MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

UNIVERSITY OF SAAD DAHLEB – BLIDA 1

FACULTY OF SCIENCES

DEPARTMENT OF COMPUTER SCIENCE

Report submitted for the fulfillment of the master degree in Computer Science

Option : Computer Systems and Networks

# Graduation Project

## Audio search engine based on joint embedding

Realized by:

- ✓ Kadi Abdelhakim

Supervisor :

- ✓ Mr A. KAMECHE

Examiners :

- ✓ Mr Ould Khaoua Mohamed
- ✓ Mme Ykhlef Hadjer

Academic year : 2023/2024

# ABSTRACT

Audio retrieval based on language allows users to search for audio content using natural language queries. This technology, which has gained popularity in recent years, has numerous applications in fields such as entertainment, education, and healthcare. To achieve our goal, we conducted several tests and validated our results using a phonetic subtitle dataset, converting the sentences into vectors using sBert. We extracted log mel spectrograms from the corresponding audio files. Our analysis was further deepened by applying a convolutional neural network (CNN) architecture to extract features from the log mel spectrograms. We then calculated the similarity with subtitles using the cosine metric. This research underscores the potential for enhanced audio retrieval systems, paving the way for more intuitive and effective methods for accessing audio information.

**Keywords:** Language-based audio retrieval, natural language queries, log mel spectrogram, sBert

# RÉSUMÉ

La récupération audio basée sur la langue permet aux utilisateurs de rechercher du contenu audio en utilisant des requêtes en langage naturel. Cette technologie, qui a gagné en popularité ces dernières années, trouve de nombreuses applications dans divers domaines tels que le divertissement, l'éducation et la santé. Pour atteindre notre objectif, nous avons mené plusieurs tests et validé nos résultats en utilisant l'ensemble de données de légende phonétique, convertissant les phrases en vecteurs à l'aide de sBert. Nous avons extrait les spectrogrammes log mel des fichiers audio correspondants. Notre analyse a été approfondie en appliquant une architecture de réseau de neurones convolutifs (CNN) pour extraire les caractéristiques des spectrogrammes log mel. Nous avons ensuite calculé la similarité avec les sous-titres en utilisant la métrique du cosinus. Cette recherche souligne le potentiel des systèmes de récupération audio améliorés, ouvrant la voie à des méthodes plus intuitives et efficaces pour accéder à l'information audio.

**Mots-clés :** Récupération audio basée sur la langue, requêtes en langage naturel, spectrogramme log mel, sBert

# ملخص

تتيح استرجاع الصوت بناءً على اللغة للمستخدمين البحث عن المحتوى الصوتي باستخدام استفسارات باللغة الطبيعية. هذه التقنية، التي اكتسبت شعبية في السنوات الأخيرة، لها العديد من التطبيقات في مجالات مثل الترفيه والتعليم والرعاية الصحية. لتحقيق هدفنا، أجرينا عدة اختبارات وحققنا نتائجنا باستخدام مجموعة بيانات العناوين الصوتية، حيث قمنا بتحويل الجمل إلى متجهات باستخدام .sBert استخرجنا الأطياف الميلية اللوغاريتمية من الملفات الصوتية المقابلة. تم تعميق تحليلنا بشكل أكبر من خلال تطبيق هيكلية الشبكة العصبية الالتفافية (CNN) لاستخراج الميزات من الأطياف الميلية اللوغاريتمية. ثم قمنا بحساب التشابه مع العناوين باستخدام مقياس جيب التمام (الكوساين). هذا البحث يؤكد على الإمكانيات لتحسين نظم استرجاع الصوت، مما يمهد الطريق لطرق أكثر فعالية وسهولة للوصول إلى المعلومات الصوتية

الكلمات الرئيسية: ا استرجاع الصوت بناءً على اللغة، استفسارات باللغة الطبيعية، الطيف الميلي اللوغاريتمي، sBert

# Acknowledgement

First, we thank ALLAH Sobhanou for giving us the will and courage to undertake and complete this work. We would like to express our deepest and most sincere gratitude to our promoter and coach ABDALLAH HICHAM KAMECHE, University of Blida, for guiding and enriching our work. We thank him for his availability, his valuable advice, his confidence despite our rather light knowledge in the field of audio processing. We also thank him for his attention to detail and for giving us the opportunity to conduct this research and providing us invaluable guidance throughout our work that led to this work been completed.

Our thanks also go to the members of the jury for agreeing to examine our work and enrich it with their proposals. We would also like to thank the faculty and administrative staff at BLIDA University who have contributed to the success of our university studies.

Finally, many thanks go to all the people who have contributed to the completion of our research work directly or indirectly.

Thanks again.

# Table of Contents

## List of Figures :

# Introduction General

With the exponential growth of digital media, particularly audio content ranging from music and podcasts to spoken-word archives and voice recordings, the demand for efficient and accurate audio retrieval systems has surged. Users now expect to locate specific audio segments in massive databases swiftly and effortlessly. Traditional audio search engines, primarily reliant on keyword-based searches or manual tagging, are increasingly inadequate. These conventional methods often fall short in capturing the inherent richness and depth of audio content. For example, keyword-based searches depend on predefined metadata or textual descriptions, which can result in mismatches between the search query and the actual content. Additionally, they fail to capture the semantic meaning of sound—nuances such as emotions, tone, and context—which are critical for meaningful retrieval in domains like music, education, media production, and even security.

Moreover, the inherent characteristics of audio data pose unique challenges that are not as prevalent in text-based search systems. Acoustic variability—the differences in how audio is produced, recorded, and perceived—compounds the difficulty of creating effective search tools. Furthermore, sound is subject to subjective interpretation: the same piece of audio can evoke different emotions or be understood differently by various listeners. Coupled with the linguistic diversity of audio files, where multiple languages or dialects can be involved, it becomes clear that a more sophisticated approach is needed.

In response to these challenges, joint embedding has emerged as a powerful technique to improve audio retrieval systems. Joint embedding enables both audio content and textual queries to be represented within a shared vector space [1], allowing for more accurate matching and deeper semantic understanding. By mapping both types of data into a unified space, it becomes possible to assess their similarity in a more meaningful way, which can significantly enhance search precision. This method does not rely on keyword matching alone but instead leverages the underlying content, creating opportunities for better search experiences, particularly for applications where nuanced interpretation of sound is necessary.

In this context, our research focuses on developing a high-performing audio search engine based on joint embedding techniques. We aim to address the limitations of existing systems by proposing a joint architecture that can represent audio and text within the same vector space, leading to more accurate retrieval.

This work is timely, given the increasing demand for robust search systems in industries that rely heavily on multimedia content.

## 1. Motivations:

The principal objective of this research is to design and implement a joint embedding architecture—specifically an Encoder-Encoder model—that effectively represents both audio data and textual queries within a common vector space. This unified representation will enable more precise matching between audio content and text, overcoming the aforementioned challenges associated with acoustic variability and subjective interpretation. Moreover, we aim to develop optimization mechanisms to enhance the similarity between the vectorized representations of audio and text, thus improving the overall accuracy and relevance of the search results.

## 2. Objectives:

The principal objective of this research is to develop a joint embedding architecture (Encoder-Encoder) that can efficiently represent audio data and textual queries within a unified vector space. Furthermore, it is imperative to devise optimization mechanisms that enhance the similarity between audio and text representations, thereby facilitating more accurate and precise audio retrieval.

## 3. Thesis Organization:

This thesis is organized as follows:

- **Chapter 1:** This chapter provides a comprehensive overview of the fundamental principles of audio signal processing and natural language processing, along with a review of related work in the field.
- **Chapter 2:** This chapter outlines our proposed approach, including a detailed description of the architectures and methodologies employed.
- **Chapter 3:** This chapter presents the empirical results of our research, offers a thorough analysis of the findings, and discusses potential directions for future work.

# Chapter 1: fundamental concepts

## 1.1 Introduction

In this chapter, we will explore two significant areas of digital signal processing: audio processing and text processing. These domains involve the use of algorithms and techniques to manipulate and analyze data, but the types of data and applications are different.

In the first part of this chapter, we will focus on audio processing. We will start by discussing the fundamentals of sound waves, audio signals, and sound parameters. We will then explore the various features of audio signals and the extraction pipeline. Following this, we will delve into the time and frequency domains of audio signals, including the Fourier transform and its different forms. We will also examine Mel spectrograms and Mel-frequency cepstral coefficients (MFCC) as essential features for audio processing.

In the second part of this chapter, we will concentrate on text processing. Specifically, we will address the field of natural language processing (NLP) and its various techniques for analyzing and processing textual data. We will explore the basics of tokenization, stemming, and lemmatization, as well as techniques for handling stop words and text encoding. Additionally, we will discuss the bag-of-words (BOW) model, TF-IDF, word embeddings, and some popular algorithms such as Word2Vec [1], GloVe, BERT [2], and sBERT [3].

Overall, this chapter will provide an overview of the fundamental concepts and techniques used in both audio and text processing, with a focus on practical applications and concrete examples.

## 1.2 Pipeline ML(CNN, Autoencoder):

Deep learning (DL), a significant branch of machine learning (ML) and artificial intelligence (AI), has revolutionized the handling of unstructured and large datasets, surpassing traditional ML techniques. This advancement has profoundly impacted diverse fields such as speech recognition, healthcare, autonomous driving, cybersecurity, and predictive analytics. Despite its successes, designing effective deep learning models remains challenging due to the complexity and dynamic nature of real-world problems. This paper surveys various deep learning models, including Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Generative Models, Deep Reinforcement Learning (DRL), and Deep Transfer Learning. We explore their structures, applications, advantages, and limitations. Additionally, we analyze the performance of six prominent models—CNN, Simple RNN, Long Short-Term Memory (LSTM), Bidirectional LSTM, Gated Recurrent Unit (GRU), and Bidirectional GRU—using the IMDB, ARAS, and Fruit-360 datasets [4].

### 1.2.1 2. Convolutional Neural Networks (CNN):

Convolutional Neural Networks (CNNs) are a powerful class of deep learning models widely used in tasks such as image classification, object detection, and speech recognition. CNNs automatically extract features from data through convolutional operations, eliminating the need for manual feature extraction. Their architecture, inspired by visual perception, includes several key components:

- **Convolutional Layer**: This layer performs convolution operations using learnable kernels to extract features. Lower layers capture basic features like edges and textures, while higher layers detect more complex patterns.
- **Pooling Layer**: This layer reduces the dimensionality of data through operations like Max Pooling or Average Pooling, which helps in reducing computational load and preventing overfitting while maintaining robustness against distortions.
- **Fully Connected Layer**: Positioned at the end of the CNN, this layer connects every neuron to all neurons in the previous layer. It flattens the output from the convolutional and pooling layers to make predictions or classifications.

### 1.2.2 3. Autoencoder

Autoencoders are neural networks designed to learn a compressed representation of data through unsupervised learning. They consist of two main components:

- **Encoder**: Maps input data to a lower-dimensional feature space.
- **Decoder**: Reconstructs the original data from this feature space.

Autoencoders are useful for tasks like dimensionality reduction, feature extraction, and anomaly detection. They come in various forms:

- **Regularized Autoencoders**: Include models such as Sparse Autoencoder (SAE) and Denoising Autoencoder (DAE), which are used for tasks like classification and feature learning.
- **Variational Autoencoders (VAE)**: Employ probabilistic methods to generate new data by learning distributions over the latent space. VAEs

include an encoder that estimates a posterior distribution and a decoder that generates new samples from this distribution, facilitating data generation and representation learning.

## 1.3 Textual Feature Extraction

### 1.3.1 Introduction:

Text processing involves the systematic manipulation and analysis of textual data to extract meaningful insights. This encompasses a variety of techniques for extracting, cleaning, transforming, and analyzing text data to uncover information, patterns, and trends. Such methods are applied across different applications, including natural language processing (NLP).

## 1.4 Natural language processing:

Natural language processing (NLP) [5] encompasses various submodules designed to handle different aspects of text and language data. Improved training data plays a crucial role, with large datasets from diverse sources like social media and online texts enhancing model performance in understanding and generating natural language. Multilingual and cross-lingual NLP research focuses on leveraging vast amounts of multilingual data to improve tasks such as information retrieval and machine translation. Domain-specific NLP utilizes large datasets from specialized fields, such as medical or legal texts, to create models tailored for specific domains, improving precision in those areas.

Ethical considerations are also becoming increasingly important, addressing biases in training data and ensuring fair, inclusive NLP applications. Future research will likely explore how to handle these challenges and incorporate principles of accountability and transparency.

Class-based language modeling (CBLM) improves language modeling efficiency by grouping words into classes based on semantic or syntactic similarity, reducing model complexity and enhancing generalization. Techniques like unsupervised clustering methods are used to classify words, which helps in handling out-of-vocabulary terms and capturing broader language patterns.

### 1.4.1 Key NLP techniques include:

- **Machine Translation**: Automatically translating text between languages, a complex task requiring deep semantic and syntactic understanding.

- **Discourse Analysis**: Examining text to understand relationships between segments, such as identifying speech acts and discourse structures.
- **Morphological Analysis**: Breaking down words into their morphemes, crucial for languages with complex word structures.
- **Natural Language Generation and Understanding**: Converting data into human-readable language and formalizing it for computational purposes.
- **Named Entity Recognition**: Identifying and classifying entities like people, organizations, and locations within text.
- **Text Parsing**: Analyzing grammatical structures of sentences, often facing challenges with ambiguities.
- **Speech Recognition**: Converting spoken language into text, a complex task that involves segmenting continuous speech.
- **Sentiment Analysis**: Determining the sentiment expressed in text, often used in marketing and social media analysis.
- **Word Boundary Detection**: Identifying word boundaries in languages with or without explicit delimiters.
- **Word Sense Disambiguation**: Determining the correct meaning of a word based on context.

### 1.4.2 Advanced NLP strategies include:

- **Named Entity Recognition (NER)**: Identifying and classifying entities in text using deep learning methods.
- **Attention Mechanisms**: Enhancing model performance by focusing on relevant parts of the input sequence.
- **Neural Machine Translation (NMT)**: Using deep learning for more accurate and fluent translations.
- **Transfer Learning**: Leveraging pre-trained models for specific tasks, improving performance on smaller datasets.

### 1.4.3 Conclusion and Recommendations

NLP, a field bridging linguistics, artificial intelligence, and computer science, focuses on enabling effective communication between computers and human language. Current research emphasizes unsupervised and semi-supervised learning methods. Future work should explore advanced transformer models like

BERT for contextual understanding and LSTM networks for capturing sequential patterns in text.

## 1.5 Audio Feature Extraction

### 1.5.1 Introduction:

As communication technologies have evolved, the role of sound in data processing has become increasingly significant. Sound encompasses a wide range of sources, including musical instruments, human speech, environmental noises, and other emitted sounds. These diverse audio sources add complexity to the data, with background noise often obscuring key information. This complexity underscores the importance of effective voice processing techniques.

This chapter explores various aspects of audio processing. We begin by examining the nature of sound and waveforms, followed by an exploration of sound parameters and the analog-to-digital conversion process. We will then delve into audio features within both the temporal and frequency domains before discussing sound visualization and feature extraction pipelines.

### 1.5.2 Sound and Waveforms:

Sound originates from the vibrations of objects, which create oscillations that interact with surrounding air molecules. These interactions cause variations in air pressure, forming sound waves. Waveforms graphically represent these variations over time, providing insights into the frequency, intensity, and duration of the sound. Unlike electromagnetic waves, which can travel through a vacuum, sound waves require a medium like air to propagate.

### 1.5.3 Sound Parameters:

Key sound parameters include:

- **Period (T)**: The interval between consecutive peaks or troughs of a sound wave.
- **Frequency (f)**: The reciprocal of the period, measured in Hertz (Hz). Higher frequencies correspond to higher pitches. Frequencies are perceived similarly if they differ by a factor of 2.
- **Pitch**: A logarithmic measure of frequency perception. The octave is divided into 1200 cents, and a pitch difference of 10-25 cents is perceptible.

- **Amplitude (A)**: Indicates the extent of pressure variation from zero, with larger amplitudes representing louder sounds.
- **Phase (ϕ)**: Represents the phase difference between two waveforms of the same frequency, assessing the frequency shift or cycle discrepancy between signals.
- **Sound Power**: The rate at which energy is emitted by a sound source in all directions, measured in Watts (W).
- **Sound Intensity**: The power per unit area, measured in Watts per square meter ($W/m^2$). Intensity level is represented in decibels (dB), where a 3 dB increase signifies a doubling of intensity. The decibel scale is given by: $dB(I) = 10 * \log10(I/I_0)$, with $I_0 = 10^{-12}$ $W/m^2$.
- **Threshold of Hearing**: The minimum sound intensity detectable by the human ear.
- **Threshold of Pain**: The sound intensity levels that cause discomfort or pain.

### 1.5.4 Audio Features:

Audio features describe various aspects of sound that are useful for intelligent audio systems. Features can be categorized based on their level of abstraction, temporal scope, musical relevance, signal domain, and machine learning applications. They are generally divided into two categories: temporal and spectral features.

#### *1.5.4.1 Temporal Domain Features:*

Temporal domain features analyze audio signals over time. Key features include:

- **Amplitude Envelope (AE)**: The maximum amplitude within a frame, which gives an indication of intensity and sensitivity to outliers. AE is used for music genre classification and onset detection.
- **Root Mean Square (RMS) Energy**: Represents the average quadratic energy within a frame, indicating volume with reduced sensitivity to outliers compared to AE. RMS is applied in audio segmentation and genre classification.

- **Zero-Crossing Rate (ZCR)**: Counts the number of times a signal crosses the zero-axis, useful for speech recognition and music processing, particularly for distinguishing percussive sounds and estimating pitch.

#### *1.5.4.2 Spectral Domain Features:*

Spectral domain features involve analyzing the frequency composition of audio signals. Key features include:

- **Fourier Transform (FT)**: Converts a time-domain signal into its frequency components, represented by amplitude and phase. The FT output includes Fourier coefficients that provide information about the presence of specific frequencies in the signal.
- **Inverse Fourier Transform (IFT)**: Reconstructs the original signal by summing sinusoidal components weighted by their amplitude and phase.
- **Discrete Fourier Transform (DFT)**: Converts continuous signals into discrete frequency components using a finite number of samples. The DFT reveals redundancy due to symmetry and typically focuses on frequencies up to the Nyquist frequency.
- **Short-Time Fourier Transform (STFT)**: Analyzes the frequency content of a signal over time by applying the Fourier Transform to overlapping short segments of the signal.
- **Mel Spectrogram**: Represents audio frequency content using the Mel scale, which aligns with human auditory perception. This involves transforming frequencies into Mel scale and calculating the Mel spectrogram.
- **Mel-Frequency Cepstral Coefficients (MFCCs)**: Extract key features relevant to human auditory perception by applying Mel scaling to the cepstrum and performing a discrete cosine transform.
- **Band Energy Ratio (BER)**: Measures the distribution of energy across different frequency bands, providing insights into the relative energy contribution of various frequency regions.
- **Spectral Centroid (SC)**: Estimates the "center of gravity" of a signal's spectrum, offering information on spectral characteristics and balance.
- **Bandwidth (BW)**: Measures the spread of frequency content in a signal's spectrum, indicating the range of frequencies present.

### 1.5.5 Sound Visualization (Spectrograms)

Spectrogram visualization involves creating a matrix by squaring the amplitude of the Short-Time Fourier Transform (STFT). Unlike the complex numbers in the original STFT, this matrix uses real numbers. Spectrograms are essential in audio AI applications as they provide crucial features for algorithmic analysis. In a spectrogram, the x-axis represents discrete time intervals, while the y-axis shows frequency components. This time-frequency representation allows for observing how different frequency components change over time.

### 1.5.6 Feature Extraction Pipeline

**A. Temporal Domain Feature Pipeline:** Starts with analog sound, performs ADC (sampling and quantization), and digitizes the sound. The signal is divided into frames to obtain images. Temporal domain features are calculated for each frame, followed by aggregation (mean, median, or Gaussian Mixture Models) to produce a feature vector or matrix for the entire sound.

**B. Spectral Domain Feature Pipeline:** Transitions from the time domain to the frequency domain using the Fourier Transform. After ADC and framing, windowing is applied to reduce spectral leakage, with overlapping frames addressing signal loss. The Fourier Transform is then applied, and features are calculated and aggregated to obtain feature vectors or matrices.

## 1.6 Related Works

### 1.6.1 Introduction:

Audio retrieval using human-generated subtitles is an emerging research area with potential applications across various fields. Key challenges in this domain include developing accurate and reliable subtitling systems and effective methods for aligning subtitles with the corresponding audio content. Machine learning algorithms have been investigated to accurately match subtitles to audio despite language, dialect, and tone variations, while natural language processing techniques have been explored to enhance subtitle matching accuracy. Despite notable advancements in recent years, many research questions and challenges remain unresolved. The following sections will review relevant studies in this field and discuss recent developments and future research directions.

### 1.6.2 CD-JKU [6]:

The study presents an innovative text-to-audio retrieval system that utilizes pretrained text and spectrogram transformers. This system maps audio recordings and textual descriptions into a unified space where related items are positioned close to one another. The authors highlight two critical factors for retrieval efficacy: the use of a self-attention-based audio encoder for embedding and the incorporation of extensive human-generated and synthetic datasets during pre-training. They also explored enhancing ClothoV2 captions with additional keywords, which led to slight improvements. Notably, their system achieved first place in the 2023 DCASE Challenge, surpassing the existing state-of-the-art performance on the ClothoV2 benchmark by 5.6 percentage points in mAP@10.

Natural language-based audio retrieval focuses on ranking audio recordings in relation to textual descriptions. Current approaches typically employ a dual-encoder framework, which converts both recordings and descriptions into high-level representations and aligns them within a shared embedding space. Ranking is then determined by the proximity between candidate audio recordings and textual descriptions within this space. This dual-encoder setup is prevalent in audio retrieval systems due to its efficiency in ranking and the advantage of utilizing pre-trained models. Typically, CNN architectures pre-trained on AudioSet [2] are used for audio encoding, while large transformer models like BERT [7] and RoBERTa [8] are employed for text encoding. Recent advancements have included the use of WavCaps [9], a large dataset featuring synthetic captions, which set a new benchmark in performance on ClothoV2 [10].

The authors’ approach for subtask 6b of the 2023 DCASE challenge builds on the dual-encoder method but introduces three significant innovations. First, they utilize the PaSST [11] audio spectrogram transformer instead of the traditional CNN14 [2] for audio embedding. PaSST [11], which has shown superior performance on AudioSet [2] and other benchmarks, includes Patchout during training to enhance speed and memory efficiency while also serving as a regularizer. This change notably enhances retrieval performance. Second, the authors pre-train their models on AudioCaps [12] and WavCaps [9]—large datasets with both human-generated and synthetic captions—to address data limitations. This pre-training yields considerably improved retrieval results. Third, they augment training captions with additional metadata and generate extra captions using the GPT-3.5-turbo API (ChatGPT). Although this augmentation helps reduce overfitting during fine-tuning, it provides only marginal performance gains. The system implementation and the keyword-augmented captions are accessible in the authors’ GitHub repository.

#### 1.6.3 QFORMER [13]:

The paper provides an overview of the audio retrieval system submitted for Task 6B of the DCASE2023 Challenge, which focuses on retrieving audio based on natural language queries. The authors' system combines a frozen pretrained audio encoder with a Qformer text encoder. For the contrastive learning component, the system leverages paired data from the AudioCaps and Clotho datasets, following the methodology of BLIP-2. The process involves encoding natural language queries using the text encoder, then retrieving the top-k audio embeddings. These embeddings are matched with the query text to form k data pairs, which are then reranked based on the model's matching performance to produce the final retrieval results. The system achieved a mean Average Precision (mAP) of 26.47% and a 16.02% recall at 1 (R@1) on the Clotho test set, showing improvements over the baseline system, which had an mAP of 22.2% and an R@1 of 13.0%.

Task 6B of the DCASE2023 challenge addresses the task of audio retrieval using natural language [14], a significant area in cross-modal research. Progress in this task is anticipated to improve the understanding of acoustic scenes and enable innovative manipulation of audio signals, with implications for fields such as audio content creation and acoustic scene analysis. The authors utilize the BEATs model [15] as their audio feature extractor. BEATs, a pre-trained model based on self-supervised learning, employs a discrete Tokenizer and a feature extractor guided by Masked Audio Modeling and is used for tasks such as classification.

The advent of CLIP [16] has greatly advanced the development of visual-language multimodal models, with subsequent models like BLIP-2 [17] extending this advancement to larger language models. In the realm of audio-related multimodal research, models such as AudioClip [18], Wav2Clip [19], and CLAP [20] have applied CLIP-style contrastive learning techniques to audio signals. The authors' approach incorporates contrastive learning of audio and text using the Qformer and its multitask training methodology introduced by BLIP-2 [21]. The Qformer, a transformer encoder akin to BERT, processes both individual audio and text inputs to generate single-modal representations and can also produce multimodal embeddings to assess the compatibility between audio and text.

#### 1.6.4 IRIT-UPS [22]:

The provided text offers an in-depth overview of the systems submitted for tasks 6a, "Automated Audio Captioning" (AAC), and 6b, "Language-Based Audio Retrieval" (LBAR) in the DCASE Challenge 2023. For task 6a, the authors employed four distinct submission strategies. The first utilized a conventional CNN14 encoder paired with a transformer decoder to generate captions for audio content. In the second approach, they replaced the CNN14 encoder with a ConvNeXt [23] model to improve audio representation. The third submission incorporated additional training data and introduced a novel task embedding technique to differentiate between various writing styles and audio types. The fourth approach involved an ensemble method that combined five models, each trained with different seeds, to enhance caption quality.

For task 6b, the authors adapted their AAC models and proposed a novel approach that leverages the AAC system's loss function to perform language-based audio retrieval. This strategy was implemented without requiring additional training, showcasing an efficient way to repurpose AAC models for LBAR tasks. The authors report that their most successful AAC and LBAR systems achieved a SPIDErFL score of 0.320 and an mAP@10 score of 0.269, reflecting significant improvements of 22.6% and 21.2% over the baseline scores for AAC and LBAR, respectively.

The AAC task aims to generate captions that succinctly describe audio content, relationships, and attributes within a single sentence. In contrast, the LBAR task is focused on retrieving specific audio recordings from a database based on free-form textual descriptions. The DCASE2023 challenge presented an opportunity to evaluate systems addressing these multimodal tasks, and the authors aimed to develop a unified AAC model capable of handling both tasks effectively.

The AAC system employed a standard encoder-decoder architecture, utilizing a pre-trained encoder for audio modeling and a transformer decoder for caption generation. Enhancements to this system included additional data from captioning datasets, various data augmentation techniques, improvements in beam search for inference, and the integration of a task embedding to assist with caption generation across different datasets. For the LBAR task, the authors proposed a novel strategy that utilizes the AAC model's loss function to rank audio files in response to textual queries. The source code for these systems is expected to be made available on GitHub following the conclusion of the challenge.

The remainder of the text details the systems and experimental setup used by the authors, presents and discusses the results, and concludes with final remarks on their findings and contributions to the field.

## 1.7 Overview

| Related Works | Audio Modelling | Acoustic Features | Text Modelling | Metric Monitored for Training | Dataset |
|---|---|---|---|---|---|
| CD-JKU [6] | PaSST | Log-mel energies | BERT RoBERTa | mAP | Clotho V2.1 |
| QFORMER [13] | BEATs | Mel energies | Qformer | recall | Clotho V2.1 |
| IRIT-UPS [22] | CNN | ConvNeXt-tiny | Transformer | FENSE | Clotho V2.1 |

## 1.8 Conclusion

This chapter has provided a thorough examination of audio processing, text processing, and related research. It began with an introduction that set the stage for the subsequent discussions. The first section focused on audio processing techniques, while the second section explored text processing within the context of natural language processing (NLP). The final section reviewed various studies and applications in the field. Overall, this chapter has established a robust foundation for further investigation into the intriguing domains of audio and text processing.

# Chapter 2: Proposed Methodology

## 2.1 Introduction

Audio retrieval with human-written captions is a process that involves multiple steps to enable users to search for and retrieve specific audio content using written descriptions. In this process, we begin with written text, which serves as a search query to retrieve the relevant audio content.

The first step is to extract features from the written text, which can be accomplished through natural language processing techniques. These features may include keywords, entities, and other relevant information that can be used to identify and retrieve audio content matching the search query.

Once the textual features are extracted, they are passed to a model trained to convert these features into audio features. This is achieved by mapping the text features to audio characteristics such as pitch, tone, and rhythm, which are specific to the audio content.

After the audio features are extracted, they are used to synthesize the audio content corresponding to the written text query. This process involves converting the audio features into audio signals that can be played through speakers or headphones, allowing users to hear the audio content that matches their search query.

Overall, audio retrieval with human-written captions is a powerful tool that can help users quickly find and access relevant audio content using written descriptions. By extracting features from text and mapping them to audio features, this process allows users to search for and retrieve audio content in a more efficient and accessible manner.

## 2.2 Used Pipeline

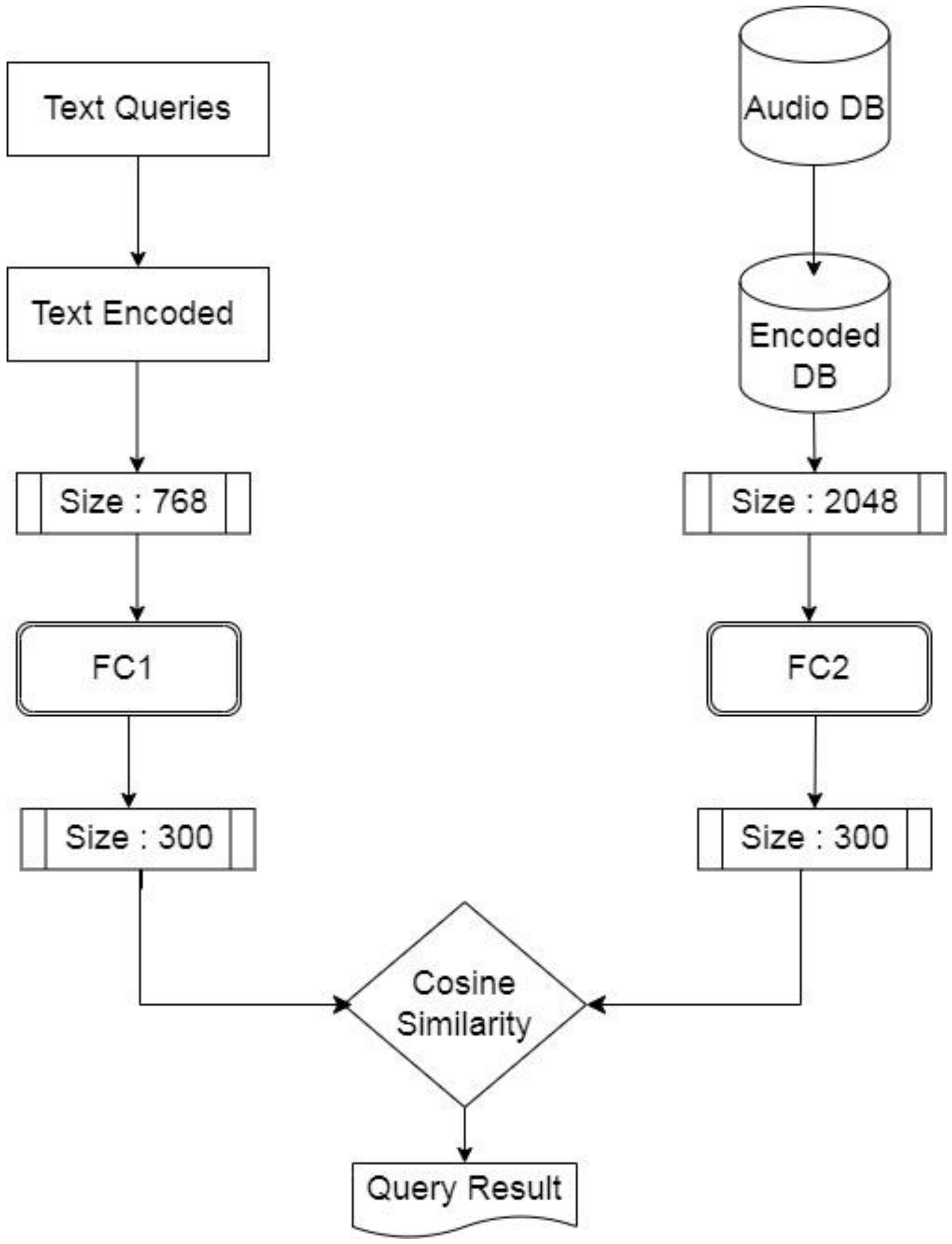

***Figure 1: Schema of Global Workflow***

The proposed system for text-based audio retrieval operates through a structured two-segment process designed to effectively match text queries with an audio database. This process is outlined as follows:

#### 2.2.1 Text Query Processing:

- **Text Encoder:** Text queries are processed through SBert (Sentence-BERT), which generates 768-dimensional embeddings for each text input.

- **Dimensionality Reduction:** These embeddings are subsequently reduced to 300 dimensions using a fully connected layer (FC1) to facilitate comparison with audio embeddings.

#### 2.2.2 2. Audio Database Processing:

- **Audio Feature Extraction:** Audio data are converted into 2048-dimensional feature vectors.
- **Dimensionality Reduction:** These feature vectors are then reduced to 300 dimensions using another fully connected layer (FC2) to match the text embeddings.

#### 2.2.3 3. Similarity Computation:

Cosine similarity is computed between the 300-dimensional text embeddings and the 300-dimensional audio feature vectors to retrieve the most relevant audio files based on the text queries.

## 2.3 Preprocessing Text

In the text preprocessing stage, we utilize Sentence-BERT (SBERT) embeddings to transform textual data into a numerical format that can be efficiently processed by machine learning models. This process involves several key steps.

First, we define global parameters, including the directory where our dataset is stored and the specific splits of the dataset we will be working with (development, validation, and evaluation). The SBERT model used for generating embeddings is the 'all-mpnet-base-v2', which produces 768-dimensional embeddings for each text input.

For each dataset split, we read the corresponding text data from CSV files. These files contain the textual information that needs to be embedded. Each text entry is associated with a unique identifier (tid).

As we iterate through the text data, we extract the raw text for each entry and use the SBERT model to encode this text into a fixed-dimensional embedding. These embeddings capture the semantic meaning of the text, allowing for effective downstream processing and analysis.

The generated text embeddings are stored in a dictionary, with the unique identifier as the key and the corresponding embedding as the value. Once all text data has been processed, we save the embeddings to a file using the pickle module.

This serialized file format ensures that the embeddings can be easily loaded and utilized in subsequent stages of our pipeline.

By preprocessing the text data in this manner, we convert unstructured text into a structured numerical format that retains the semantic information necessary for effective analysis and model training.

## 2.4 Preprocessing Audio

Preprocessing audio data is a critical step in preparing it for neural network analysis. The process involves transforming raw audio signals into a format that is more suitable for machine learning models. This section outlines the detailed steps involved in preprocessing audio data, focusing on two main tasks: extracting log-mel spectrograms and transferring pretrained CNN14 model parameters.

### 2.4.1 Log-Mel Spectrogram Extraction

To effectively preprocess audio data, it is first converted into log-mel spectrograms. A log-mel spectrogram is a powerful representation of the audio signal, capturing both time and frequency information in a compact form. The process involves several key steps:

- **Loading Audio Files**: The audio data is organized into different splits, such as development, validation, and evaluation sets. Each audio file is identified using a unique identifier that maps to its filename. This information is stored in a pickle file, which is loaded at the beginning of the process.
- **Setting Parameters**: Several parameters are defined to control the spectrogram extraction process. These include the sample rate, window length, hop length, and the number of mel bands. The sample rate determines how many samples per second are taken from the audio signal. The window length specifies the duration of each segment of the audio signal to be analyzed, while the hop length determines the interval between successive segments. The number of mel bands defines the resolution of the frequency analysis.
- **Generating Mel Spectrograms**: The raw audio waveform is transformed into a mel spectrogram using the Short-Time Fourier Transform (STFT). This step involves segmenting the audio signal into overlapping windows, computing the Fourier transform for each window, and mapping the resulting frequency bins to the mel scale, which is designed to mimic the human ear's perception of sound.

- **Applying Logarithmic Transformation**: To enhance the dynamic range of the spectrogram and make it more suitable for neural networks, a logarithmic transformation is applied. This converts the linear mel spectrogram into a log-mel spectrogram, which emphasizes the lower amplitude components and compresses the higher amplitude ones.
- **Saving Spectrograms**: The resulting log-mel spectrograms are stored in HDF5 files, organized by the dataset splits. Each spectrogram is associated with its corresponding audio file identifier, ensuring that the data is well-structured and easily accessible for subsequent processing.

### 2.4.2 Transferring Pretrained CNN14 Model Parameters

The next step in preprocessing audio data involves utilizing a pretrained CNN14 model to enhance feature extraction. This process transfers the knowledge gained from a large-scale training on diverse audio data to a custom neural network encoder. The steps are as follows:

- **Mapping Parameters**: A mapping is created to associate the parameter names of the custom encoder with those of the pretrained CNN14 model. This ensures that the parameters are correctly transferred to the corresponding layers in the custom encoder.
- **Loading Pretrained Parameters**: The pretrained parameters of the CNN14 model, which have been trained on a large audio dataset, are loaded from a specified path. These parameters contain valuable information that can significantly improve the performance of the custom encoder.
- **Initializing Custom Encoder**: A custom CNN14 encoder is initialized with a specified output dimension. This encoder is designed to process the log-mel spectrograms and extract relevant audio features.
- **Transferring Parameters**: The pretrained parameters are transferred to the custom encoder by copying the state dictionary and mapping the keys accordingly. This step ensures that the custom encoder inherits the learned features and patterns from the pretrained model.
- **Saving Custom Encoder**: The state dictionary of the custom encoder, now containing the transferred parameters, is saved to a specified path. This enables the custom encoder to be reused for various audio processing tasks, leveraging the benefits of the pretrained CNN14 model.

By following these steps, the audio data is effectively transformed and prepared for neural network analysis, ensuring that the models can efficiently learn and extract meaningful patterns from the audio signals.

## 2.5 Post processing:

The post-processing stage in this study is pivotal for evaluating and interpreting the performance of the cross-modal retrieval system. This phase involves calculating the similarity scores between audio and text data and subsequently measuring the effectiveness of these scores in retrieving relevant information. The post-processing tasks are primarily focused on two critical operations: computing cross-modal similarity scores and evaluating retrieval performance metrics.

### 2.5.1 cosine measure:

The cosine similarity measure is a fundamental component of the post-processing pipeline, employed to quantify the similarity between audio and text embeddings. In our approach, we utilize cosine similarity to evaluate the relevance of audio-text pairs in the cross-modal retrieval system.

**Computation of Cross-Modal Scores:**

The cosine similarity measure is used to compute the cross-modal scores between audio and text features. This process involves encoding both modalities—audio and text—using their respective neural network branches. The encoded features are then compared using cosine similarity to determine their relative similarity. Specifically, each audio feature vector is compared against all text feature vectors using cosine similarity, resulting in a score that reflects the degree of correspondence between the audio and text data.

The procedure involves the following steps:

- **Text Encoding**: Text data is encoded into feature vectors through a text branch of the neural network. These feature vectors represent the semantic content of the text.
- **Audio Encoding**: Similarly, audio data is encoded into feature vectors via an audio branch. These vectors encapsulate the auditory information from the audio inputs.
- **Similarity Calculation**: Cosine similarity is computed between the audio and text feature vectors. This metric quantifies how closely the audio and text features align, with higher scores indicating greater similarity.

**2. Retrieval Metrics Evaluation**

Post similarity computation, retrieval metrics are used to assess the performance of the system. The evaluation focuses on two primary aspects: the accuracy of retrieval results and the effectiveness of the similarity scores.

- **Retrieval Accuracy**: Metrics such as recall and mean average precision (mAP) are computed to evaluate how well the system retrieves relevant items. Recall metrics assess the proportion of relevant items retrieved at various cutoffs (e.g., top-1, top-5, top-10), while mAP provides a measure of the average precision across multiple queries.
- **Performance Analysis**: The effectiveness of the retrieval system is analyzed by comparing the retrieved results against the ground truth. This involves measuring how well the system retrieves relevant items and the ranking quality of these items.

The results from these metrics are essential for understanding the performance of the cross-modal retrieval system. They provide insights into the accuracy of the retrieval process and the effectiveness of the similarity measure in distinguishing between relevant and non-relevant items.

In summary, the post-processing phase, specifically through the cosine similarity measure, plays a crucial role in evaluating and fine-tuning the cross-modal retrieval system. By calculating similarity scores and assessing retrieval metrics, we gain valuable insights into the system's performance and its ability to effectively match audio and text data.

## 2.6 Conclusion

This chapter has presented the proposed approach for sound retrieval with human-written captions. The primary objective was to elucidate each component of the architecture employed. The chapter discussed various elements and techniques involved in achieving this goal. These included text features, the storage of audio in the database, the application of models such as CNN and SBERT, and the significance of the loss function. By examining each of these aspects, we have established a solid foundation for the subsequent chapters, which will delve deeper into the implementation and evaluation of the system.

# Chapter 3: Achievement

## 3.1 Introduction

This chapter provides a comprehensive overview of the experimental setup utilized in our investigation of audio retrieval using human-written captions. In the previous chapter, we outlined our approach to this task. Here, we delve into the tools employed in the development of our audio retrieval system, which encompasses a diverse range of technologies. Additionally, we offer a detailed description of our dataset, including the sources and characteristics of the audio and captions used in our experiments. Furthermore, we describe the procedure followed to analyze and discuss the results of our experiments, which involved evaluating the performance of our system using various metrics such as precision and recall. Overall, this chapter serves as a thorough guide to the development and evaluation of our audio retrieval system, highlighting the effectiveness of using human-written captions to match relevant audio content.

## 3.2 Dataset presentation

The Clotho dataset is specifically designed for the task of audio captioning, which involves generating textual descriptions of audio signals. Unlike speech-to-text tasks, audio captioning focuses on describing general audio content, including sound events, acoustic scenes, and environmental sounds. The dataset consists of 4,981 audio samples, each ranging from 15 to 30 seconds in duration, accompanied by a total of 24,905 captions.

The dataset was created by collecting audio samples from the Freesound platform. Captions were crowdsourced through Amazon Mechanical Turk, ensuring diversity in descriptions. Each audio sample is annotated with five captions, each consisting of 8 to 20 words. This process helps capture a range of descriptions for each audio signal, enhancing the dataset's overall diversity.

To ensure the quality and relevance of the captions, unique words and named entities were removed during post-processing. Additionally, speech transcriptions were excluded to maintain focus on general audio content.

Clotho was split into three distinct sets for training, development, and evaluation. The development and evaluation splits are publicly available, while the testing split is withheld for potential scientific challenges. This careful splitting ensures that each word appears in either the development split or one of the other two splits, which facilitates a balanced learning and evaluation process.

For detailed information and access to the Clotho dataset, refer to the provided Link[1].

## 3.3 Used Tools

Deep learning research often requires large datasets and intensive computational operations, highlighting the need for parallel computing to accelerate model training. Although GPUs are a common choice for this purpose, their high acquisition and maintenance costs can lead to issues such as equipment depreciation and excessive use. To address these concerns, we opted for cost-effective alternatives like COLAB and Kaggle. COLAB, a browser-based platform, and Kaggle, a renowned data science platform, enabled us to perform resource-intensive calculations and run Python [24] code efficiently. These economical alternatives to GPUs allowed us to speed up the training process while managing costs effectively. Additionally, we utilized Google Drive for secure storage and management of our datasets, model checkpoints, and experimental results, facilitating access and sharing within our research team while providing a reliable backup and synchronization solution.

For development, we employed Jupyter, Conda, and PyCharm [25]. Jupyter served as an interactive development environment for experimentation and prototyping. Conda helped manage packages and create reproducible environments, ensuring consistency in our research. PyCharm provided a robust integrated development environment (IDE) for coding, debugging, and project management, enhancing our development workflow.

Regarding libraries, we utilized a range of specialized tools. Librosa provided comprehensive audio analysis capabilities, while SBERT (Sentence-BERT) offered advanced models for sentence embeddings. Additionally, Tkinter played a key role in developing graphical user interfaces (GUIs) for our interactive applications. Finally, we leveraged the combined capabilities of Keras and TensorFlow, leading deep learning frameworks, to effectively implement and experiment with various deep learning models.

[1] https://zenodo.org/records/4783391

## 3.4 Dataset Description:

The Clotho v2 dataset [10] is a valuable resource for audio analysis and retrieval. It comprises 6,974 audio samples, each accompanied by five human-written captions, totaling 34,870 captions. The audio clips range from 15 to 30 seconds in length, and the captions consist of 8 to 20 words. This dataset is instrumental in advancing and evaluating audio retrieval algorithms.

The dataset is sourced from the Freesound platform [26], and the captions are generated through Amazon Mechanical Turk. It is divided into three subsets: Development, Validation, and Evaluation. The dataset also includes metadata stored in CSV files, which provide information such as file names, keywords, URLs, and uploader/user details.

Managing the size and resources of this dataset is crucial due to the large number of captions and audio samples. Nevertheless, the well-organized structure of the Clotho v2 dataset [10] facilitates access and analysis, making it an invaluable resource for researchers and developers in the field of audio analysis and retrieval.

## 3.5 Evaluation Metrics:

### 3.5.1 Precision and Recall of a Binary/Non-Binary Classifier:

For binary classification problems (where there are only two classes), precision and recall can be calculated using the following formulas:

- **Precision** = TP / (TP + FP)
- **Recall** = TP / (TP + FN)

Where TP stands for True Positives, FP for False Positives, and FN for False Negatives.

For non-binary classifiers, precision and recall can be calculated individually for each class using the same formulas.

### 3.5.2 Precision and Recall at Threshold k:

Precision and recall at threshold k are evaluation metrics that consider the top k predictions made by a model. These metrics calculate the ratio of true positives (TP) to the sum of true positives and false positives (FP) for precision, and the ratio of true positives (TP) to the sum of true positives and false negatives (FN) for recall. This approach is useful when evaluating the model's performance at a specific threshold or cutoff.

### 3.5.3 Mean Average Precision (MAP):

Mean Average Precision (MAP) is a commonly used evaluation metric in information retrieval and recommendation systems. This metric provides a comprehensive assessment of a model's performance across different recall levels. It calculates the average precision at each recall level and then computes the overall average of these precision values.

MAP is extensively used to evaluate the performance of cross-modal retrieval algorithms [27].

### 3.5.4 Mean Average Precision at K (mAP@K):

Mean Average Precision at K (mAP@K), also known as "Mean Average Precision at 10" when K=10, is a variant of MAP that focuses on precision and recall at a specific threshold of 10. It measures the average precision of the top 10 predictions made by a model, providing insight into the model's performance on the most relevant predictions.

## 3.6 Tests

### 3.6.1 Architecture of SBERT:

The training loss values (trained for 300 epochs using the triplet loss function) are as follows:

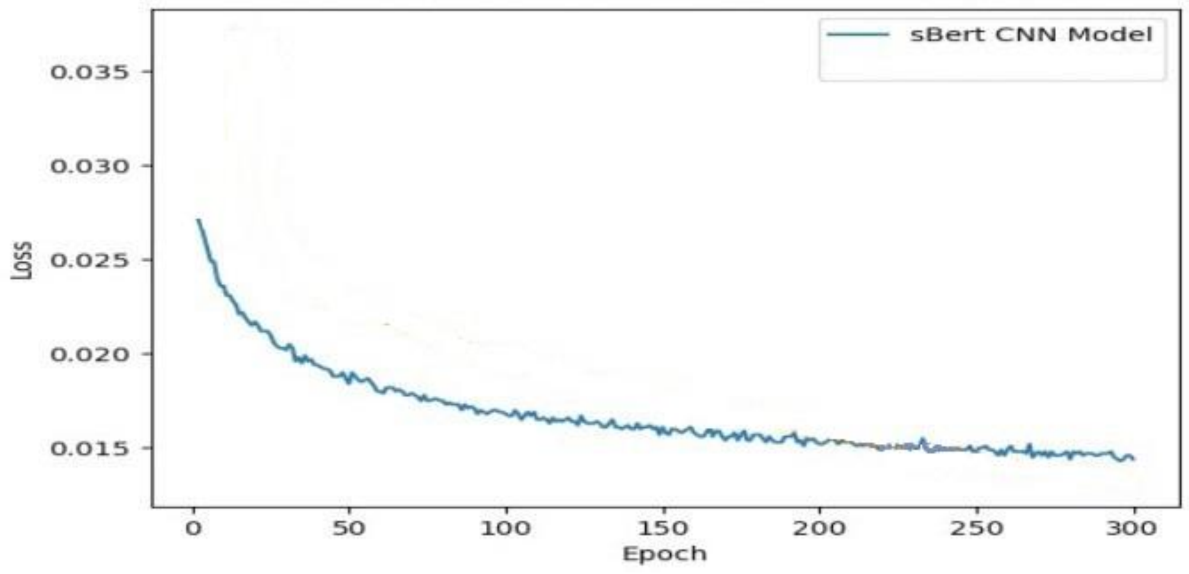

***Figure 2: les valeurs de perte pour le modèle sBert CNN14***

We calculated the 10 closest audio samples for each value in our evaluation dataset using the dot product. The results of the evaluation are as follows:

| **Metric** | **Value** |
|---|---|
| R1 | 0.075 |
| R5 | 0.179 |
| R10 | 0.251 |
| mAP10 | 0.072 |

***Figure 3: Results of the sBert CNN14 Architecture***

## 3.7 Discussion sur les résultats:

In this section, we present the results obtained from the architecture used in our study. The performance metrics, including R1, R5, R10, and mAP10, were evaluated for the sBert architecture combined with CNN models using log mel spectrograms.

### 3.7.1 sBert and CNN Performance:

Our analysis demonstrated that the sBert and CNN model achieved significant performance across all evaluated metrics. The model delivered robust results for R1, R5, R10, and mAP10, showcasing its effectiveness in capturing audio patterns and performing well in recognition tasks. These results highlight the effectiveness of combining sBert with CNNs for audio signal processing and classification.

#### 3.7.2 Method Evaluation:

An important finding from our study is that the approach of extracting features from log mel spectrograms yielded superior performance compared to other methods involving direct predictions. This emphasizes the importance of feature extraction in enhancing the overall performance of the sBert and CNN architecture.

#### 3.7.3 Overall Performance:

The sBert and CNN architecture demonstrated impressive performance in our experiments, achieving high recognition accuracy and effective retrieval results. This indicates that this architecture is well-suited for the audio processing tasks considered in our study.

## 3.8 Conclusion

In this chapter, we outlined our approach to audio retrieval using human-generated subtitles. We began by detailing the tools and programming languages employed in our work, followed by a description of the dataset used for our experiments. We assessed our method using various parameters and presented the results along with an analysis. Our approach yielded promising outcomes, highlighting the potential of utilizing human-written subtitles for audio retrieval.

# Conclusion

In our proposed approach for audio retrieval using human-written subtitles, we provided a comprehensive overview of the techniques employed for audio and text processing, highlighting relevant related work in the field. We aimed to leverage advanced text processing techniques such as SBERT and utilized the Clotho dataset to enhance retrieval performance. Additionally, we employed various neural network architectures, specifically CNNs, to develop a model capable of extracting features from both textual and audio inputs.

Our experiments assessed the effectiveness of our approach using different configurations that combined SBERT with CNN to derive log mel spectrogram features. Despite encountering challenges related to hardware limitations, we achieved promising results that underscored the potential of using human-written subtitles to enhance audio retrieval performance.

We also detailed the tools utilized, including the Clotho dataset and evaluation metrics. Future research directions include exploring different neural network architectures, focusing on real-time automated audio subtitling, improving hardware compatibility, and extending the approach to new platforms. Additionally, integrating Pre-trained Audio Neural Networks (PANNs) could offer a promising avenue for future work, as they leverage transfer learning to capture valuable audio features from large-scale datasets, potentially enhancing the model's ability to recognize complex audio patterns and improving performance across various applications.

References:

[1] K. Koutini, J. Schluter, H. Eghbal-zadeh et a. G. Widmer, «Efficient training of audio transformers with patchout,» *in 23rd Annual Conf. of the Int. Speech Communication Association, Interspeech,* 2022.

[2] T. Mikolov, K. Chen, G. Corrado et a. J. Dean, «Efficient estimation of word representations in Vector Space,» *arXiv:1301.3781,* 2013.

[3] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang et a. M. D. Plumbley, «PANNs: Large-scale pretrained audio neural networks for audio pattern recognition,» *IEEE/ACM Trans. Audio Speech Lang. Process., pp. 2880–2894,* 2020.

[4] R. N et G. I, «Sentence-bert: Sentence embeddings using siamese bert-networks,» *arXiv preprint arXiv:1908.10084,* 2019 Aug 27.

[5] M. S. FARHAD, P. THINAGARAN, M. NORWATI et M. RAIHANI, «A Comprehensive Overview and Comparative Analysis on Deep Learning Models: CNN, RNN, LSTM, GRU,» *University Putra Malaysia (UPM), Malaysia, arXiv:2305.17473,* 2023.

[6] J. P. Bharadiya, «A Comprehensive Survey of Deep Learning Techniques Natural,» *European Journal of Technology, Doctor of Philosophy Information Technology, University of the Cumberlands, USA,* 2023.

[7] P. Paul, K. Khaled et W. Gerhard, «ADVANCING NATURAL-LANGUAGE BASED AUDIO RETRIEVAL,» *Institute of Computational Perception (CP-JKU), LIT Artificial Intelligence Lab, Johannes Kepler University, Austria,* 2023.

[8] J. Devlin, M. Chang, K. Lee et a. K. Toutanova, «BERT: pretraining of deep bidirectional transformers for language understanding,» *in Proc. of the North American Ch. of the Ass. for Computational Linguistics: Human Language Technologies, NAACL-HLT,* 2019.

[9] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer et a. V. Stoyanov, «Roberta: A robustly

optimized BERT pretraining approach,» *CoRR, vol.abs/1907.11692,* 2019.

[10] X. Mei, C. Meng, H. Liu, Q. Kong, T. Ko, C. Zhao, M. D.Plumbley, Y. Zou et a. W. Wang, «Wavcaps: A chatgptassisted weakly-labelled audio captioning dataset for audiolanguage multimodal research,» *CoRR, vol. abs/2303.17395,*, 2023.

[11] K. Drossos, S. Lipping et a. T. Virtanen, «Clotho: An audio captioning dataset,» *in Proc. ICASSP*, 2020.

[12] C. D. Kim, B. Kim, H. Lee et a. G. Kim, «AudioCaps: Generating captions for audios in the wild,» *in Proc. of the North American Ch. of the Ass. for Computational Linguistics: Human Language Technologies, NAACL-HLT*, 2019.

[13] F. Ziye et Z. Fengyun, «QFORMER BASED TEXT AUDIO RETRIEVAL SYSTEM,» *R&D, Lingban Technology Ltd,., Beijing, China* , 2023.

[14] A.-M. Oncescu, A. S. Koepke, J. F. Henriques, Z. Akata et a. S. Albanie, «Audio Retrieval with Natural Language Queries,» July 2021.

[15] S. Chen, Y. Wu, C. Wang, S. Liu, D. Tompkins, Z. Chen et a. F. Wei, «BEATs: Audio Pre-Training with Acoustic Tokenizers,» Dec. 2022.

[16] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger et a. I. Sutskever, «Learning Transferable Visual Models From Natural Language Supervision,» *https://arxiv.org/abs/2103.00020v1,* Feb. 2021.

[17] J. Li, D. Li, S. Savarese et a. S. Hoi, «BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models,» Jan. 2023.

[18] A. Guzhov, F. Raue, J. Hees et a. A. Dengel, «AudioCLIP: Extending CLIP to Image, Text and Audio,» *https://arxiv.org/abs/2106.13043v1,* June 2021.

[19] H.-H. Wu, P. Seetharaman, K. Kumar et a. J. P. Bello, «Wav2CLIP: Learning Robust Audio Representations From CLIP,» *https://arxiv.org/abs/2110.11499v2,* Oct. 2021.

[20] B. Elizalde, S. Deshmukh, M. A. Ismail et a. H. Wang, «CLAP: Learning Audio Concepts From Natural Language Supervision,» *https://arxiv.org/abs/2206.04769v1,* June 2022.

[21] J. Li, D. Li, S. Savarese et a. S. Hoi, «BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models,» Jan. 2023.

[22] L. Etienne, P. Thomas et P. Julien, «IRIT-UPS DCASE 2023 AUDIO CAPTIONING AND RETRIEVAL SYSTEM,» *IRIT (UMR 5505), Universite Paul Sabatier, CNRS, Toulouse, France, Artificial and Natural Intelligence Toulouse Institute (ANITI)* , 2013.

[23] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell et a. S. Xie, «A convnet for the 2020s," in 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),» p. pp. 11 966–11 976., 2022.

[24] Python et ". "Python, «https://www.python.org/downloads/release/python-394/,» Apr. 04, 2021.

[25] F. d. PyCharm, «https://www.jetbrains.com/help/pycharm/quick-start-guide.html».

[26] F. Font, G. Roma et a. X. Serra, «Freesound Technical Demo,» *in Proceedings of the 21st ACM International Conference on Multimedia, ser. MM '13. New York, NY, USA: Association for Computing Machinery,* p. pp. 411–412, 2013.

[27] P. Kaur, H. S. Pannu et a. A. K. Malhi, «Comparative analysis on crossmodal information retrieval: A review,» *Comput. Sci. Rev., p. 100336,* 2021.