-with-audio-data-processing-techniques-and-key-challenges-420dc5233163

[31] *FFT*. (n.d.). https://www.nti-audio.com/en/support/know-how/fast-fourier-transform-fft#:~:text=The%20%22Fast%20Fourier%20Transform%22%20(,contains%203%20distinct%20dominant%20frequencies.

[32] Neutelings, I. (2024, September 2). *Fourier series & synthesis*. https://tikz.net/fourier_series/

[33] *01_ShortTime_Fourier_Transform_and_Spectrograms*. (n.d.). https://books.mercity.ai/books/Audio-Analysis-and-Synthesis---Introduction-to-Audio-Signal-Processing/audio_analysis_techniques/01_ShortTime_Fourier_Transform_and_Spectrograms

[34] *Figure 4 Illustration of short-time Fourier transform on the test. . .* (n.d.). ResearchGate. https://www.researchgate.net/figure/Ilustration-of-short-time-Fourier-transform-on-the-test-signal-xt_fig3_220650705

[35] *Google Colab*. (n.d.). https://colab.research.google.com/github/Speech-Interaction-Technology-Aalto-U/itsp/blob/main/Representations/Spectrogram_and_the_STFT.ipynb

[36] Awan, A. A. (2023, August 30). *What are Neural Networks?* https://www.datacamp.com/blog/what-are-neural-networks

[37] GeeksforGeeks. (2025, April 5). *Activation functions in Neural Networks*. GeeksforGeeks. https://www.geeksforgeeks.org/activation-functions-neural-networks/

[38] Brownlee, J. (2021, January 21). *How to choose an activation function for deep learning*. MachineLearningMastery.com.
https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/

[39] *Layers in a Neural Network explained*. (n.d.). Deeplizard. https://deeplizard.com/learn/video/FK77zZxaBoI

[40] Nigam, J. (2025, April 28). Why ANN (Artificial Neural Network) is not preferred for Image Processing. *Medium*. https://medium.com/@jyotinigam2370/why-ann-artificial-neural-network-is-not-preferred-for-image-processing-7267d296a368

[41]B, M. (2023, October 1). Neural networks - murali B - medium. *Medium*. https://medium.com/@muralibabu1729/neural-networks-3912e9f95be6

[42] Khadka, N. (2023, May 1). *How Gradient Descent algorithm works*. Dataaspirant. https://dataaspirant.com/gradient-descent-algorithm/

[43] *Gradient descent*. (n.d.). Questions and Answers in MRI. https://mriquestions.com/back-propagation.html

[44] *Lecture 5 - CS50's Introduction to Artificial Intelligence with Python*. (n.d.). https://cs50.harvard.edu/ai/2023/notes/5/

[45] Shreshtha, S., & Arun, S. (2023). State of the art convolutional neural networks. *International Journal of Performability Engineering*, *19*(5), 342. https://doi.org/10.23940/ijpe.23.05.p6.342349

[46] Li, T., Chen, J., Hou, H., & Li, M. (2021, January). Sams-net: A sliced attention-based neural network for music source separation. In *2021 12th International Symposium on Chinese Spoken Language Processing (ISCSLP)* (pp. 1-5). IEEE.

[47] Yamashita, R., Nishio, M., Gian, R. K., DO, & Togashi, K. (2018b). Convolutional neural networks: an overview and application in radiology. *Insights Into Imaging*, *9*(4), 611–629. https://doi.org/10.1007/s13244-018-0639-9

[48] Admin. (2019, October 18). *What is Convolutional Neural Network? What are all the layers used in it?* I2tutorials. https://www.i2tutorials.com/what-is-convolutional-neural-network-what-are-all-the-layers-used-in-it/

[49] Shreshtha, S., & Arun, S. (2023). State of the art convolutional neural networks. *International Journal of Performability Engineering*, *19*(5), 342. https://doi.org/10.23940/ijpe.23.05.p6.342349

[50] *7.3. Padding and Stride — Dive into Deep Learning 1.0.3 documentation*. (n.d.). https://d2l.ai/chapter_convolutional-neural-networks/padding-and-strides.html

[51] Chowdhury, M. (2024, November 18). Padding and strides in CNN - Minhaz Chowdhury - medium. *Medium*. https://medium.com/@minhazc.engg/padding-and-strides-in-cnn-58dc56493887

[52] Zafar, A., Saba, N., Arshad, A., Alabrah, A., Riaz, S., Suleman, M., Zafar, S., & Nadeem, M. (2024). Convolutional Neural Networks: A comprehensive evaluation and benchmarking of pooling layer variants. *Symmetry*, *16*(11), 1516. https://doi.org/10.3390/sym16111516

[53]Guissous, Alla Eddine. (2019). Skin Lesion Classification Using Deep Neural Network. 10.48550/arXiv.1911.07817.

[54] Ronneberger, O., Fischer, P., & Brox, T. (2015, May 18). *U-NET: Convolutional Networks for Biomedical Image Segmentation*. arXiv.org. https://arxiv.org/abs/1505.04597

[55] Jansson, A., Humphrey, E., Montecchio, N., Bittner, R., Kumar, A., & Weyde, T. (2017). Singing voice separation with deep u-net convolutional networks.

[56] Henning, R., Choudhry, A., & Ma, M. (2021). Deep learning based music source separation. *SCSU Journal of Student Scholarship*, *1*(2), 3.

[57] He, K., Zhang, X., Ren, S., & Sun, J. (2015, December 10). *Deep residual learning for image recognition*. arXiv.org. https://arxiv.org/abs/1512.03385

[58] Oktay, O., Schlemper, J., Folgoc, L. L., Lee, M., Heinrich, M., Misawa, K., Mori, K., McDonagh, S., Hammerla, N. Y., Kainz, B., Glocker, B., & Rueckert, D. (2018, April 11). *Attention U-Net: Learning where to look for the pancreas*. arXiv.org. https://arxiv.org/abs/1804.03999

[59] Hu, J., Shen, L., Albanie, S., Sun, G., & Wu, E. (2017, September 5). *Squeeze-and-Excitation networks*. arXiv.org. https://arxiv.org/abs/1709.01507

[60] *Medium*. (n.d.). Medium. https://medium.com/@tahasamavati/squeeze-and-excitation-explained

[61] s*queeze and Excitation Block [34].* (n.d.). ResearchGate. https://www.researchgate.net/figure/Squeeze-and-Excitation-Block-34_fig1_378722442

[62] Azizi, F., Ramadhan, M. M. L., & Jatmiko, W. (2023). Encoder-Decoder with Atrous Spatial Pyramid Pooling for Left Ventricle Segmentation in Echocardiography. *Jurnal Ilmu Komputer Dan Informasi*, *16*(2), 163–169. https://doi.org/10.21609/jiki.v16i2.1165

[63] Singh, V., & Singh, V. (2024, January 10). *DeepLabV3 & DeepLabV3+ The Ultimate PyTorch guide*. LearnOpenCV – Learn OpenCV, PyTorch, Keras, Tensorflow With Code, & Tutorials. https://learnopencv.com/deeplabv3-ultimate-guide/

[64] GeeksforGeeks. (2023, April 14). *Dilated convolution*. GeeksforGeeks. https://www.geeksforgeeks.org/dilated-convolution/

[65] GeeksforGeeks. (2025c, April 8). *Natural Language Processing (NLP) overview*. GeeksforGeeks. https://www.geeksforgeeks.org/natural-language-processing-overview/

[66] Yuan, Y., Liu, X., Liu, H., Plumbley, M. D., & Wang, W. (2024, September 11). *FlowSep: Language-Queried Sound Separation with Rectified Flow Matching*. arXiv.org. https://arxiv.org/abs/2409.07614

[67] GeeksforGeeks. (2024, January 5). *Word embeddings in NLP*. GeeksforGeeks. https://www.geeksforgeeks.org/word-embeddings-in-nlp/

[68] Ferrer, J. (2024, January 9). *How Transformers Work: A detailed exploration of transformer architecture*. https://www.datacamp.com/tutorial/how-transformers-work

[69] Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., ... & Wei, J. (2024). Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, *25*(70), 1-53. FLAN T5

[70] Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2018, October 11). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. arXiv.org. https://arxiv.org/abs/1810.04805

[71] Yenduri, G., M, R., G, C. S., Y, S., Srivastava, G., Maddikunta, P. K. R., G, D. R., Jhaveri, R. H., B, P., Wang, W., Vasilakos, A., V., & Gadekallu, T. R. (2023, May 11). *Generative Pre-trained Transformer: a comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions*. arXiv.org. https://arxiv.org/abs/2305.10435

[72] Mljourney. (2025, May 17). *What are the main components of a transformer model?* ML Journey. https://mljourney.com/what-are-the-main-components-of-a-transformer-model/

[73] Legarski, R. (2024). *LLMs: From origin to present and future applications*. SolveForce. https://books.google.dz/books?id=0E8eEQAAQBAJ

[74] Helwan, A. (2025, February 20). Cross-Modal Attention - Abdulkader Helwan - Medium. *Medium*. https://abdulkaderhelwan.medium.com/cross-modal-attention-efa7a774c80a

[75] Nayak, V. (2023, June 18). *Unleashing the power of cross attention: Revolutionizing natural language processing*. https://www.linkedin.com/pulse/unleashing-power-cross-attention-revolutionizing-natural-nayak

[76] *Audio-Language Datasets of Scenes and Events: A survey*. (2025). IEEE Journals & Magazine | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/10854210

[77] J. F. Gemmeke *et al.*, "Audio Set: An ontology and human-labeled dataset for audio events," *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, LA, USA, 2017, pp. 776-780, doi: 10.1109/ICASSP.2017.7952261.

[78] Chen, H., Xie, W., Vedaldi, A., & Zisserman, A. (2020, April 29). *VGGSound: a large-scale Audio-Visual dataset*. arXiv.org. https://arxiv.org/abs/2004.14368

[79] Kim, C. D., Kim, B., Lee, H., & Kim, G. (2019, June). Audiocaps: Generating captions for audios in the wild. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (pp. 119-132).

[80] Mei, X., Meng, C., Liu, H., Kong, Q., Ko, T., Zhao, C., Plumbley, M. D., Zou, Y., & Wang, W. (2024). WAVCapS: a CHATGPT-Assisted Weakly-Labelled audio captioning dataset for Audio-Language multimodal research. *IEEE/ACM Transactions on Audio Speech and Language Processing*, *32*, 3339–3354. https://doi.org/10.1109/taslp.2024.3419446

[81] Drossos, K., Lipping, S., & Virtanen, T. (2019, October 21). *Clotho: an audio captioning dataset*. arXiv.org. https://arxiv.org/abs/1910.09387

[82] Sun, L., Xu, X., Wu, M., & Xie, W. (2023, September 20). *Auto-ACD: a large-scale dataset for Audio-Language Representation learning*. arXiv.org. https://arxiv.org/abs/2309.11500

[83] Dcase. (n.d.). *Language-Queried Audio Source Separation - DCASE*. https://dcase.community/challenge2024/task-language-queried-audio-source-separation

[84] Gu, Z., Zhou, W., Zhang, J., Xu, G., & Jia, Y. (2024). *Network simulation and evaluation: Second International Conference, NSE 2023, Shenzhen, China, November 22–24, 2023, proceedings, part I* (Communications in Computer and Information Science). Springer Nature Singapore. https://books.google.dz/books?id=GVoXEQAAQBAJ

[85] *Source Separation Benchmarks - Signal to Distortion Ratio Study*. (n.d.). Music AI. https://music.ai/blog/research/source-separation-benchmarks/

[86] Xiao, F., Wang, W., Xu, D., Qi, S., Zhang, K., Zhu, Q., & Guan, J. (2024). Language-queried audio source separation with GPT-based text augmentation and ideal ratio masking. *DCASE2024 Challenge, Tech. Rep*.

[87] Elizalde, B., Deshmukh, S., Al Ismail, M., & Wang, H. (2023, June). Clap learning audio concepts from natural language supervision. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 1-5). IEEE.

[88] Yuan, Y., Liu, X., Liu, H., Plumbley, M. D., & Wang, W. (2024, September 11). *FlowSep: Language-Queried Sound Separation with Rectified Flow Matching*. arXiv.org. https://www.arxiv.org/abs/2409.07614v1

[89] Yin, H., Bai, J., Xiao, Y., Wang, H., Zheng, S., Chen, Y., Das, R. K., Deng, C., & Chen, J. (2024, September 20). *Exploring Text-Queried Sound Event Detection with Audio Source Separation*. arXiv.org. https://arxiv.org/abs/2409.13292

[90] Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N., & Liang, J. (2018, July 18). *UNet++: A Nested U-Net Architecture for Medical Image Segmentation*. arXiv.org.

[91] Georgian. (2023, October 21). The practical guide to LLMs: Flan-T5 - Georgian Impact Blog - medium. *Medium*. https://medium.com/georgian-impact-blog/the-practical-guide-to-llms-flan-t5-6d26cc5f14c0

[92] Gupta, G. (2024, November 20). Understanding the T5 model: A Comprehensive guide - Gagan Gupta - Medium. *Medium*. https://medium.com/@gagangupta_82781/understanding-the-t5-model-a-comprehensive-guide-b4d5c02c234b

[93] Tan, Siao & Lee, Chin-Poo & Lim, Kian & Tee, Connie & Alqahtani, Ali. (2024). QARR-FSQA: Question-Answer Replacement and Removal Pretraining Framework for Few-Shot Question Answering. IEEE Access. PP. 1-1. 10.1109/ACCESS.2024.3487581.

[94] *Inverse Short Time Fourier Transform algorithm described in words*. (n.d.). Signal Processing Stack Exchange. https://dsp.stackexchange.com/questions/2757/inverse-short-time-fourier-transform-algorithm-described-in-words

[95] *Phase — Open-Source Tools & Data for Music Source separation*. (n.d.). https://source-separation.github.io/tutorial/basics/phase.html

[96] *istft - Inverse short-time Fourier transform - MATLAB*. (n.d.). https://www.mathworks.com/help/signal/ref/istft.html

[97] Rad, A. B., & Virtanen, T. (2012, May). Phase spectrum prediction of audio signals. In *2012 5Th international symposium on communications, control and signal processing* (pp. 1-5). IEEE.

[98]*PyTorch*. (n.d.). https://ai.meta.com/tools/pytorch/

[99] *Transformers*. (n.d.). https://huggingface.co/docs/transformers/en/index

[100] *librosa — librosa 0.11.0 documentation*. (n.d.). https://librosa.org/doc/latest/index.html

[101] *Introduction to PyDub | Python*. (n.d.). https://campus.datacamp.com/courses/spoken-language-processing-in-python/manipulating-audio-files-with-pydub

[102] *NumPy*. (n.d.). https://numpy.org/

[103] *pandas - Python Data Analysis Library*. (n.d.). https://pandas.pydata.org/

[104] *scikit-learn: machine learning in Python — scikit-learn 1.6.1 documentation*. (n.d.). https://scikit-learn.org/stable/index.html

[105] *Matplotlib — Visualization with Python*. (n.d.). https://matplotlib.org/

[106] Hanna, K. T., Mixon, E., & Wigmore, I. (2025, May 8). *What is Google Drive?* Search Mobile Computing. https://www.techtarget.com/searchmobilecomputing/definition/Google-Drive

[107] GeeksforGeeks. (2025, January 2). *How to use Google Colab*. GeeksforGeeks. https://www.geeksforgeeks.org/how-to-use-google-colab/

[108] *RunPod: Cloud-Based AI & ML GPU Access Revolutionized | DeepGram*. (n.d.). Deepgram. https://deepgram.com/ai-apps/runpod

[109] *Pydub mix 2 sounds with different energy levels*. (n.d.). Stack Overflow. https://stackoverflow.com/questions/49368287/pydub-mix-2-sounds-with-different-energy-levels/49389518

[110] *PAD | AI-Powered Audio Solutions | Music AI*. (n.d.). Music AI. https://music.ai/modules/utilities/pad/

[111] Stoller, D. (2017, November 28). *Spectrogram input normalisation for neural networks*. Dans World. https://dans.world/Spectrogram-input-normalisation-for-neural-networks/

[112] *torch.nn.functional.cosine_similarity — PyTorch 2.7 documentation*. (n.d.). https://docs.pytorch.org/docs/stable/generated/torch.nn.functional.cosine_similarity.html

[113] Agarwal, R. (2023, September 13). *Complete guide to the ADAM Optimization Algorithm*. Built In. https://builtin.com/machine-learning/adam-optimization

[114] Bodner, B. (2024, November 19). 10 Must-Have Pytorch Schedulers you didn't know you need. *Medium*. https://medium.com/@benjybo7/10-must-have-schedulers-that-will-boost-your-models-performance-2ff0c446ac98

[115] Chugani, V. (2025, May 16). *A gentle introduction to learning rate schedulers*. MachineLearningMastery.com. https://machinelearningmastery.com/a-gentle-introduction-to-learning-rate-schedulers/

[116] *Proceedings of the 2024 3rd International Conference on Engineering Management and Information Science (EMIS 2024)*. (n.d.). Google Books. https://books.google.dz/books?id=2wYUEQAAQBAJ

[117] Arik, S. O., Jun, H., & Diamos, G. (2018). Fast spectrogram inversion using Multi-Head convolutional neural networks. *IEEE Signal Processing Letters*, *26*(1), 94–98. https://doi.org/10.1109/lsp.2018.2880284