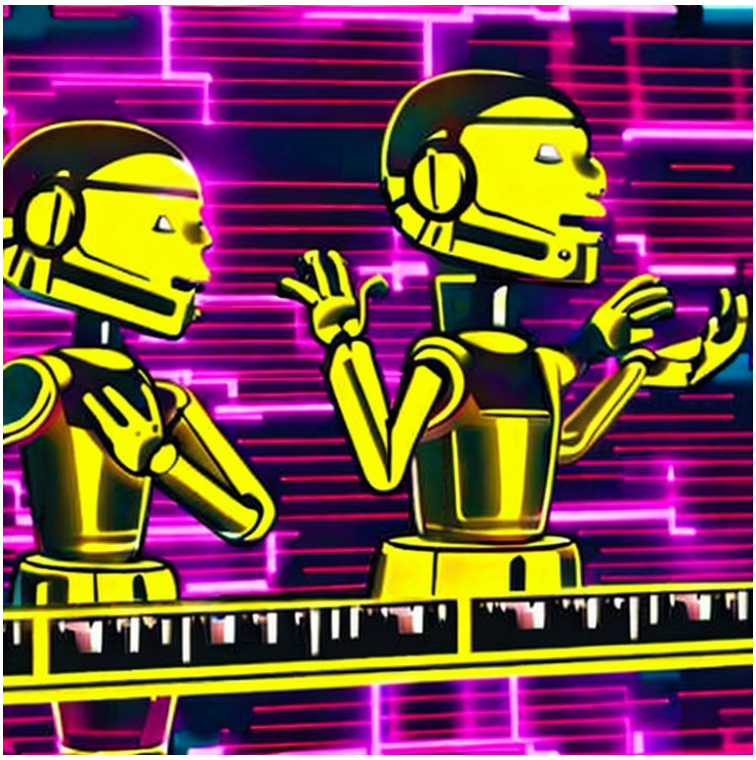

Multi-task French speech analysis with deep learning

Emotion recognition and speaker diarization models for end-to-end conversational analysis tool

Master's thesis in Complex Adaptive Systems

JULES SINTES

DEPARTMENT OF PHYSICS

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2023
www.chalmers.se

Master's thesis 2023

Multi-task French speech analysis with deep learning

Emotion recognition and speaker diarization models for end-to-end conversational analysis tool

JULES SINTES

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Physics
Chalmers University of Technology
Gothenburg, Sweden 2023

Multi-task French speech analysis with deep learning
Emotion recognition and speaker diarization models for end-to-end conversational analysis tool
JULES SINTES

Supervisor: Nhut DOAN NGUYEN, La Javaness
Examiner: Mohsen MIRKHALAF, Department of Physics

Master's Thesis 2023
Department of Physics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Generated with Stable Diffusion 2.1, prompt *"synthwave cartoon style audio waveform with two humanoid robots talking"*

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2023

Multi-task French speech analysis with deep learning
Emotion recognition and speaker diarization models for end-to-end conversational analysis tool
JULES SINTES
Department of Physics
Chalmers University of Technology

Abstract

Automatic Speech Recognition has become a key application of deep learning and neural networks. Thanks to the development of new model architectures such as transformers, audio processing tasks such as speech-to-text, audio classification, or audio segmentation technologies are now a crucial part of human-computer interaction systems and widely used in commercial products. In addition, while models are becoming more accurate and robust, an interest in emotion recognition systems is growing to assist operators in their interaction with customers (or patients in the context of healthcare). This thesis aims at improving the previous proof of concept and develop speech emotion recognition and speaker diarization models for real-life data.

Firstly, for speech emotion recognition task, we create a new conversational dataset in French language based on real-life recordings from TV documentaries. It contains a large plurality of speakers in various contexts, expressing a wide diversity of emotions. We conduct a comparative study of various approaches and models with our dataset and achieve state-of-the-art performance, beating pre-trained English-based benchmark models on real-life data while still achieving acceptable results on the RAVDESS benchmark dataset.

Next, speaker diarization relates to answering the question "Who spoke when?" We conduct an in-depth comparative study of major open-source frameworks on chosen test cases, with an emphasis on optimizing accuracy along with inference time and hardware requirements.

Finally, we implement the emotion recognition and speaker diarization models in an end-to-end conversational analysis tool, which generates a diarized text transcription of the conversational content, along with intensity and emotion recognition on a segment level for both text and audio. The tool also includes a zero-shot topic detection feature, which can be easily extended with various other NLP tasks. The web application can be used as a demonstration tool for business cases and showcases the scalability and flexibility of the proposed approach.

Keywords: deep learning, automatic speech recognition, speech emotion recognition, speaker diarization.

Acknowledgements

I would like to thank Nhut Doan Nguyen my supervisor at La Javaness for all the support and guidance through this project. I also would like to thank all the Unstructured Data Team and all the people from the Data Lab at La Javaness for their help, feedback and advice during this thesis project. Finally I would like to thank Mohsen Mirkhalaf for acting as the examiner of this thesis.

Jules Sintes, Paris, June 2023

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AI	Artificial Intelligence
ASR	Automatic Speech Recognition
DL	Deep Learning
FFT	Fast Fourier Transform
GDPR	General Data Protection Regulation
GPU	Graphics Processing Units
IT	Information Technology
LSTM	Long short-term memory
ML	Machine Learning
NER	Named Entity Recognition
NLP	Natural Language Processing
NN	Neural Network
R&D	Research and Development
RNN	Recurrent Neural Networks
SER	Speech Emotion Recognition
SOTA	State-Of-The-Art
STT	Speech to text
TPU	Tensor Processing Units
VAD	Voice Activity Detection

Contents

List of Figures

List of Tables

1
Introduction

Artificial Intelligence is currently one of the most active fields of research. Especially, deep learning is widely used to solve a large variety of tasks, and companies are actively seeking to take advantage of these new powerful tools [1]. The quantity of data collected has exploded in the last decade, and the robustness and accuracy of neural network models have incredibly improved in the last few years thanks to academic research but also to tech companies. The fast improvement in the performance of deep learning models have accelerate the development of applications for audio and speech processing. The specific field of speech processing and recognition is an old and well known problem : the first attempt to automatically retrieve language information from speech recordings wad made in 1952 in the Bell Labs to recognize spoken digits [2]. Nowadays, automatic speech processing includes a large variety of tasks aiming at extracting meaningful information from speech recordings. These tasks can be considered as part of the larger field of Natural Language Processing (NLP). Training models to automatically retrieve information from audio requires specific architectures such as transformers [3] with a large number of parameters (up to several billions) [4]. Thanks to recent advances, the development of unsupervised learning allowed to significantly increase the quantity of training data available to develop efficient language model for audio processing [5]. As an example, the current state of the art model for automatic speech recognition, whisper, has been trained on 680 000 hours of audio content [6]. In addition, the use of transfer learning and fine tuning to leverage pre-training of large audio embedding models such as wav2vec2 [7] allowed major advances in the field of audio classification as well as automatic speech recognition.

While speech recognition models allow to extract information from audio input, speaker diarization is critical when processing long conversational recordings. Speaker diarization relates to answering the question : "who spoke when ?". It allows to segment audio input and cluster segments according to speakers [8] which constitute a major challenge for information retrieval in conversations.

This thesis project, within the start-up La Javaness in Paris, focuses on state-of-the-art deep learning techniques for Automatic Speech Recognition (ASR), Speech Emotion Recognition (SER), and Speaker Diarization. The main scope of this project is to develop a multi-task tool that performs various speech analysis tasks for the French language in real-life contexts. Such a tool could be use for various application : call center analytics [9, 10], media monitoring [11], language learning [12], etc.

1.1 Context and scope of the project

As part of Research & Development team at La Javaness[1], a French start-up based in Paris, the project follows an initial Proof-of-concept developed in 2022 for an end-to-end speech processing application. The project was initially launched by the company to develop expertise and increase knowledge in the field of deep learning for audio processing. The initial work includes a complete pipeline implementing automatic speech recognition (ASR) along with speech emotion recognition (SER) and various natural language processing tasks (NLP) such as topic detection or text tonality recognition.

While the proof-of-concept seems very promising and works great on research benchmark datasets such as the Canadian French Emotional Speech Dataset [13], results are not yet good enough to develop a working tool for real-life applications for the French language. In addition, new machine learning models for audio processing have been released recently, such as Whisper from OpenAI (September 2022) [6]. Thus, this thesis project aims at improving previous results on emotion classification tasks, especially making the SER model more accurate for real conversations and developing an efficient speaker diarization for the audio processing pipeline.

1.2 Planning and Progress

In the context of the R&D on Unstructured Data team, this project deeply emphasized on following IT/ML development best practices, and on exploring new methods and approaches for data-driven and deep learning based products. The project was limited to 20 weeks and was divided into 3 distinct parts:

1. **Speech Emotion Recognition (SER)** - Construction of the dataset, exploration of data, feature engineering, improvement of previous models, design of new approaches. This includes developing precise framework and tools for audio dataset generation and labeling.
2. **Speaker Diarization** - State-of-the-art (SOTA) model/framework evaluation and comparison, hyperparameters optimization, integration in the speech analysis pipeline.
3. **Implementation of complete pipeline** - Improving demonstration application for multitask speech analysis tool by implementing SER model and speaker diarization pipeline within a web application for demonstration purpose.

1.3 Ethical Considerations

Dealing with voice data and speech processing models require some ethical considerations regarding the whole development and potential deployment process. Recent

[1] La Javaness helps European companies and public organizations to develop and deploy machine learning-based tools in the context of the so-called digital transition.

advances within the connected fields of machine learning, big data and generally artificial intelligence raised several ethical and environmental issues [14]. Especially, regarding this project, one should be careful with :

1. **Voice recordings** - Conversational audio can be considered personal data regarding the General Data Protection Regulation (GDPR)[2] - the European regulation on data protection and privacy. Indeed, the content of the conversation as well as the voice itself can identify individuals and can contain personal information. Thus, collecting, processing, and using such audio data should be done carefully.
2. **Models** - Deep learning models are often called "black boxes" as they are meant to approximate functions without providing insights on the actual form of the targeted function. Neural networks architecture relies on complex non-linear algebra and most recent models for Computer Vision (CV) and NLP can have several billion parameters. Therefore, it makes them less directly explainable, which can be an issue in many applications, especially when considering security or health issues (e.g., in autonomous vehicles, conversational agents, etc). Thus, one needs to be able to control or at least evaluate precisely the quality of the model. Handle and mitigate errors is critical, and one should try to maximize explainability. More specifically, in this project, SER is a highly subjective task, and thus, there is a high risk of including bias in the training data, and the quality of such a model can be quite difficult to evaluate.
3. **Model deployment** - When deploying AI-based applications, one should also consider the use of the model and its potential consequences. Here, the work focuses on ASR, and one major potential application of an ASR multi-task tool could be, as mentioned above, in a call center. Considering this particular domain of application, two major ethical issues can be identified: On the one hand, it could help the operator in their work with a better comprehension of the customer needs using an analytical tool based on speech recognition. However, it is not meant to replace the emotional intelligence of the operator, and the tool should be used wisely as it is only a support tool for the operator. On the other hand, such a tool could be used to actually assess and control the work of an operator. The use for worker surveillance should be considered very carefully. Such a deployed application should be able to evaluate and quantify possible errors in the predicted analytics.
4. **Ecological impact** - Finally, the ecological impact of AI needs to be considered when working with very large machine learning models. The process of storing, processing huge amounts of data, as well as training and inference with models, can have a massive carbon footprint. Indeed, most of the data and models are stored remotely on servers with GPUs, which need a lot of energy to work properly, and the resulting carbon footprint and ecological impact cannot be ignored. Thus, tech companies have a significant responsibility when they develop, manage, and deploy AI-based tools to make them as frugal as possible to limit ecological impact [15].

[2] https://gdpr-info.eu/

1.4 Contribution

This thesis contributes to the development of AI-based voice processing with a focus on developing deep learning models for real life conversation recordings and implementing them in a complete machine learning based speech processing pipeline. The final outcome is an end-to-end pipeline able to process and extract meaningful information from long conversation recordings. The detailed contributions on speech emotion recognition, speaker diarization and multi-task speech processing are fully described in the following subsections.

1.4.1 Speech Emotion Recognition

During this project, a new dataset based on real-life French conversational content was built to provide a more versatile dataset, allowing for a real data approach for the training of more robust models. While advanced deep learning models for SER usually achieve accuracy scores up to 92% on research benchmark sets [16], they actually have poor performance on real-life data. Moreover, most of the datasets available contain only english data. Hence, this project tackle the lack of robust models for recognition of non-stereotyped emotion expression for the French language.

Indeed, a multilabel classifier model was trained on the new dataset. This model adopts a new approach with a reduced number of emotions, which leads to improve the previous results from the first proof-of-concept and gives higher scores on real-life conversational test sets than publicly available SER models. In addition, the trained model still achieves good scores on English benchmark research sets.

Furthermore, this project also explored data preprocessing and augmentation, as well as different promising model architectures, such as multitask, multimodal and quantitative approaches. This constitutes an interesting baseline for further work to improve SER models and highlights the possibility to develop robust models for emotion recognition in real-life contexts.

1.4.2 Speaker Diarization

Speaker Diarization is a key task when developing conversation analysis tools. This project tries to leverage pre-existing frameworks and conducts an in-depth study of the performance and behavior of speaker diarization frameworks. Especially, a diarization pipeline is optimized for different test cases considering hardware and time constraints. State of the art pipelines for speaker diarization such as pyannote achieves, on benchmark sets, diarization error rates ranging from 8.2% (REPER dataset) to 32.4% (CALLHOME dataset) [17, 18]. Through a deep understanding of speaker diarization frameworks and pipeline, the final optimized diarizer matches expected behavior and achieves a diarization error rate of 11% for our defined test cases. This makes the optimized pipeline suitable for conversation transcription with automatic speech recognition models.

1.4.3 Speech Analysis Tool

Finally, an improved proof of concept for multitask speech analysis tools is proposed as a web application able to predict from a single audio file: a clear segmentation of speakers, a text transcription (or alternatively a translation to another language), emotion recognition (speech and text) and may include various other natural language processing (NLP) tasks.

2
Theory

This chapter aims at introducing and defining all the general concepts studied during this project. It provides some context and basic scientific background of the main domains explored throughout the thesis. A background in computer science and mathematics is assumed as well as basic knowledge of machine learning.

2.1 Audio data and signal processing

2.1.1 Sound and Audio signal

Sound is a form of energy that travels through a medium as a series of pressure waves, which our ears detect and interpret as sound. The properties of sound waves, such as their frequency and amplitude, determine the characteristics of the sound we hear, such as its pitch and volume. The numerical representation of sound is an audio signal represented as a time series as in Figure 2.1. Usually, analog signals use voltage levels, while digital signals use a series of binary numbers. The frequency range of audio signals matches the lower and upper limits of human hearing, generally going from roughly 20 to 20000 Hz.

The sample rate is an essential parameter of the signal, defining its resolution and, to some extent, the audio quality. In this project, all processed audio signals are generally sampled at a rate of $f_s = 16000$ Hz, meaning that an audio of length $1s$ will be a vector of size $(1, 16000)$. According to the Nyquist-Shannon sampling theorem [19], with a sampling rate of 16000 Hz, the maximum frequency that can be represented in the audio signal is $f_{max} = \frac{f_s}{2} = 8000$ Hz.

Such a frequency range, up to 8000 Hz, is convenient for processing human voices and has become the standard for Deep Learning-based applications with speeches. As a reference, audiovisual content (music, TV, etc.) is usually sampled and recorded at a sample rate of $f_s = 44100$ Hz or $f_s = 48000$ Hz to keep the whole frequency range audible to human ears.

2.1.2 FFT and Mel Spectrogram

When using audio data with deep learning algorithms, usually, the audio signal is not used directly as input data and first needs to be processed to be used. Especially, when doing voice processing, it is much easier to process the frequency content than

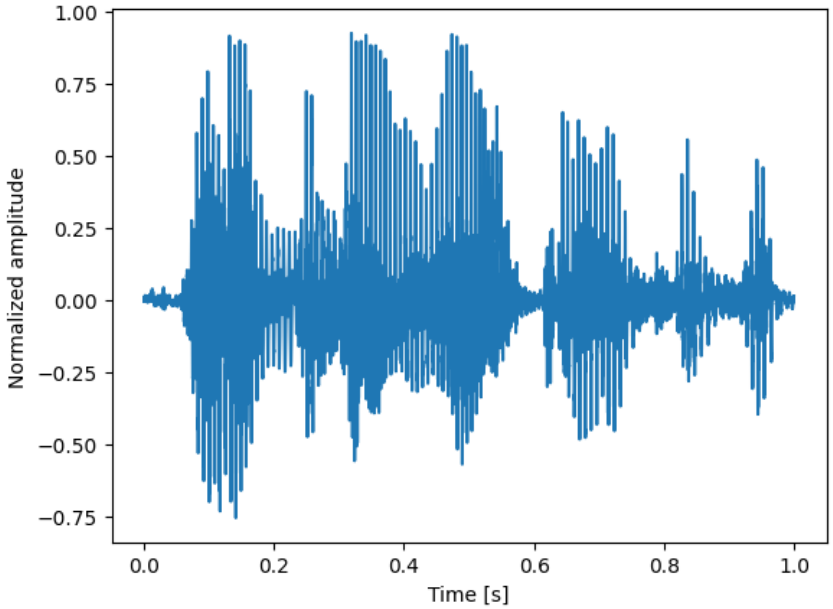

Figure 2.1: Audio signal example.

the raw time series from audio signal as highlighted in a recent review on feature extraction technique by Labid and Belangour [20].

The Fast Fourier Transform (FFT) is an algorithm that computes the Discrete Fourier Transform (DFT) of a signal, which is a mathematical technique that decomposes a time-domain signal into its constituent frequency components. It transforms a sequence of complex numbers x_n into another discrete sequence of complex numbers X_k according to equation 2.1.

$$X_k = \sum_{n=0}^{N-1} x_n e^{-2i\pi kn/N}, \quad k = 0, 1, ..., N-1\,. \tag{2.1}$$

Here, X_k is the k^{th} element of the output sequence (the Fourier transform), x_n is the n^{th} element of the input sequence. $e^{-2i\pi kn/N}$ represents a primitive N^{th} root of unity.

In simpler terms, FFT is a method for analyzing a signal in terms of its frequency content. In deep learning models for audio, FFT is used as a preprocessing step to transform audio signals from the time domain to the frequency domain. This transformation allows the model to analyze the audio data in terms of its spectral content. Especially, the input audio signal is typically first divided into small segments called "frames," which are typically 10-30 milliseconds in duration. FFT is then applied to each frame to compute a "spectrogram," which is a 2D representation of the signal's frequency content over time. The spectrogram is then used as input to the deep learning model.

Especially, the most common way to preprocess audio for speech analysis tasks with deep learning is to generate a Mel Spectrogram of the audio and then treat this input spectrogram as an image [20]. The Mel scale, as shown in Figure 2.2 is a non-linear

transformation of frequency that more closely matches the way humans perceive pitch. Therefore, it provides a more accurate representation of the frequency content of a signal as it relates to human perception. This type of pre-processing is used by models such as wav2vec2 [7] and whisper [6].

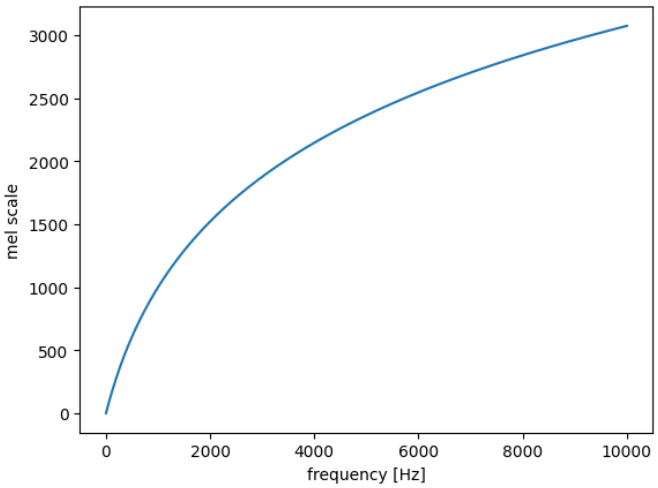

Figure 2.2: Mel scale versus frequency in Hz.

To create a Mel Spectrogram (Figure 2.3), the signal is first broken down into short time intervals, and the frequency content of each interval is determined using the FFT as described above. The resulting spectrum is then mapped onto the Mel scale, and the amplitude of each frequency bin is represented by a color or grayscale value.

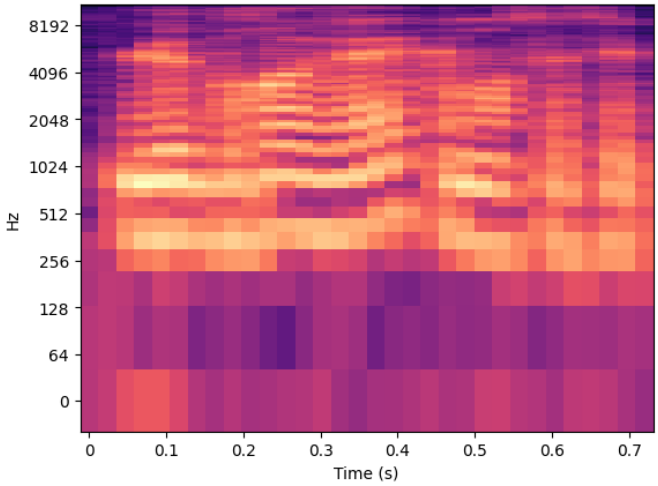

Figure 2.3: Mel spectrogram of a short audio sample.

2.2 Neural networks and deep learning

2.2.1 Overview

Neural networks are a type of machine learning model that is inspired by the structure and function of the human brain [21]. A neural network, as shown in Figure 2.5 is composed of interconnected nodes or "neurons", represented in Figure 2.4, that work together to process and learn from input data. Each neuron receives input from other neurons, applies an activation function to the input, and then passes the output to other neurons. The process of learning is done by optimizing a defined loss function which mainly depends on the targeted task.

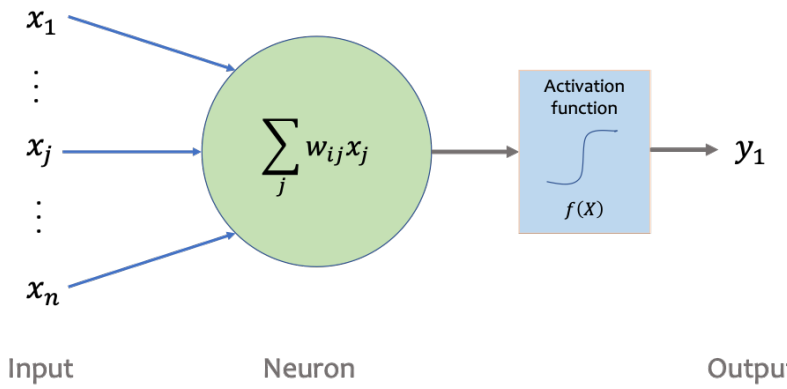

Figure 2.4: Simple artificial neuron operator scheme.

Deep learning (DL) is a subset of machine learning that involves the use of neural networks with multiple layers (DNNs), hence the term "deep". Deep learning models can learn and extract complex features from large amounts of data, making them well-suited for tasks such as image and speech recognition, natural language processing, and more.

There are 3 main types of DNN models for supervised learning [22, 21] :

- Feedforward Neural Networks, as shown in 2.5, are the simplest type of neural networks, in which the information flows in only one direction, from input to output. They are often used for tasks such as classification, regression, and pattern recognition.
- Recurrent Neural Networks or alternatively Transformers have loops in them, allowing information to persist and be processed over time. The simplest recurrent neuron unit is represented in Figure 2.6. They are commonly used for tasks that involve sequences, such as speech recognition, language translation, and time series prediction.
- Convolutional Neural Networks are designed to process data that has a grid-like topology, such as images or videos. They use convolutional layers to extract features from the input data and pooling layers to reduce the dimen-

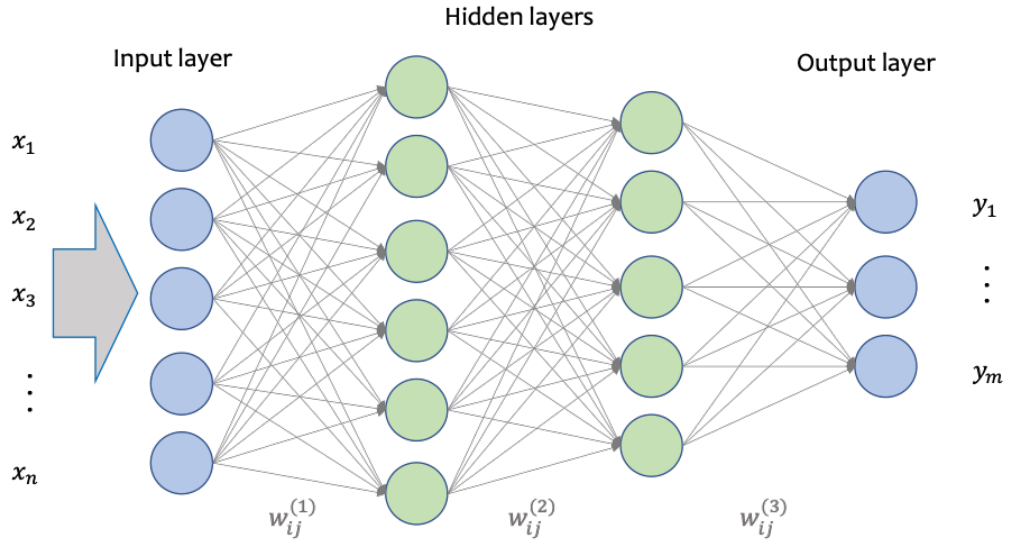

Figure 2.5: Simple scheme of a feed forward dense neural network.

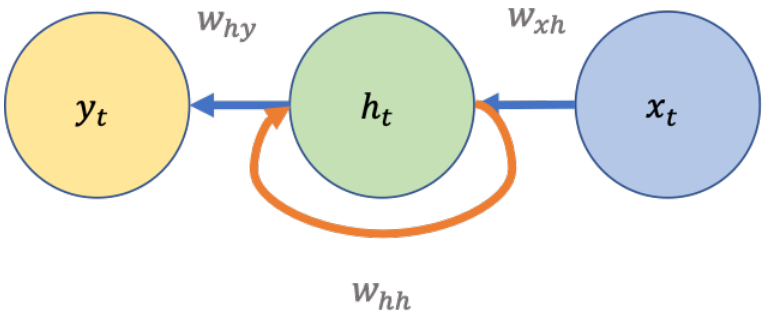

Figure 2.6: Simple scheme of a recurrent neuron unit.

sionality of the data [23]. This type of architecture is represented in Figure 2.7 They are commonly used for image and video classification, object detection, and image segmentation.

2.2.2 Typology of problems studied

Throughout this project, many machine learning approaches are discussed and tested. For a given general problem (here SER), the optimization problem can be formulated in several ways. The approach of a problem is mainly defined by the data available (type, content) and the architecture of the model. This subsection introduces important definitions for the main approaches studied during the project, this has been extensively described in a 2021 review by Shiam & al [24].

Supervised vs Unsupervised - Supervised machine learning involves training a model using labeled data (pairs of input/output), while unsupervised machine learning involves finding patterns and relationships in unlabeled data without explicit guidance.

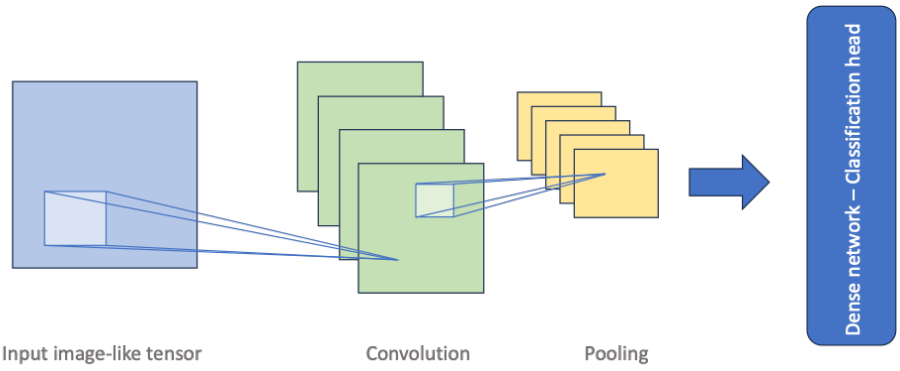

Figure 2.7: Simple scheme of a typical one layer convolution network for image classification, operation of convolution is followed by a pooling layer.

Regression vs Classification - Among supervised approaches, regression and classification are 2 main types of tasks. Regression models predict a continuous numerical output, while classification models predict a categorical output, hence, discrete.

Multiclass Classification vs Multilabel Classification - Multiclass classification involves assigning instances to one of several classes, whereas multilabel classification involves assigning multiple binary labels to instances. In other words, multiclass classification deals with mutually exclusive classes, while multilabel classification deals with non-mutually exclusive labels. Practically, it involves using a different loss function and handling output differently. For multiclass problem, a standard loss function is the cross-entropy loss function 2.2 while for multilabel problem, one may use binary cross-entropy loss function 2.3 :

$$\mathcal{L}_{cross-entropy}(Y_o, \hat{Y}_o) = -\sum_{c=1}^{M} y_{o,c} \log(\hat{y}_{o,c}) \,, \tag{2.2}$$

$$\mathcal{L}_{BCE}(Y_o, \hat{Y}_o) = -\frac{1}{M}\sum_{c=1}^{M} y_{o,c} log(\hat{y}_{o,c}) + (1 - y_{o,c}) log(1 - \hat{y}_{o,c}) \,, \tag{2.3}$$

where, in these equations, $\hat{Y}_o$ is the vector containing predicted output $\hat{y}_{o,c}$ and Y_o the vector containing ground-truth values $y_{o,c}$.

2.2.3 Training

Training a DNN requires a large amount of data. One of the main assets of DNN is that it often uses non-linear activation, which allows the network to learn non-linear patterns and, therefore, makes it very powerful in learning complex information from input data (eventually unstructured). Training a DNN is an optimization problem that relates to finding the model's weights w_{ij} of $F_{Network}$, the global forward function of the network, that minimize the defined loss function between the output from the

network and the real expected output from the training data. It is formulated mathematically in 2.4 :

$$\min_{w_{ij}}(\mathcal{L}(y_{data}, F_{Network}(x_{data}))) \,, \tag{2.4}$$

where $\mathcal{L}$ is the loss function, y_{data} the ground truth value from the data and the predicted output from the forward function of the model is $F_{Network}(x_{data})$.

The most common approach used to solve this task is the backpropagation algorithm. It is an efficient way to compute the gradient of the loss function with respect to the weights and biases of the network, which can then be used to update the parameters through an optimization algorithm such as gradient descent. An input example is fed to the network (forward pass) and the error between output and ground truth is computed. Then, starting from output layer of the network, for each neuron in a layer, the error gradient is calculated with respect to its inputs by using the chain rule of calculus.

2.2.4 Transformers models

Transformers models have been widely used in deep learning since 2017, primarily for NLP and CV tasks but also in many other domains [3]. Such models use a self-attention mechanism, which weights the importance of each part of the input sequence when processing it. The basic building block of a transformer is called an attention mechanism. The attention mechanism takes a sequence of vectors as input and computes a weighted sum of these vectors based on their similarity to a query vector. This mechanism is described by the scaled dot product attention equation 2.5 :

$$Attention(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \,, \tag{2.5}$$

where Q, K and V represents respectively query, key and values vectors and QK^T product is scaled down by $\sqrt{d_k}$, where d_k is the dimension of query and value vectors.

Transformers use multi-head attention, which means that they compute several different attention mechanisms in parallel and then concatenate their outputs, as described in Figure 2.8. This allows the model to capture relationship between elements of the input sequence at different scale level and between all elements.
The attention mechanism was already used in Recurrent Neural Network (RNN) with Long short-term memory (LSTM) architectures [25], but in 2017, a paper titled *Attention is all you need* [3] showed that attention mechanisms and especially transformer units with multi-head attention alone were powerful enough to outperform RNN-based models. Especially, transformer units do not need to process input sequences in a specific order while RNN models do. Although this gives a large advantage in learning features with global context information thanks to the attention mechanism, it allows for more parallelization for model training and thus reduces training time.

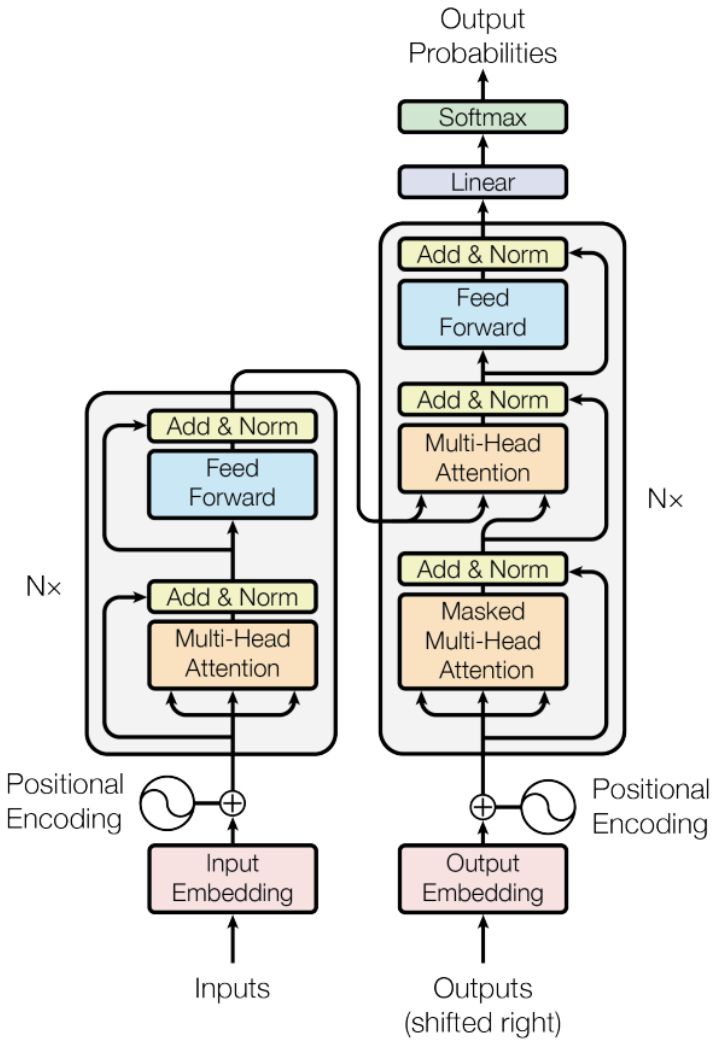

Figure 2.8: Transformers model architecture, figure from *Attention is all you need* by Vaswani & al. [3].

This led to the most recent advance in DL, especially in the field of NLP, with the training of large language models (LLMs), with several billion parameters, on massive amounts of data (see Figure 2.9). Some examples of such models are: Generative Pre-trained Transformers (the so-called GPT, which is the base model for ChatGPT), Bert (Google) [26], and many other models.

2.2.5 Transfer learning and fine tuning

During this project, transfer learning and fine tuning techniques of open source models have been extensively used. Fine-tuning and transfer learning are two related techniques in deep learning that are used to get high performance with machine learning models by leveraging pre-existing knowledge.

Transfer learning is the process of taking a model that has been pre-trained for a specific task that is in some way related to the new task. The idea behind transfer

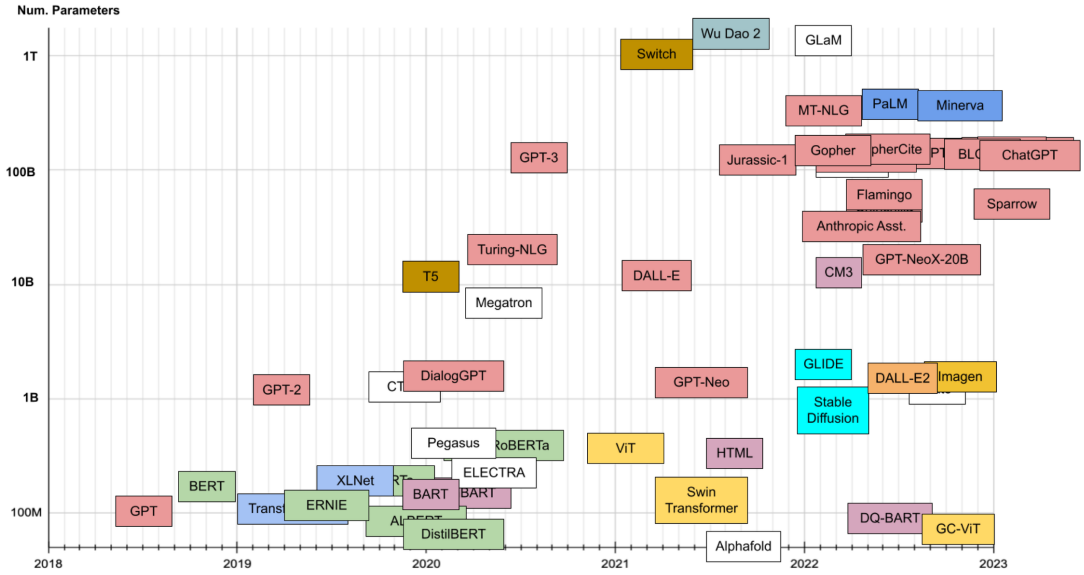

Figure 2.9: Catalog of recent transformers models in 2023, by X. Amatriain [4]. It gives the year of release vs the number of parameters in the model.

learning is that the pre-trained model has already learned general features that are useful for a wide range of tasks, and that this knowledge can be transferred to the new task. Especially, models already trained on very large dataset for general tasks (e.g. Object detection on images with ResNet [27]) can be used on smaller datasets for more specific tasks (e.g. Flower recognition).

Fine-tuning is a specific form of transfer learning where the pre-trained model is further trained on a new dataset, often smaller, that is similar to the original dataset it was trained on. The weights of the pre-trained model are adjusted during this process to better fit the new dataset. When fine-tuning a model one can use a pre-trained model and freeze part of the layers, or just fine tune the whole model and eventually additional layers (e.g. when adding classification head on embedding model). This process has been proven to be very efficient especially as very large model (such as SOTA-models for CV or NLP) are expensive to train since they requires very large hardware resources (Graphics Processing Units - GPU - and Tensor Processing Units - TPU) [28].

2.3 Deep Learning Applied to Speech Analysis

Machine learning with audio data is not an easy task, and it is a less developed field when compared to Computer Vision (CV), Natural Language Processing (NLP) with text data, or more common tasks based on structured data. Most of speech analysis task can be considered as part of the larger field of NLP. NLP is the field of Machine Learning that processes unstructured language data and is often associated with text-related tasks. There are various tasks considered in NLP, ranging from classical tasks such as classification or clustering to more complex tasks such

as text summarization or text generation (e.g. ChatGPT). ASR involves many different tasks and is a key component of voice assistants such as Google Home, Alexa (Amazon), or Siri (Apple).

Figure 2.10 shows the general architecture for Automatic Speech Recognition models. The Digital Signal Processing (DSP) part handles the preparation of the data with eventual pre-processing (e.g. noise reduction, frequency filtering, resampling, etc.) and then the feature extraction is performed. The feature extraction is always specific to a model because it defines the size of the input tensor to the network [20]. In this project, mainly pre-trained models have been used for fine-tuning and inference, thus the feature extraction methods used were already implemented along with the neural network model architecture.

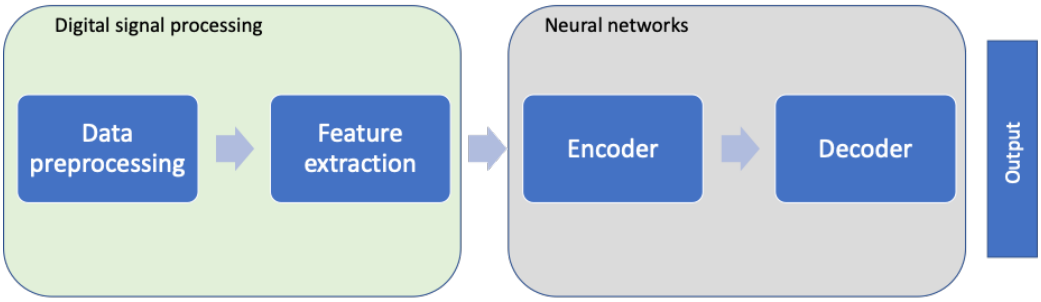

Figure 2.10: Typical Automatic Speech Recognition algorithm general architecture.

The feature extraction part takes the raw audio signal as input in the form of a single-dimensional vector with a defined sample rate ($16kHz$ in this project) and performs FFT to construct the Mel Spectrogram of the input audio. The resulting output tensor is defined by the parameters of the FFT and Mel Spectrogram. The main parameters of the feature extraction are defined in Table 2.1. Especially, *feature_size* and *n_fft* define the size of the resulting tensor that will be passed to the NN.

Then, the neural network part is often made of an encoder and a decoder part. The encoder handles the embedding of extracted features of the mel spectrogram in the latent space. The size of encoded tensor depends on the architecture of the model. The encoder is the part that is always loaded pre-trained in this project. The quality of the embedding in the latent space is crucial because it needs to extract the meaningful information for the considered task.

Finally, the decoder is responsible for generating the output of the network. The

Table 2.1: Main parameters of typical sequence feature extractor based on Transformers library source code [29].

Parameter	Definition
feature_size	The feature dimension of the extracted features
sampling_rate	The sampling rate at which the audio files should be digitalized expressed in hertz
hop_length	Length of the overlaping windows for the STFT used to obtain the Mel Frequency coefficients.
chunk_length	The maximum number of chuncks of "sampling_rate" samples used to trim and pad longer or shorter audio sequences.
n_fft	Size of the Fourier transform.

architecture of the decoder depends on the task considered and may take several forms with an input tensor of size defined by the size of the embedding in the latent space and output the target information : text sequence, classification output, regression output, etc... This part of the network can be loaded pretrained (e.g. speech-to-text, speech-to-speech) or initialized randomly, for specific classification or regression tasks for example.

In the following subsections, the main DL tasks studied in this project are introduced briefly along with the dedicated SOTA models architectures.

2.3.1 Speech-to-Text

Speech to text (STT) is a task in which spoken language is automatically transcribed into written text. The goal is to retrieve the text transcription of a given input audio containing spoken language as precisely as possible. This task is a sequence-to-sequence task, which means that the network will take a sequence as an input (i.e the tensor of extracted features from the audio signal) and output a sequence, primarily a sequence of numeric tokens that can be transcribed into a sequence of words, i.e the predicted text transcription.
The STT task is the most studied problem within the field of ASR. The first STT algorithm was developed by Bell Labs in the 1950s and was based on pattern matching techniques and was able to recognize a restricted vocabulary [2]. The major improvement in STT technologies occurred with the development of Deep Neural Networks (DNNs), which can model complex relationships between the acoustic features and the corresponding text transcriptions. In 2014, the first end-to-end deep learning-based speech recognition system, called DeepSpeech, was developed by Baidu [30]. This system directly maps the audio waveform to the corresponding text transcription using a deep neural network, bypassing the need for feature extraction and other processing steps.

More recently, the performance of STT models significantly improved thanks to transformer-based models. In 2020, Wav2Vec2 developed by Facebook AI Research introduced a self-supervised method [7]. The model was trained on a large dataset of

untranscribed speech data and learned to predict the original audio waveform from a sequence of masked or corrupted versions of the waveform. This pre-training stage was followed by fine-tuning on a smaller dataset of transcribed speech data to adapt the model to a specific speech recognition task. In September 2022, OpenAI released the open source model Whisper, trained on 680,000 hours of audio. This large scale training enables the model to achieve human-like performance on English language [6]. This project leverages both the Wav2vec2 and Whisper models extensively to perform audio embedding and STT tasks.

2.3.2 Speech Emotion Recognition

Overview

Speech Emotion Recognition (SER) is a task within the field of Speech Processing and Automatic Speech Recognition that aims at predicting the tonality and emotion expressed in an audio sample. It is a very complex task since emotion and tonality are directly associated with human social behavior. This means that a Speech Emotion Recognition DNN should learn to mimic human-like emotional intelligence. This type of problem depend highly on data since learned patterns are highly influenced by the quality, quantity and potential bias of the training data. Therefore, it requires a large amount of high-quality data.

Moreover, for SER with DNNs, specialized architectures and techniques such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs) (or alternatively Transformers) are required, and often a combination of both. These architectures and techniques have been shown to be effective for capturing and modeling the complex temporal and spectral characteristics of speech signals associated with emotional states [16]. For handling audio embedding for SER task, encoder models such as Wav2vec2 are suitable since they rely on both CNNs and Transformers layers.

In the context of speech emotion recognition the encoder part of the model, as defined in Figure 2.10, is a submodule of the global neural network that takes embedded audio as input and outputs a vector representing the emotion classes (generally one-hot-encoded). It is refered as the classification head. Meaningful feature extraction is done through the embedding (or encoder) part of the network. In this project, classification and regression heads are designed as simple dense feed forward networks with 2 to 3 layers, as shown in Figure 2.11.

Dense layer is the most common type of neural network layer. Layers of neurons are fully connected to each other. Rectified Linear Unit (or ReLU) activation function 2.6 is also one of the most common activation function used in neural network [21] :

$$ReLU(x) = max\{0, x\}\,. \tag{2.6}$$

Dropout is a regularization technique used in deep learning models to prevent from overfitting. The dropout operator works by randomly dropping out (i.e., setting

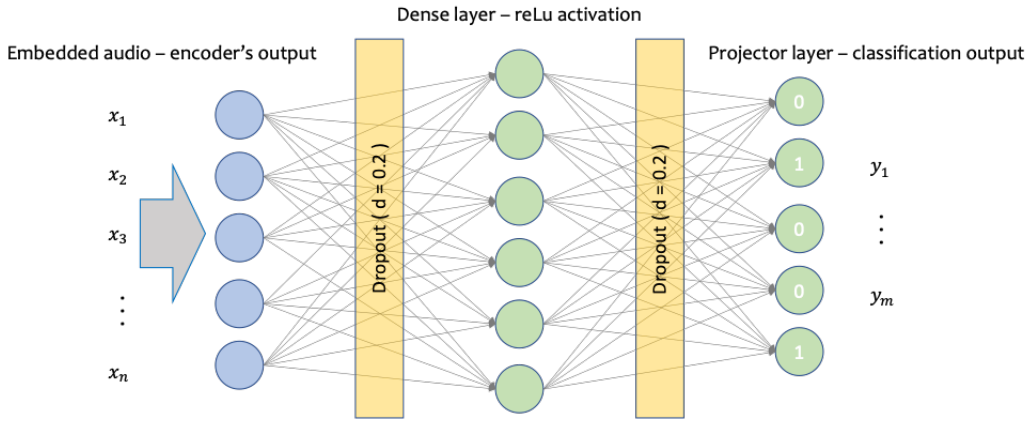

Figure 2.11: Basic architecture of classification head used in the project.

to zero) a certain percentage of the input units of a layer during each training iteration. This means that some neurons will be temporarily ignored during the forward pass, and their weights will not be updated during the backward pass. By randomly dropping out neurons during training, the network is forced to learn redundant representations for each input, reducing its reliance on any single input feature. This can help prevent overfitting, which occurs when the model learns to memorize the training data instead of generalizing to new data. Common values for the dropout rate are between 0.2 and 0.5.

Metrics and evaluation

Since SER is basically a classification task, or alternatively a regression task depending on the formulation of the problem, standard metrics for supervised tasks can be used, such as accuracy and F1-score.

Accuracy is the ratio of correct predictions to the total number of predictions. It is a very simple metric; however, when dealing with an unbalanced dataset where some classes are underrepresented, it lacks information on how well the classes are recognized and predicted. This is especially important for critical applications, such as disease detection. More precisely, accuracy in equation 2.7 can be defined with the notions of *True/False Positive and Negative* as :

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}\,. \tag{2.7}$$

In this equation, TP stands for "True Positive," which refers to the number of correctly identified positive cases; FP stands for "False Positive," which refers to the number of incorrectly identified positive cases; TN stands for "True Negative," which refers to the number of correctly identified negative cases, and FN stands for "False Negative," which refers to the number of incorrectly identified negative cases. Precision measures the proportion of correct positive predictions out of all positive predictions made, while recall measures the proportion of correct positive predictions out of all actual positive cases.

A more meaningful metric is the F1-score , which is widely used to measure classifier performance. It is defined as the harmonic mean of precision and recall (2.8 and 2.9) as :

$$\begin{cases} Precision = \frac{TP}{TP+FP} \\ Recall = \frac{TP}{TP+FN} \end{cases}, \tag{2.8}$$

$$F_1 = 2\frac{Precision \times Recall}{Precision + Recall}. \tag{2.9}$$

In multiclass and multilabel classification problems, the F1 score is computed for each class, giving an F1 score per class. Then, a global metric can be computed as an F1 micro score, which is the precision and recall across all classes. The F1 micro score is similar to accuracy and is heavily influenced by major classes in the dataset. Alternatively, one can compute the F1 macro score by averaging the precision and recall across classes and computing the global F1 score. The F1 macro score is preferred in this project (and generally in classification problems with unbalanced datasets) since it weights each class equally independently from its size in the dataset. This leads to a better model evaluation.

When dealing with a regression model, where the output is a floating value (or a vector of floats), one can use the standard Mean Squared Error (MSE) 2.10 :

$$\mathcal{L}_{MSE}(Y_o, \hat{Y_o}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2, \tag{2.10}$$

where y_i represents the actual value of the target variable for the i-th observation, $\hat{y_i}$ represents the predicted value of the target variable for the i-th observation, and n represents the total number of observations in the dataset. The MSE is a popular metric used to evaluate the performance of regression models, and it represents the average of the squared differences between the actual and predicted values of the target variable.

Alternatively, the coefficient of determination (named "R squared" or R^2) 2.11 can be used to assess how well the trained regression model is able to represent the distribution of training data :

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y_i})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}. \tag{2.11}$$

While these metrics are not as meaningful as accuracy for classification problem in the context of SER. They are still useful to compare models one to another and were used to assess the global quality of a model.

2.3.3 Speaker Diarization : Who spoke when ?

Overview

Speaker diarization is the process of automatically identifying and segmenting an audio recording into distinct speech segments, where each segment corresponds to a particular speaker. In simpler words, the goal is to answer the question : *Who spoke when ?*. It involves analyzing the audio signal to detect changes in speaker identity, and then grouping together segments that belong to the same speaker. The standard architecture for speaker diarization pipeline is represented in 2.12.

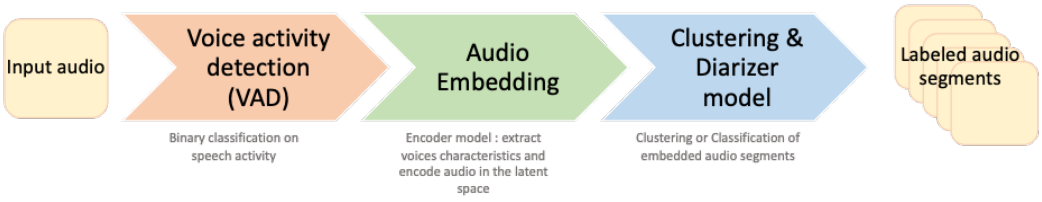

Figure 2.12: Typical deep learning based speaker diarization pipeline.

There are two main approaches for such a problem [8]:

- Supervised approach / Classification problem: The model is trained to recognize a finite number of speakers (i.e. classes). Thus, it is expected that it only works with speakers that were used among the training data.
- Unsupervised approach / Clustering problem: The model clusters audio segments according to the speaker based on extracted audio features. It is the most versatile approach since it can detect the number of speakers involved (number of clusters) and assign each voice segment to a specific cluster.

This project mainly focuses on the second approach.

Metrics and evaluation

The standard metric for speaker diarization problem, described in 2.12, is the Diarization Error Rate (DER) [31] :

$$DER = \frac{FalseAlarm + MissedDetection + Confusion}{Total}. \tag{2.12}$$

This metric computes the error in diarization in terms of a duration ratio. It can be considered as a micro metric similar to standard accuracy for classification problems. Here, *False Alarm* (FA) and *Missed Detection* (MD) are related to the voice activity detection step of the speaker diarization pipeline and *Confusion* (SC) is linked to the quality of the clustering.

While DER is a convenient and simple standard micro metric to evaluate global quality of a model, conversational content can contains short speech segment that are highly important in terms of linguistic content in the conversation. Therefore,

the Conversational Diarization Error Rate (CDER) 2.13 was introduced in recent research [32]. It is similar to the DER but consider the error rate in terms of segments and not in duration :

$$CDER = \frac{NumberOfMistakes}{TotalSegments}\,. \tag{2.13}$$

The number of mistakes is computed by checking for each segment in the reference segmentation/diarization if a corresponding segment exists in the predicted segmentation/diarization.

To handle imbalanced distribution among speakers, with large difference between cluster size, one can defined the Jaccard error rate (JER), described in 2.14, which computes error for each speaker and thus weight each cluster equally when computing error rate [31] :

$$JER = \frac{1}{N}\sum_{i}^{N_{ref}} \frac{FA_i + MD_i}{TOTAL_i}\,. \tag{2.14}$$

A more advanced metrics called Balanced Error Rate (BER) was proposed recently with a sub-graph approach of reference and predicted diarization introducing a Segment-level Error Rate. This metric is much more complex but also delivers more meaningful information on the quality of a model, since it takes into account the error on a macro level by balancing error rates by speakers, and improving matching evaluation between reference and hypothesis. The details and algorithm for implementing Balanced Error Rate are extensively described in the dedicated paper [33].

3
Methods

3.1 Framework and tools overview

As part of R&D in Artificial Intelligence (AI) at La Javaness, this project tries to follow all the good IT practices and uses the IT-stack of Data and IT team of the company. Especially, the programming language Python was used for most of the development of this project except for the frontend part of the labelling tool which was coded in the JavaScript web framework *React.js*.

Python is widely used in ML development because it has very rich libraries that can handle every aspects of a ML pipeline, thus it is very convenient for quick prototyping. Its big popularity as an open source language makes it really versatile and efficient for ML and especially DL. Table 3.1 lists all the main libraries used during the project.

Table 3.1: List of main python libraries used during the project.

Library	**Usage**
PyTorch	Deep Learning Framework
Hugging Face Transformers	API for SOTA pre-trained models
Hugging Face Datasets	Data Loading and preprocessing
Pandas	Data processing and analysis
Audiomentations	Data augmentation for audio
Nemo Nvidia	Speaker Diarization framework
Pyannote	Speaker Diarization framework
Gradio	ML web app prototyping

Hugging Face's libraries [4, 34, 35] are widely used in industry and research for deep learning applications and comes with convenient features for prototyping. PyTorch was used as the main python deep learning frameworks for this project [36]. Moreover, code management was handled with git and gitlab, datasets and trained model are stored on S3 buckets (Amazon Web Services) and training was mainly performed on a remote server with a 32Go Nvidia GPU, which allowed to train very large model.

3.2 Speech Emotion Recognition

The aim of the SER part of this thesis project was to leverage state-of-the-art (SOTA) approaches for SER and improve upon previous research and proof-of-concept developed within the R&D team. Due to time constraints, we choose to adopt an iterative approach in order to get good results. Training and evaluation of models along the process of the dataset creation allowed to address arising issues with data collection and labeling within the short deadlines. Indeed, the process of creating, increasing, and improving the new dataset was done in parallel with the continuous training and evaluation of the models. This working process is inspired by the CRISP DM methodology for data driven product development [37]. It allowed the monitoring of results and continuous adaptation of the whole process of generating, labeling, and processing data along with the design and training of neural network models. Moreover, as Speech Recognition is a very active field of research, several newly released models were tested during the project, and the pipelines developed can be easily used with a wide variety of similar models, making it possible to upgrade them at any time.

3.2.1 Emotion representation

Building emotion recognition systems implies defining a representation of emotions. There are several ways to represent emotions, but the most common strategy is to create a discrete map of emotions with a small number of emotions that the system will be trained to recognize.

First, a set of meaningful emotions should be defined. Previous works on the subject within the R&D team at La Javaness proposed a multilabel approach with 9 classes that mixed common general emotions ("anger", "joy", "sadness", "neutral", "worry-depression") with more linguistic ("unsatisfaction", "hesitation") and para-linguistic ("emphasis", "surprise") features of spoken language. While it provides a lot of information and several comprehension levels of tonality expressed in speeches, it makes the classification highly difficult for a single model and makes the labeling data very likely to be biased.

This project proposes to focus on a simpler emotion mapping suitable for common SER-system commercial use cases. The new mapping proposed is based on the Russell's Valence-Arousal 2D-Emotion space [38] (also called circumplex model of emotions). This empirical model proposes to describe emotions through two independant dimensions, namely Valence which describes how pleasant is an emotion and Arousal which relates to the intensity of the emotion expression. In this work the Valence-Arousal space is split into 5 subspaces (see Figure 3.1) that approximate a complete mapping of common general emotions expressed in casual conversational content. Previous works on speech emotion recognition propose similar representations of emotions with a small number of classes based on a conceptual splitting of the Valence-Arousal space [39, 40] and inspire the proposed emotion space of this project. Generally, basic emotion mapping additionally includes "disgust" and "fear"

emotions [41] and are not considered in our SER model. A third dimension named Dominance is often used along with Valence and Arousal to describe an emotion. The 3 dimensions of the Valence-Arousal-Dominance emotional space are described in Table 3.2.

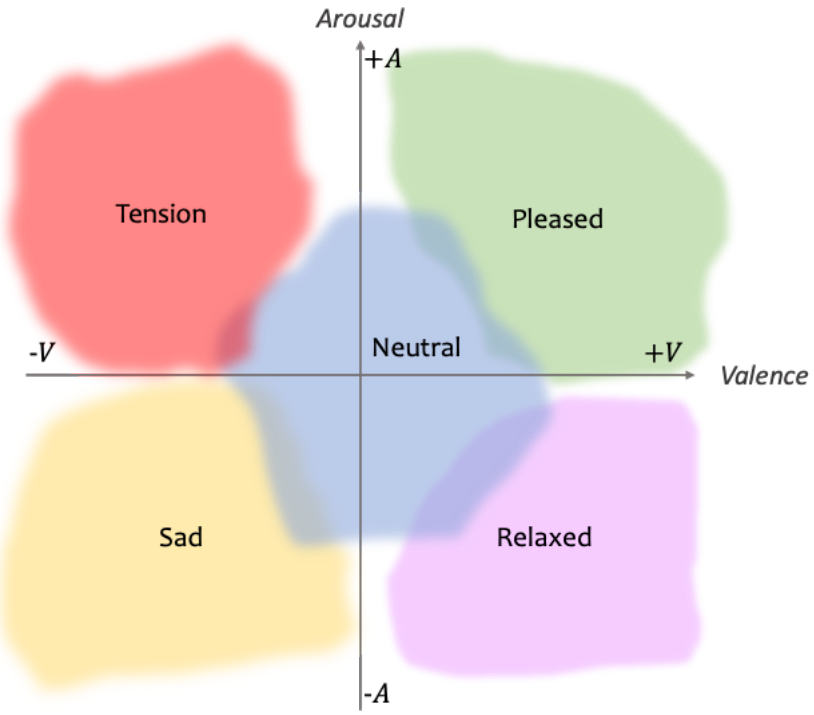

Figure 3.1: Conceptual split of Valence-Arousal emotion space into 5 discrete sub-spaces for classification task.

Table 3.2: Common definition of VAD emotion space as defined originally by Russell [38].

Dimension	Definition
Valence	Pleasantness of a stimulus
Arousal	Intensity of emotion provoked by a stimulus
Dominance	Degree of control exerted by a stimulus

Arousal and Dominance are often considered as mainly para-linguistic dimensions which refers to a non-verbal aspect of communication that conveys emotional information through variations in tone, pitch, volume, and tempo of speech. In the meantime, Valence dimension is primarily linguistic based dimension because it often relates to the context of emotion expressed.

In addition to the 5-emotion mapping, the intensity of the emotion which mainly relates to the Arousal dimension is considered separately as a binary information : Normal vs Strong intensity. This is motivate by results from [42] : By treating SER as a multitask problem with multi-class emotion classification along with binary

intensity classification, the global performance of the SER system improves.

A secondary approach of the problem was explored through a quantitative representation of emotion using the Valence-Arousal-Dominance. Focusing, on Valence and Arousal dimension, a quantitative approach can be achieve using a rating scale in each dimension. While this methods might introduce large bias without a very precise definition of the rating scale, it allows a statistical approach of the problem as in [41] which could be useful in the context of analytics for large conversational contents. In addition, despite the potential large uncertainty of such a labeling problem, it brings a significant gain in the precision of emotion representation allowing to map continuously the whole 2D-space.

To sum up, audio samples for emotion recognition are labeled on 3 different levels in this project :

- Emotion class (5 classes) : pleased, relaxed, neutral, sad, tension.
- Intensity (Binary class) : normal, strong.
- Valence-Arousal quantitative rating : 2-dimensional rating between 0.0 and 1.0.

3.2.2 Dataset

A first benchmark on available datasets in French Language for SER showed that there are a very few datasets available. The closest dataset that can be used in this project, A Canadian French Emotional Speech Dataset [13], is a research set that includes six different sentences, pronounced by six male and six female actors, in six basic emotions plus one neutral emotion. Besides this dataset, there are currently no dataset in French language publicly available for SER Classification task. This dataset does not include conversation, and the expression of emotions are very stereotyped since it is performed by actors. In addition, this dataset only includes 12 actors and 6 different sentences which clearly constitute a problem for model robustness to be used on real life conversational content. Therefore, this motivates the necessity to build a new dataset from scratch in order to train a robust model that is able to detect emotion from any voice in any context. Our dataset should include a variety of speakers male and female, preferably in real life context (not actors that are playing emotions) and expressing a large range of emotions with various intensity. Moreover, the model must be robust to variations in the quality of recorded audio.

There are 3 stages during the dataset generation as shown in 3.2, audio files are stored at each stage of the process to keep track of the processed files. In order to simplify the process and being able to easily increase the size of the dataset, a metadata format and a precise data storage organization have been defined beforehand. Then, a tool for automatically prepare samples, i.e slicing the raw audio into small samples and generates metadata have been developed as well as a web application that allows the user to label audio data in a very efficient way. The detailed method used for each stage is described in the following sub-sections.

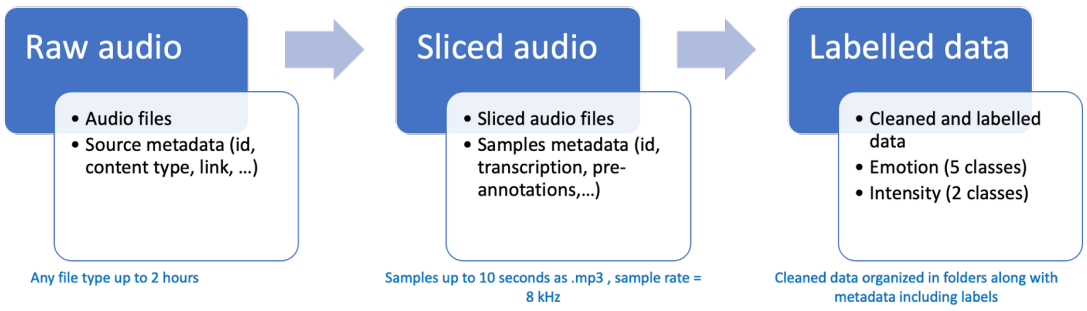

Figure 3.2: Dataset generation pipeline - it does not include eventual preprocessing that can occur after.

Raw audio selection

Firstly, raw audio sources needs to be selected wisely. There are several requirements:

1. A large number of speakers, at least 30.
2. Speeches as close as possible to real life.
3. A large part of the audio content is conversational.
4. They are a large variety of context with a large variety of emotions.
5. Good quality of audio but it can have some background noise.
6. Audio content is large enough (at least 1 hour for a given source).

In addition, considering that dataset construction and labelling is very time consuming, the audio content needs to be easily available and should contain almost only speeches in order make the slicing and processing easier to get acceptable results within the time constraint of the project. Table 3.3 sums up the selected audio sources that satisfies the requirements.

Each of these sources have several hours of content available (at least 5 hours). The most interesting source is *Strip Tease* documentary series since it provides the largest variety of conversational content. Each episode is 10 to 20 minutes long and introduces around 3 to 10 speakers in an everyday life context which includes very diverse context, hence, diverse emotional content. Therefore, it was chosen as the main source of conversational content for the SER task.

Table 3.3: List of selected audio sources.

Audio source name	Description
Strip Tease	Documentary series focusing on everyday life moments without any commentators or editing
Ultimatum	French adaptation of a reality TV-show featuring six couples
Les grosses têtes	Daily talk radio program where a panel of guests discuss current events and social issues. It is known for its lively and outspoken debates.

Automatic slicer tool

The audio slicer tool, as described in Figure 3.3, was developed to help to create consistent dataset with a clear defined pipeline. While it helps to define, automate and make the training data generation process easier, it also provide a basic and easy-to-use application to generate audio sets for augmentation purpose or even for other speech processing tasks in the future. The slicer tool is fully written in Python and it can be used as a python library to be included in notebooks or in script within a larger data processing pipeline or as a Command Line Interface (CLI) application.

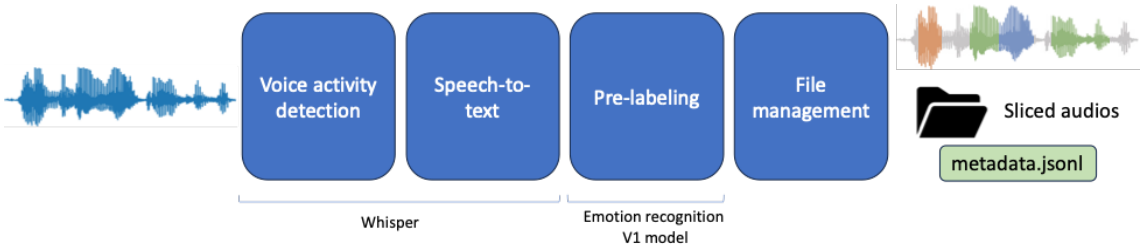

Figure 3.3: Audio slicer tool pipeline architecture.

The main features of the software are :

1. Slicing long audio files (up to several hours) into smaller audio samples (up to 10 seconds) that can be use as training data for speech processing models.
2. Managing the organization of files including the generation of metadata files to have a clean dataset ready to be labeled.
3. Generating pre-annotation from pretrained model : text transcription, tonality classification. This helps a lot during the labelling phase.

Whisper STT pre-trained model handles the text transcription of audio along with associated time stamps. This methods have 2 main advantages : It generates a text transcription with very good performances that can be corrected during labelling phase, and text is generated with associated timestamps. Thus, based on Whisper's inference timestamps the slicer tool generate audio samples along with text transcription. Each audio file has a unique id and is stored in a folder along with the metadata.

Based on the work done afterwards on speaker diarization, the slicer tool app was updated to better handle the slicing by using Whisper's inference transcription and timestamps after a first slicing handled by a Voice Activity Detection model (VAD). This helped to generate high quality data and improve performances of Whisper on transcription and timestamping tasks.

3.2.3 Data Labelling

Labeling audio data is quite a difficult task since it does not rely on visual access to the data but only on audio content. Thus, it makes the labeling process longer and the data more likely to contains errors and bias. To tackle this issue, a labeling web app has been developed to offer an efficient and convenient tool to label good quality data quickly. This tool is built as a web application where the annotator can listen to samples, edit metadata such as tags, label samples with discrete emotion and intensity classes and write (or correct) the text transcription. An additional feature of the web app is introduced to label audio on a 2-Dimensional Valence/Arousal map in order to create a dataset for regression SER models. This features, as shown in Figure 3.4, is very convenient and also add a visual interpretation of emotions contained in audio. The feature has been specifically designed for the Valence-Arousal labeling. Hence, this tool was extensively used to create a new original dataset for real-life French conversational content. The whole dataset was labeled by the same person.

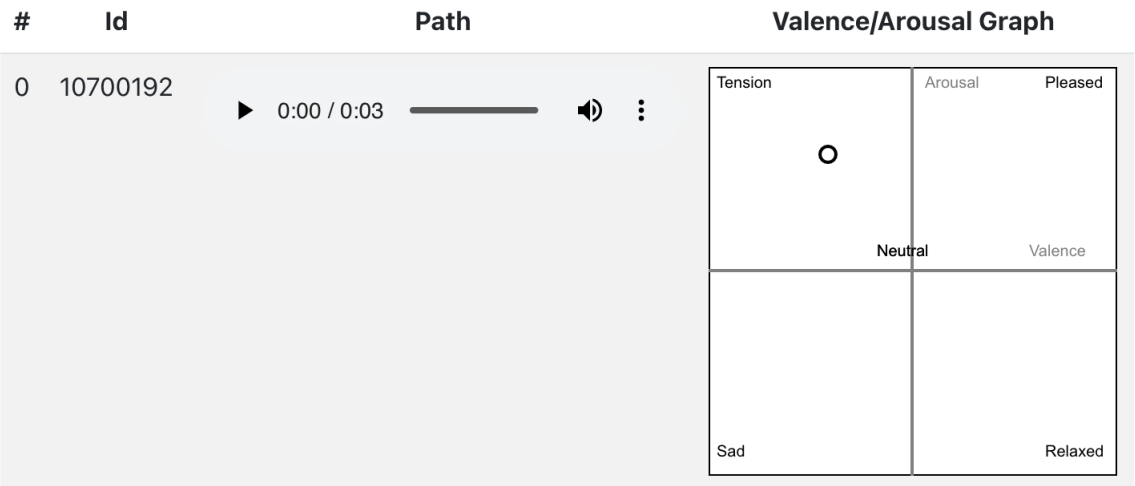

Figure 3.4: Screen capture of Valence-Arousal 2D scale labeling feature.

Data cleaning

While it was needed to quickly label a large quantity of data to be able to fine tune a classification model and get results, it was difficult to assess in the first instance how good should be the audio quality of samples to get good performances from the trained model. Hence, firstly, a large quantity of data has been labeled to be able

to train properly a first model. Then, the dataset was refined as it was increased, to get higher data quality.

The final cleaned dataset contains $N_{cleaned} = 2513$ samples against $N_{original} = 4512$ samples in the original set. The cleaning process is described in Table 3.4.

Table 3.4: Data cleaning criterion.

Description	Criterion
Too short file	<1.0s with no clear spoken content
Too long file	>10.0s
Too much content	more than 3 emotions clearly expressed
No emotion	there is no clearly identified emotional content
Noisy sample	Too noisy to clearly understand the voices

3.2.4 Data augmentation with third party datasets

As mentioned previously, Speech Recognition tasks with DNN are strongly data centric problems that relies on large quantity of high quality data. Data augmentation is the set of techniques that allow to increase the size of the dataset, i.e. to have more training examples. While already available datasets does not match exactly the targeted task, they still can be used partly or entirely to increase training data. Several third party dataset [43, 13] were included in the training data with a filtering and/or processing as described in Table 3.5.

Table 3.5: List of third party datasets used for data augmentation.

Dataset	Description
Call My Agent dataset	9-emotions multilabel dataset used in previous PoC
A Canadian French Emotional Speech Dataset	12 actors performing 6 sentences with 6 emotions
IEMOCAP	English benchmark dataset for speech emotion recognition

3.2.5 Speech emotion recognition model

This section presents the different classification models designed and trained.

Classification models

Audio classification models generally process Mel Spectrogram (frequency domain representation) as input. The spectrogram input is encoded in a latent space using an embedding model responsible for extracting important features and outputs a reduced size vector or matrix that will be further processed by a classification head to output a vector of the targeted dimension. This architecture, as represented in

Figure 3.5, is the basic architecture we used to design and train our models.

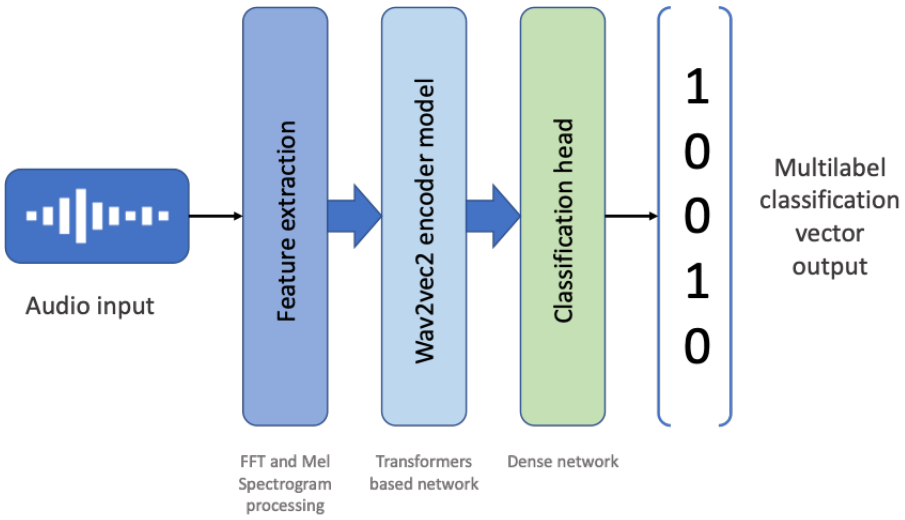

Figure 3.5: Simple classification model based on wav2vec2 model for audio embedding.

The main stage in this architecture is the encoder model which needs to be able to extract important features of voices. This type of model has a very large number of parameters (see Table 3.6). It is a general purpose model that is trained to transform an input tensor of a given dimension to a smaller one in a latent space (i.e. dimension reduction), it is very conducive to the use of transfer learning and fine tuning techniques to leverage the quality of pre-trained models on massive amount of data. Currently, the best encoder models for speech recognition task (including classification of voice content) are Wav2vec2 (Facebook) [7] and Whisper (Open AI) [6]. There are a lot of pre-trained version of these open source models which can be easily downloaded and manipulate (thanks to Hugging Face libraries [29]).

Table 3.6: Number of trained parameters in main SOTA encoder models.

Model	Parameters
Wav2vec2 - Base	95M
Wav2vec2 - Large	317M
Whisper - Tiny	17M
Whisper - Base	74M
Whisper - Small	244M
Whisper - Medium	769M
Whisper - Large	1550M

Since mainly pre-trained models are used in this project the feature exctraction part is defined given the architecture of the model. Hugging Face's libraries [4] provide complete pipelines with implemented classes and objects to handle feature extrac-

tion, processing and modeling with the studied models.

The last stage of the network is the classification head which was designed in this project as a simple feed-forward dense network. Since latent space provides an abstract representation of input audio in small dimension, a simple feed forward network with dense layers can handle the classification task from the latent space to the output vector. Having a simple architecture with a small number of parameters for the classification head makes it easier for hyperparameters tuning, it prevents from overfitting by avoiding too large number of parameters and it makes the training more efficient since the size of the classification head is negligible compared to the size of the encoder. The base architecture is represented in Figure 3.5. With the first basic architecture, two approaches are tested : Multiclass classification or Multilabel classification.

While basic architecture might lead to good performance, available data allow us to design more advanced architectures that might improve final performance of the model. Firstly, based on literature, adopting a multitask approach of the problem by adding a secondary task can improve performance on the main task (i.e. emotion classification) [42]. The architecture of the multitask approach is represented in Figure 3.7. The first task is the classification of emotion (as a multilabel or multiclass problem) and the model is trained with a secondary task as a binary classification problem on the Intensity (which is labeled as *Normal* or *Strong* in the dataset).

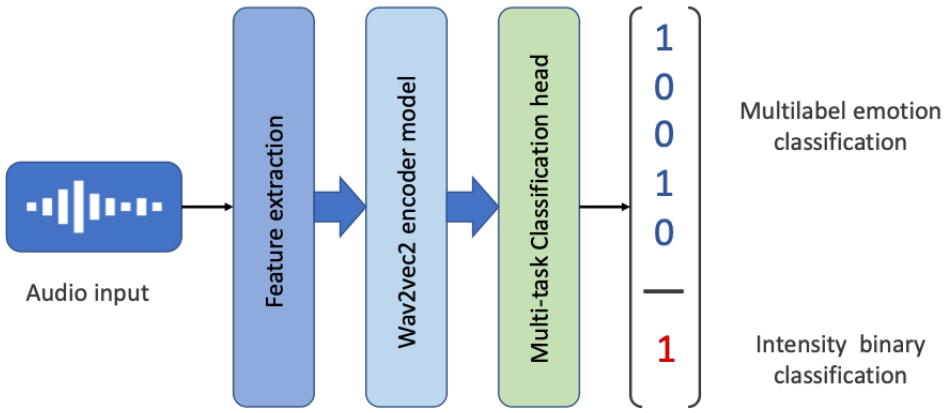

Figure 3.6: Multitask classification model architecture : the model predicts emotion as a multilabel classifier and predict intensity as a binary classifier.

Then, another alternative approach is to take advantage of the very good transcription quality made by the whisper model and to consider the text transcription of the audio as an input to the SER model (see Figure 3.7. This multimodal approach has a more complex architecture with 2 encoders model : the audio encoder (wav2vec2) and a text encoder (bert) which similarly to the audio creates an embedding vector in a latent space of the input text. This approach should ensure that purely linguistic features are also considered when predicting emotions.

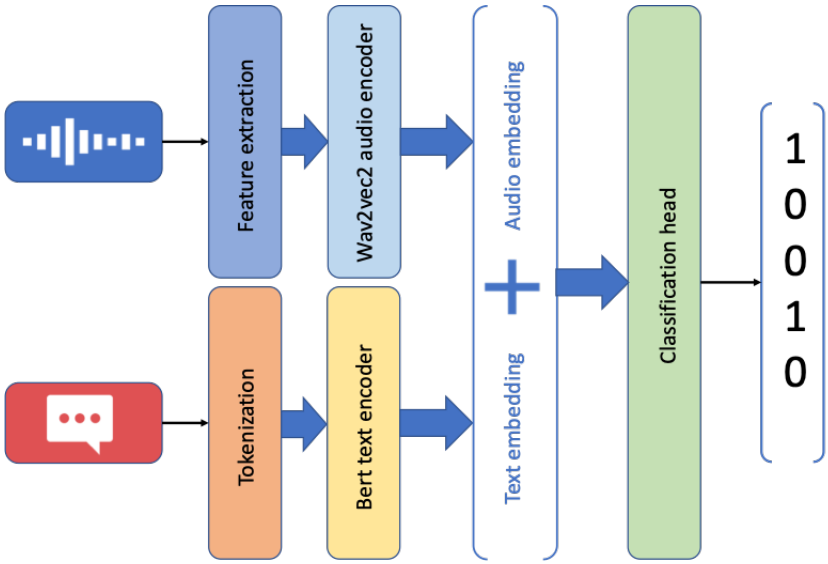

Figure 3.7: Multimodal architecture : Audio and text transcription are used as a multimodal input for the network (using Bert model for text embedding and Wav2vec2 for audio embedding).

Quantitative regression model

The classification approach is the most obvious because it allows to assign directly understandable tags to determine an emotion. However there are several disadvantages to relying on this discrete approach. Firstly, once the problem has been defined, and the data labeled, the model will only be able to predict the discrete set of emotions. Secondly, the number of classes considered is quite small and, the more classes in the problem, the more data needed which quickly limits the number of classes since data collection and labeling is highly time consuming and,thus, expensive. Finally, it limits the postprocessing possibility to compute advanced statistics.

When the data can be labeled quantitatively with continuous features, it can be very beneficial. An alternative approach of the problem involves the architecture described in Figure 3.8, the main difference is the use of a regression head which is similar to the classification head but uses different activation and loss function. On the one hand, the output as a 3 dimensional vector provides more information than a single (or multiple) category, and from this 3 defined dimension predicted (Valence, Arousal, Dominance), one can defined specific emotion sub-spaces depending on external factor such as the context. On the other hand, it provides quantitatively interpretable information, that can be further process for statistics and analysis purpose on long conversational content for example.

Summary of tested approach

Table 3.7 sums up the different approach tested for SER. Most of them are based on Wav2Vec2 model. These approach have been iteratively tested along the creation and the improvement of the dataset. Thus, the performance of each model have

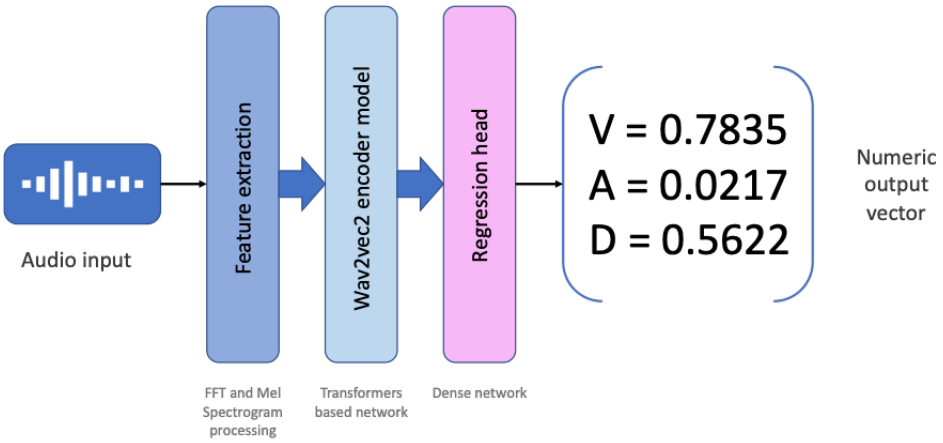

Figure 3.8: Quantitative regression architecture.

been monitored along with different dataset. This makes the evaluation of the different approach even better and allows to assess wether each architecture/approach requires more or less data than another.

Table 3.7: Summary of tested model and tasks.

Model	Task
Wav2Vec2-based simple	Multiclass classification
Whisper-based simple	Multiclass classification
Wav2Vec2-based simple	Multilabel classification
Wav2Vec2-based simple	Multitask (Multilabel + Binary) classification
Wav2Vec2+Bert-based	Multimodal multilabel classification
Wav2Vec2-based quantitative	3-Dimensional Regression

3.3 Speaker Diarization

As described previously, *Speaker Diarization* is the task that aims at determining : *Who spoke when ?*. The primary objective through this project was to leverage pre-existing SOTA approach and try to optimize the performance on specific use cases. A secondary objective was to evaluate hardware requirements as well as to assess the feasability of a real time speaker diarizer, or alternatively, trying to reduce as much as possible the computing time.

3.3.1 Framework and models

The principle of speaker diarization is quite easy to understand : identifying and segmenting an audio recording into distinct speech segments, where each segment corresponds to a particular speaker. It has becoming a well known problem, however, there are not many frameworks available in Python for this task.

Currently, the main framework are :

- **pyannote.audio** [17, 18] which is an open-source library based on pytorch framework dedicated to speaker diarization
- **Nemo** developed by Nvidia [44] is an open-source deep learning framework, also based on pytorch, for NLP tasks with a large emphasis on ASR. It provides tools (pipeline, trainer, pre-trained models, etc...) to build AI applications.

Other models and approaches exist [45] with various model architectures. Other SOTA architectures for speaker diarization includes bidirectional long short-term memory (BLSTM) [46]. However, speaker diarization involves a complete pipeline with several stages including different deep learning models for : Voice Activity Detection (VAD), Speaker Embedding, Clustering. Therefore, these two frameworks are very convenient since they implement complete pipelines with custom building blocks and provides pre-trained models. In terms of architecture, both framework provide similar pipeline with some small differences within the models :

1. **Voice Activity Detection** : Pyannote's default model is PyanNet which has the particularity to take raw audio waveform as input, hence, limiting signal processing [47]. Nemo implements MarbleNet (convolution network) [48] which takes the usual Mel Spectrogram as input.
2. **Speaker Embedding** : Both model are quite similar in their architecture, and encode input segmented audio in a latent space to extract meaningful features from speakers. Pyannote default model is called ECAPA-TDNN [49] and Nemo's default model is TitaNet [50].
3. **Clustering / Diarizer** : While Pyannote is using standard clustering algorithms (Hidden Markov Model), Nemo implements a deep-learning based clustering called *Multi-scale Speaker Diarization with Dynamic Scale Weighting* [51]. This approach relies on multiple embedding with different time scales (window size and shift length) to optimize clusters. Thus, it requires more hardware resources and time to compute.

To quick start with the models, the default recommended models and parameters are used and constitute the baseline for testing the models. Indeed, these speaker diarization libraries come ready to be used out-of-the-box with a suggested set of parameters for common applications. In practice, Nvidia provides a catalog of pre-trained models that can be used within Nemo's pipeline. Pyannote's also provides tools to extensively customize speaker diarization pipeline and replace default models with customs one.

3.3.2 Labeling

Speaker Diarization includes segmentation to detect where there are people speaking (Voice Activity Detection) and clustering to identify speakers. To be able to evaluate the quality of diarization, a reference speaker diarization is required. Labelling audio and especially segmenting is quite a long process and, hence, requires efficient tools.

Audacity is an open-source recording and audio editing software. It is developed since 1999, thus, it is a well known and supported software widely used for audio editing and processing [52]. The software comes with a marker function that is quite convenient for the speaker diarization task and works as an independant track along the audio track. Markers tracks from audacity can be exported as *csv* like files that can be easily parsed. Parsing function are implemented in a custom python package along with the testing pipeline and other useful functions as described in the next subsection.

3.3.3 Evaluation and optimization

There are two challenges in this part of the project : Get high quality results (small error on diarization task) through a deep understanding of the behavior of the models and optimizing computing time and required hardware resources for commercial and production purpose.

Test cases

Two test cases are considered :

1. **Simple test case** : 2 men speakers in the context of a TV interview (30 minutes long). High quality recording, no background noise, a very few overlapping voices.
2. **Complex test case** : 7 speakers identified in the context of a TV documentary (15 minutes long). Varying quality of recording (from poor to good quality), some background noise (outdoor working environment), some overlapping voices.

These 2 test cases serve as a baseline to optimize hyperparameters of the model and measure computing time and hardware requirements. Audios are long enough to consider results representative of the performance of the model.

Evaluation strategy

To evaluate diarization pipeline from both frameworks, the default recommended parameters are used at first and used to pre-label the audio test cases. Then, the reference file is created by relabeling correctly the most relevant prediction to create a reference file for both test cases. From this point, it is possible to understand the global behaviour of each model in terms of VAD, segmentation and clustering.

Then, hyperparameters can be tuned to improve diarization quality, and different measures can be performed to monitor hardware requirements and computing time. All these measurements are implemented in python scripts with standardized pipeline which insure their repeatability and the liability of measures.

Three ways to reduce computing time are identified :

- Hardware resources (GPU/CPU) and parallel computing parameters

- Models hyper-parameters, especially window/frame related parameters and embedding parameters
- Pre-processing input file or in industrial application context adapting recording settings.

While it is convenient to have large GPU resources when using large language and audio processing models, these resources are also very expensive and thus may be limited in production context. In addition, the carbon footprint must also be considered when performing large-scale inference, especially when deploying large-scale pipelines.

3.4 Demo application

This project aims at improving previous proof of concept for a multitask speech analysis tool and adding new features to the pipeline. Since, the development of such a tool is meant as a demonstration for potential commercial application, one must be able to showcase the results in a fashionable way. Thus, the complete pipeline is implemented as part of a simple web application to demonstrate capability of the model with a user friendly interface.

3.4.1 Speech processing pipeline

Once the SER model trained and the diarization pipeline optimized for a given use case. They can be implemented in a larger speech processing pipeline that includes :

1. Speaker diarization to generate audio segments.
2. Speech-to-text transcription to generate text from audio segments.
3. Emotion and intensity recognition models which process audio segments.
4. Various NLP tasks can be performed on the text transcription such as : Zero-shot topic classification, text tonality recognition, translation, summarization, etc.

While speaker diarization along with speech-to-text transcription constitute the base steps of the pipeline to generate short audio and text segments, a large variety of tasks can be performed thereafter which makes the developed pipeline very flexible. The scheme of the complete global architecture of the pipeline is represented in 3.9.

The application is developed using Gradio library [35] which provides very convenient tools to build web applications for demonstration purpose. The application is built as flexible as possible allowing to turn on/off each of the performed task, changing the working language and can currently handle pre-recorded audio file and record from microphone for direct processing.

3.4.2 Optimization of inference time and required ressources

The complete pipeline includes multiple large deep learning models that process either audio file or text. One major issue when sequentially computing with large

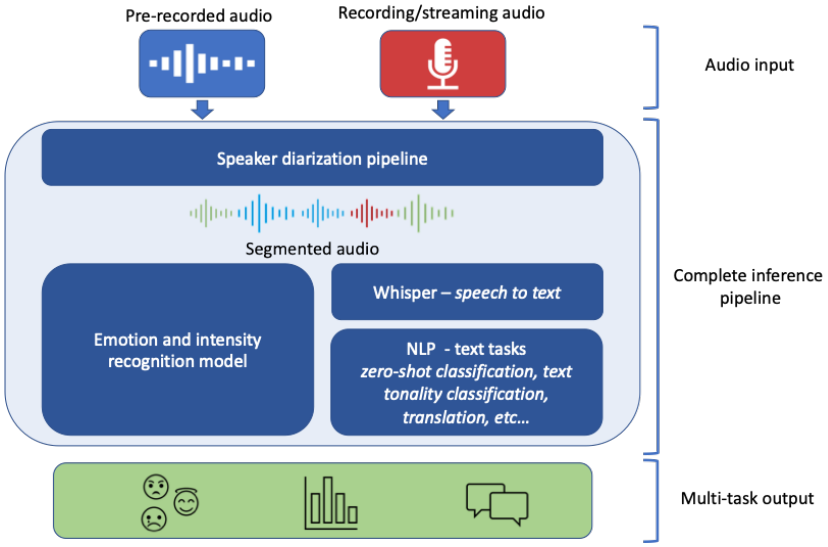

Figure 3.9: Scheme of the complete pipeline leveraging : speaker diarization, automatic speech recognition, speech emotion recognition and various NLP tasks.

models is the inference time and the reserved memory on GPU. To reduce computing time one must review carrefully object types manipulated throughout the pipeline. Especially, when processing one long audio file, the audio is loaded as a unique tensor and passed to the speaker diarization pipeline. Then, the input tensor is truncated iteratively to be pass as smaller inputs for text to speech model (Whisper) and audio classification models. This method insure that the audio file is loaded only once at the beginning of the pipeline and that a single tensor is used as a base input for the whole pipeline.

Optimizing such a pipeline is quite challenging as it involved many models that requires large hardware resources. While the pipeline is able to perform multi-task in a reasonable time, it is currently not able to perform all the tasks real-time and this issue still needs to be further investigated. This is elaborated further in the discussion chapter.

4
Results

This section sums up all the main results obtained throughout this project. Speech Emotion Recognition and Speaker Diarization sections presents the results of model training and optimization while the last section presents the final proof of concept web application.

4.1 Speech Emotion Recognition

This section includes an overview of the newly built dataset and the results of the training of the different models and approaches of the SER problem.

4.1.1 Datasets

As mentionned in previous chapter, the datasets has been created iteratively along training and testing process of models. The first batch of annotation allow to get a dataset containing 2.56 hours of audio. This larger set is called *uncleaned dataset* in Table 4.1. This first dataset did not lead to good results and thus it was needed to perform a large data cleaning by relabeling the dataset. This leads to the *cleaned dataset* which was used to train the final classification model. The *Valence-Arousal dataset* is a subset of the *cleaned dataset* and was labeled quantitatively on Valence and Arousal scale (from 0 to 1) and was used to train the regression model.

Table 4.1: Size of the datasets used to train emotion models.

Dataset	Size (samples)	Length (in hours)
Uncleaned dataset	3598	2.56
Cleaned dataset	2513	1.87
Valence-Arousal dataset	835	0.64

Figure 4.1 represents the distribution of sample length among datasets. It shows that the cleaning of the data change the distribution of sample length. Especially, a large part of very short samples (< 2s) have been removed. Indeed, very short samples are less likely to contain clear linguistic and emotionnal content.

Then, one need to check the distribution of classes among the dataset since unbalanced dataset can have influence on the quality of the trained model (see Figure 4.2. In addition, one might need to check that the data has a distribution that is somehow representative of the real life context for targeted use cases. As expected, *Neutral*

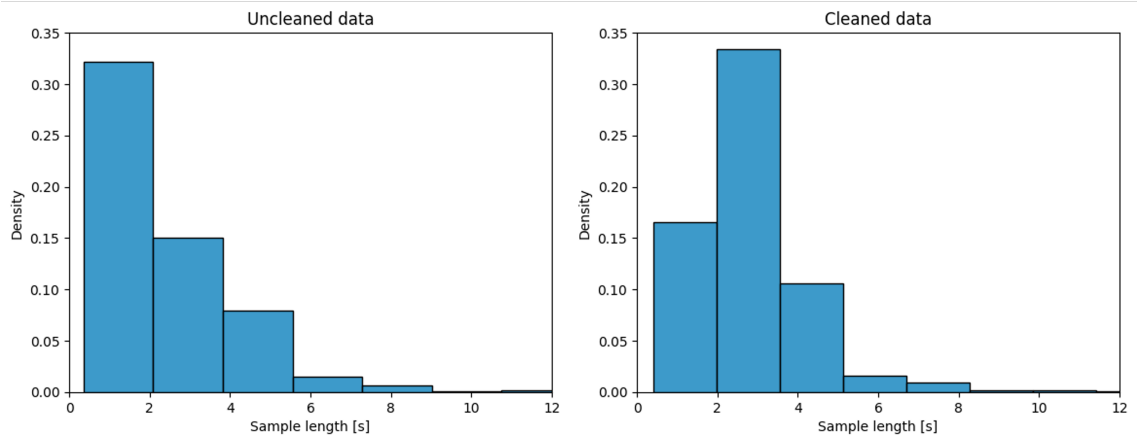

Figure 4.1: Distribution of sample length within datasets. Data-cleaning process change the distribution of sample length, especially a large part of very short samples have been removed from training data.

class (the absence of emotion expression) is the largest class in the dataset. Then *Tension* is the second most represented class which relates to the chosen source of raw audio (Reality TV show and TV documentary). Overall, the least represented class is *Sadness* with 10%, hence the dataset remains quite balanced and each classes has an acceptable number of support samples.

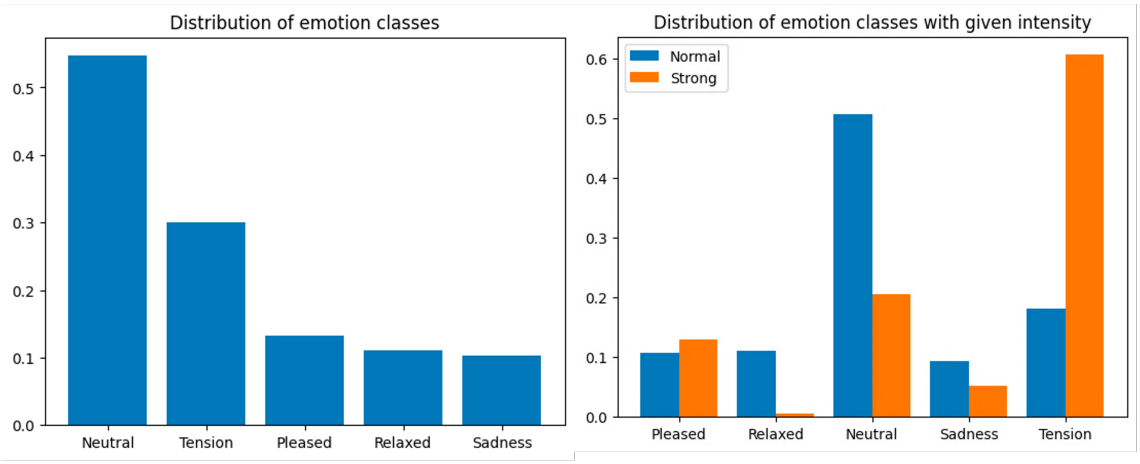

Figure 4.2: Distribution of classes within the cleaned dataset.

Morevover, when looking at the distribution of classes with given intensity, *Strong* intensity is over-represented within *Tension* classes while under represented in *Relaxed* and *Sadness* which is expected considering that intensity label is mainly linked to Arousal dimension as explained in Theory part. Therefore, intensity class seem to be a meaningful information on the tonality of sample : *Strong* intensity is most likely related to high arousal emotion subspaces *Pleased* and *Tension*.

Finally, the Figure 4.4 shows the distribution in terms of Valence and Arousal scale of the regression dataset. Neutral part (the zone around $(0.5, 0.5)$) is the most

dense and most of the 2D space is covered by the dataset.

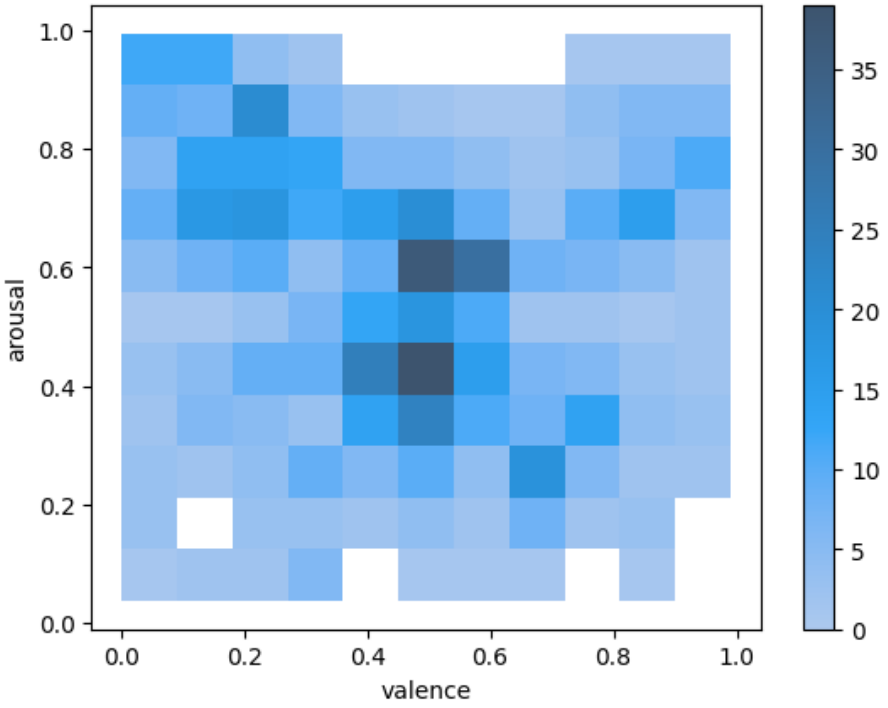

Figure 4.3: Heatmap of distribution in Valence Arousal set (as number of samples).

4.1.2 Classification

Here main results obtained through the training of multiple models and approach are presented. Complete results including the monitoring of scores with different dataset size is available in Appendix.

Training of various models

As described in Methods chapter, several approach and models have been tested during the project. Table 4.2 presents the global scores obtained with the different models trained.

First results obtained by training a multiclass classifier based on Wav2vec2 architecture lead to very poor results $f1-macro = 0.28$. As a consequence, the dataset has been largely cleaned which allow to significantly improve the results. Using these four main key points have been identified to improve model's performance :

1. Cleaning the dataset by removing very short and long samples as well as samples containing unclear emotional content.
2. Switching from a multiclass to a multilabel classification problem (mainly by changing loss function).

Table 4.2: Summary of main results of trained emotion models.

Model	Dataset	f1-micro	f1-macro
Wav2Vec2-Multiclass	Uncleaned	0.34	0.28
Wav2Vec2-Multilabel	**Cleaned+Augmentation**	**0.64**	**0.47**
Whisper L-Multilabel	Cleaned	0.47	0.32
Wav2Vec2-Multitask	Cleaned+Augmentation	0.54	0.36
Wav2Vec2/Bert-Multilabel	Cleaned+Augmentation	0.60	0.43

3. Perform data augmentation by adding third party datasets to the training data.
4. Increase the size of the dataset.

From the results obtained, the best model so far is the wav2vec2 multilabel classifier with $f1 - macro = 0.47$. However, other approach tested, especially multimodal approach with wav2vec2 and bert (audio + text) seems very promising with similar performance. For specific commercial use case, this approach might beat the simple wav2vec2 multilabel classifier. Based on litterature, these more complex alternative approaches might outperform the base approach with a larger high quality dataset [10, 53, 42]. Moreover, results obtained with Whisper-based model were not satisfying but can be explained by coding and compatibility problems that are further discussed in Discussion chapter.

Finally, Table 4.3 presents score by class with the *Wav2Vec2 - Multilabel* classifier which was chosen as the final reference model. Every class are recognized and the f1-score of each class is in correlation with the distribution of classes among the dataset.

Table 4.3: F1-score by class for *Wav2Vec2 - Multilabel* trained on cleaned data and augmentation set.

Class	f1-score
Pleased	0.29
Relaxed	0.36
Neutral	0.78
Sad	0.28
Tension	0.65

Model evaluation against pre-trained baseline

Most of the model publicly available for SER are trained on English research sets (such as the IEMOCAP [43] or the RAVDESS datasets [54]) and thus give poor results on other languages test sets. One way to evaluate the multilabel wav2vec2 model was to test it against a benchmark english pretrained model with a similar architecture. Scores on French test set obtained with both models are reported in Table 4.4

Table 4.4: Test results of english trained benchmark model against our multilabel wav2vec2-based model.

	Benchmark model	**Our Wav2vec2 Multilabel model**
F1-micro	0.41	0.56
F1-macro	0.31	0.45
F1-score by class		
Pleased	0.07	0.35
Relaxed	0.18	0.32
Neutral	0.65	0.72
Sad	0.21	0.27
Tension	0.43	0.56

The model trained on French data performs way better than English benchmark model especially when looking at f1-macro score. However, those results needs to be considered carefully since the major difference is due to low f1-score obtained on *Please* and *Relaxed* classes which could be explained by major difference between the definition of those emotions. Overall, our multilabel model can still be consider as better than the English benchmark model when tested on French conversational content.

Model testing with research dataset (RAVDESS)

In order to test if the model is robust enough to perform on other data than French conversational content included in the dataset, the model is tested on RAVDESS [54] which contains same classes as the French dataset. Results are reported in Table 4.5.

Table 4.5: f1-score of our wav2vec2 multilabel model tested on Ravdess dataset;

Class	**F1-score**	**Support**
Pleased	0.00	192
Relaxed	0.43	192
Neutral	0.33	96
Sad	0.17	192
Tension	0.76	192

On the one hand, the model is really bad at recognizing *Pleased* emotion. This could be explained again by a difference in the definition of the emotion or by linguistic differences in the expression of such positive emotion (cultural bias). On the other hand, the model performs quite well on *Relaxed* and *Tension* classes when compared to the results on the French test set.

4.1.3 Regression model

For the regression task, the training dataset was smaller than for the classification. However, english pretrained models exist and such a model can be finetuned on a

French dataset to get acceptable results. The original pre-trained model is designed for regression in a 3-dimensional space : Valence-Arousal-Dominance. The dataset is labeled in a 2-dimensional space thus the model was trained considering only Valence and Arousal dimensions. Especially, the loss function has been set to compute loss only on Valence and Arousal. This leads to a correlation coefficient $R^2 = 0.576$. Considering the size of quantitatively labeled dataset and the complexity of predicting emotions in a quantitative way, such a R^2 can be considered as quite satisfying. It suggests that the model is already able to extract information in a quantitative way and somehow reproduce the distribution of available training data. Such a model can be used to get a quantitative approach of the SER and allows to get statistical distribution of emotional content on long conversational content as shown in Figure 4.4. This is an example of possible output to evaluate global emotional content in long audio. The heatmap relates directly to the 2-dimensional emotional space and gives insights on the tonality expressed during a conversation.

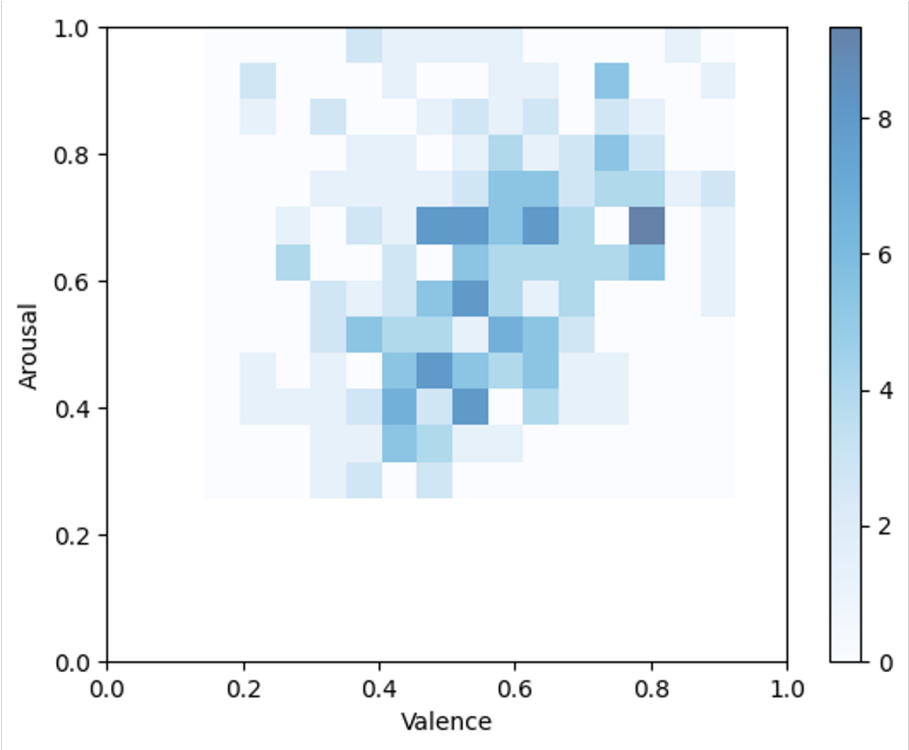

Figure 4.4: Example of a Valence-Arousal heatmap on long conversational content (10 minutes).

4.2 Speaker Diarization

This section presents the results obtained on speaker diarization. The objective was to test and evaluate available frameworks, optimize parameters on a chosen framework and perform an in-depth study of requirements and possibility for commercial application.

4.2.1 Framework comparison

Test case 1 - 2 speakers, high quality recording

The first test case as described in methods chapter is a 30 minutes TV interview with 2 speakers. This test case is simple as there is a limited number of clusters with balanced size and the audio is recorded professionally. Here, the general behavior of the models are tested as well as their performance. Table 4.6 give the number of segments in reference file and in model's prediction.

Table 4.6: Number of segments - Reference segmentation and model.

Model	Number of predicted segments
Reference labeling	251
Pyannote	126
Nemo	1209

There is a significant difference in the behavior of the two models when used with default parameters as described in Table 4.6. While pyannote is way more faster to diarize a 30 minutes audio file than Nemo pipeline (2 vs 8 minutes), it also generates 10 times less audio segments. Hence, the granularity of each of the two models is different.

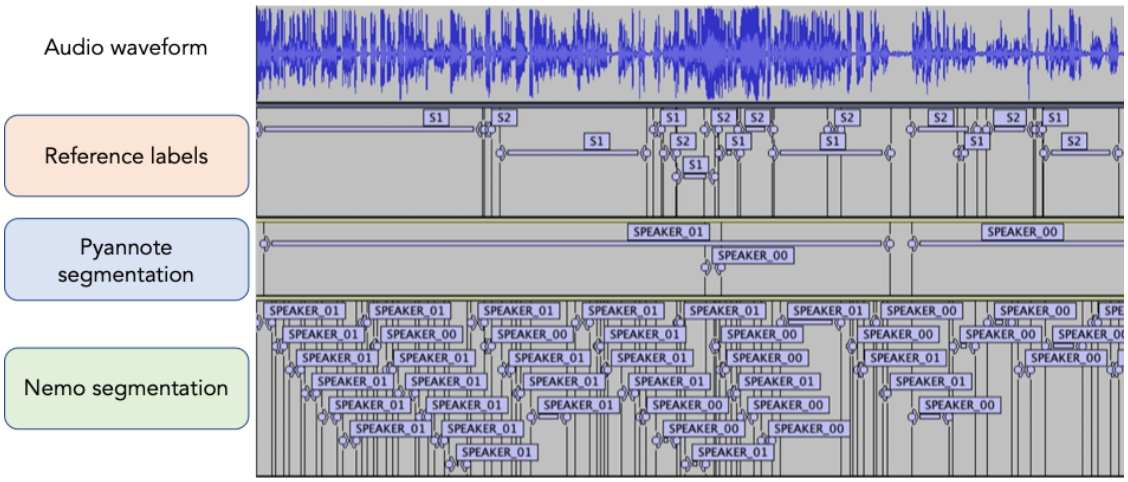

Figure 4.5: Results of diarization for the simple test case (Screen capture from Audacity interface).

This difference can be qualitatively observed in Figure 4.5. Pyannote is more likely to create long segments, merging several sentences and creating overlapping segments while Nemo detects voices with way finer granularity detecting only audio segments with linguistic content and deleting small non-speech segments (mouth noise, breathing, etc...). This has been checked by taking results from Nemo diarization and concatenating non-speech segments in a new audio file. Then, this

audio file has been labeled to detect missed speech (voice activity with actual linguistic content). Actually, there are 8% of linguistic content in predicted non-speech segments. Therefore, in spite of the difference in number of segments between Reference and Nemo's prediction, the error rate calculated should be taken carefully considering that Nemo's VAD model detects speech segments with a very fine granularity.

Then, one can compute metrics as described in Theory chapter to evaluate performance of the model. Error rates are computed against reference labeling and reported in Table 4.6. Results are reported in Table 4.7. As expected, Pyannote achieves quite low error rate (especially DER) while Nemo with default parameters has very high error rates. However, the major contribution of DER ($DER = MS + FA + SC$) is the Missed Speech contribution MS which, as mentionned above, actually contain only 8% of actual linguistic content. When optimizing parameters of Nemo's VAD model (mainly window size and shift lenght) one can change the granularity of voice detection and generate a segmentation similar to the reference file (last row in 4.7)

Table 4.7: Diarization score on the simple test case (2 speakers).

Model	DER	CDER	JER	BER	SER	MS	FA	SC
Pyannote	0.10	0.14	0.14	0.18	0.21	0.01	0.06	0.03
Nemo default parameters	0.37	0.32	0.35	0.44	0.60	0.36	0.01	0.01
Nemo optimized VAD	0.11	0.16	0.11	0.15	0.20	0.04	0.06	0.01

Since the reference segmentation was labeled by a human, it considers segments with a linguistic aspect : A segment is, generally, a complete sentence without any noticeable pause in the discours. Hence, when listened to, isolated segments keep a clear and undertstandable linguistic content which is critical to be used with Speech-to-Text or SER models.

Test case 2 - 7 speakers, medium quality recording

For the second test case, Nemo's VAD parameters are kept to get larger segments similar to reference files. This test case is more complex and evaluate how the model performs with lower quality audio and more clusters. A sample of diarization results is represented in Figure 4.6. Here Nemo provides a segmentation closer to the reference than Pyannote segmentation. Then, the error rates is measured and reported in Table 4.8

Nemo seems to perform way better than pyannote on this test case. Especially, DER and BER are way lower. Moreover, when tuning embedding model's hyperparameters from Nemo, one can obtain even better results. In addition to the lower error rate obtained with Nemo on this test case, Nemo's pipeline is more flexible

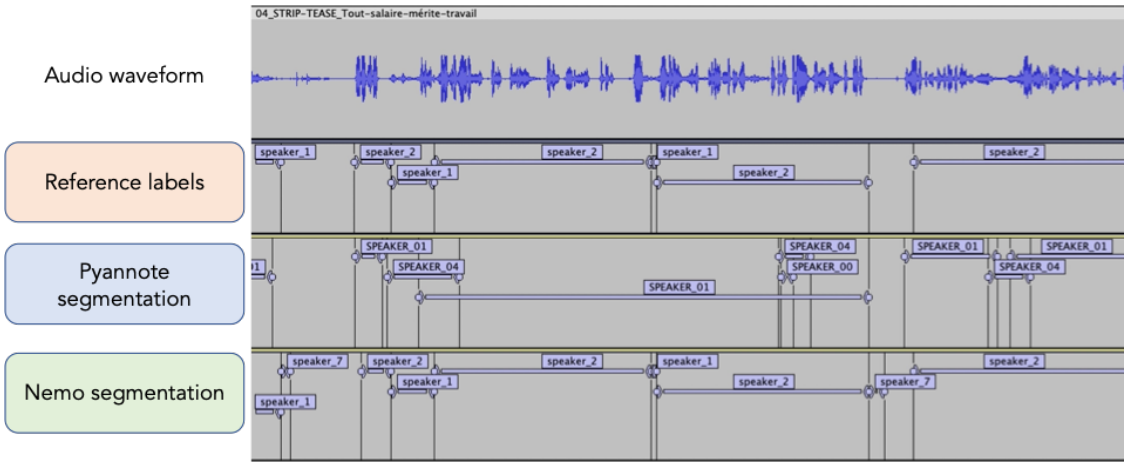

Figure 4.6: Results of diarization for the complex test case (Screen capture from Audacity interface).

Table 4.8: Diarization score on the simple test case (7 speakers).

Model	DER	CDER	JER	BER	SER	MS	FA	SC
Pyannote 7 clusters specified	0.49	0.80	0.63	0.74	0.45	0.07	0.23	0.19
Nemo no cluster nb.	0.17	0.84	0.14	0.24	0.18	0.08	0.07	0.02
Nemo optimized param.	0.12	-	-	0.15	-	-	-	-

and hyperparameters can be easily tuned through a YAML file. Therefore, Nemo was chosen as the reference framework, hence, following inference time measures and hardware dimensioning was done only for Nemo Framework.

4.2.2 Inference time and hardware resource dimensioning

All the following results are obtained only for Nemo framework

Sample rate

Input format has a large influence on computing time and performance of the model. Nemo's models process audio at $f_s = 16,000$ Hz according to the documentation[1]. When using complete pipeline as an end-to-end diarization tool, one can provide audio at any sample rate. However, it has been noticed that providing audio files already sampled at 16,000Hz reduce by around 85% inference time.

Batch size

Figure 4.7 shows the influence of batch size on computing time for 2 cases : a 30 minutes conversation and a 2.5 hours conversation. From theses results, one can

[1]https://docs.nvidia.com/deeplearning/nemo

define an optimal batch size that can be adapted with GPU memory available. To get significant improvement in computing time with resonnable GPU size, one can choose a batch size between 4 and 16 above this batch size there is no significant improvement.

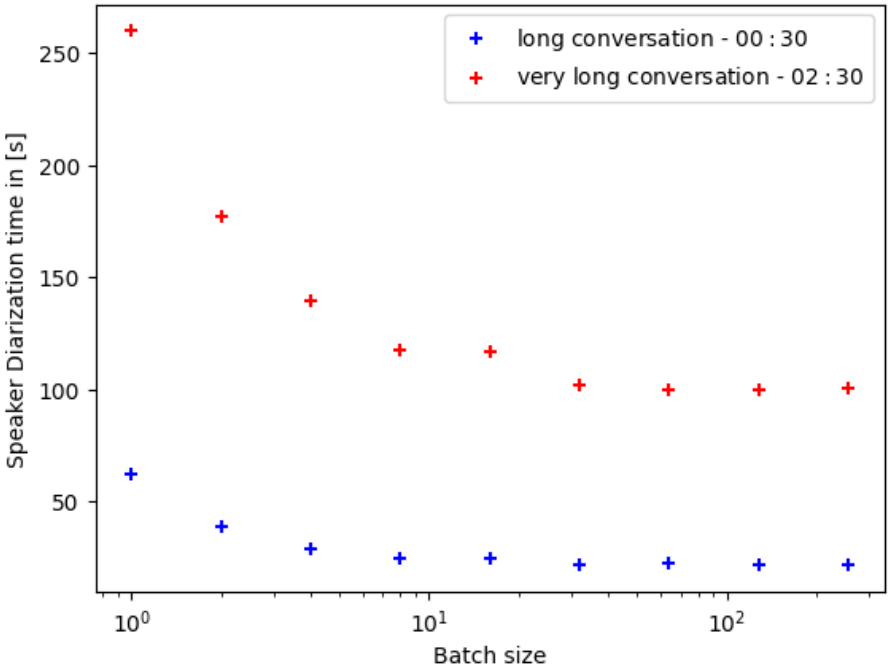

Figure 4.7: Speaker diarization time vs Batch size.

GPU memory

Then, the required GPU memory can be monitored, results are reported in Figure 4.8. The GPU memory reserved by our diarization pipeline is computed at every second and plotted. The main observation here is that there is a minimum of memory reserved ($1.5Go$) by the pipeline for VAD model. Then, the larger the batch size, the larger the memory reserved. These results confirms that having a batch size of at most 32 is convenient and limits hardware requirements ($< 4Go$) while keeping low inference time.

Multiscale embedding

Nvidia clustering method involves multiscale clustering MSDD. This method relies on several consecutive embeddings which are weighted and used for the clustering step. The optimal number of embeddings for typical use case (such as our 2 test cases) is 5 [51]. This has been checked through several test cases with a minimum of DER for 5 embeddings equally weighted. However, one should also notice that the influence of the number of embedding seem to vary with the sample rate. In

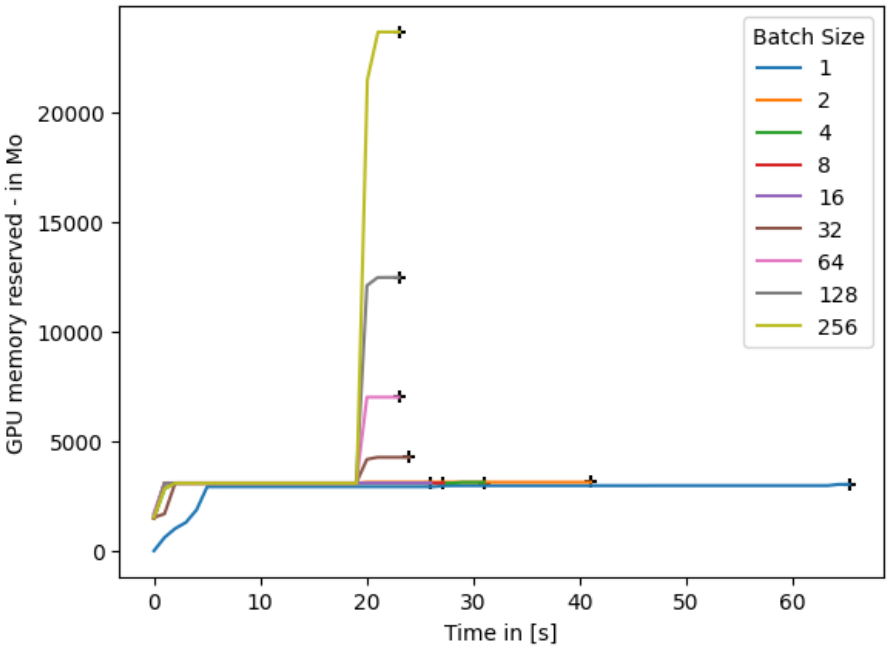

Figure 4.8: GPU memory reserved during speaker diarization for different batch sizes.

spite of an in depth study of Nemo's source code[2], the cause of this observation has not been found. In addition, for the given test, case it seems that there are no significant differences when changing the number of embedding (from 1 to 8). Thus, 1 embedding can be sufficient when trying to minimize inference time and 5 embeddings can be used to obtain slightly lower error rates.

Recommended parameters

To sum up, Table 4.9 provides recommended value for some parameters based on all the results obtained through the testing and evaluation of Nemo's pipeline.

4.3 Multi-task conversational processing tool

This sections presents the final web application used to showcase the complete multitask pipeline along with results on a real test case based on the recording of a weekly team meeting of the company.
The trained models and optimized pipelines are implemented in a complete data pipeline taking audio file as input and generating predictions output for multiple tasks. The frontend part of the web application (see Figure **??**) is developed with Gradio library and the back-end part use pipelines and datasets from Nemo and Transformers libraries.

[2]https://github.com/NVIDIA/NeMo

Table 4.9: Recommended parameters for optimal error rate with low inference time.

Parameter	Value
General	
Input sample rate (Hz)	16 000
Batch size	4 to 32
VAD	
Window size (s)	1.0
Shift length (s)	0.08
Embedding	
Number of embeddings	1 or 5
if 1 use only the larger value for window and shift	
Window size (s)	[1.5, 1.25, 1.0, 0.75, 0.5, 0.25]
Shift length (s)	[0.75, 0.625, 0.5, 0.375, 0.25]

Input panel - Left

The application is quite simple, the left panel provides 2 tabs for audio input : one to directly record audio from local microphone and the second to upload audio file. The left panel also come with a set of menue and checkboxes for pipeline settings. Especially, one can switch on/off each of implemented models, choose the transcription language in French (transcribe mode) or English (translate mode), choose a SER model with different output mapping and version of the model.

Output panel - Right

The output panel, comes with the transcription of the conversation in the form of a table (i.e a Pandas DataFrame in Python) providing timestamps, speaker labels, text transcription, emotion recognition outputs. In addition, a zero-shot topic recognition model predict potential topic within the conversation. This prediction is based on the whole text transcription of the conversation.

Screen-captures of the front end user interface of the application are available as appendix.

5
Discussion

This chapter addresses discussion of key results in further details. It highlights main potential issues, limits as well as assets of the developed methods and results.

5.1 Results overview

The final outcome of this project is an improved multitask speech processing tool that performs speaker diarization, text transcription, and emotion recognition. It can be easily extended with zero-shot topic classification and various other NLP and audio processing tasks. This tool is meant as a proof of concept and a demonstration for potential commercial development in the near future.

While the results seem quite promising, there is a huge gap between an R&D demonstration tool and a potential commercial tool. Firstly, the quality of models needs to be improved, or at least one should be able to accurately assess errors and mitigate them better. Increasing size of datasets is critical as well as being able to get high quality data, this issue is discussed further in following sections. Secondly, the pipeline developed needs to be significantly improved in terms of computing time and efficiency of the code. The complete pipeline performing the different tasks includes several large neural networks : 317 millions of parameters for SER model and 1550 millions for Whisper model (ASR task) the two largest models. Not only this requires very expensive hardware for storage and computing (GPU, servers) but one should carefully think about the use of such a tool and the cost of deploying huge machine learning based pipeline in terms of cost but also ethics and ecology.

Nevertheless, the great development of automatic speech recognition (text-to-speech, SER, speaker diarization) allied with new Large Language Models such as ChatGPT or LLama could lead to major innovation in the field of human-computer interactions in many research and industrial fields: healthcare, entertainment, customer services, art, etc.

5.2 Context specific applications

The final pipeline contains models trained for general-purpose analysis. On the one hand, it makes the approach very versatile and, as shown in the results, efficient in a general context. This is a great asset for showcasing the possibilities of the models for a large variety of potential business cases with acceptable results. Moreover,

training models on general-purpose data at first might improve results when further fine-tuning them. Here, our approach provides models with already acceptable performance. On the other hand, while the results are promising, they might not be good enough to scale the method for a business case. Therefore, the models need to be fine-tuned for a specific use case and require a context-specific dataset as in [10, 42].

Some examples of limitations of our implemented models are:

- The SER model can predict emotions in a large variety of contexts with a reduced number of classes, thus the same *emotion* can be expressed in very different ways depending on the context of the conversation.
- Speaker diarization hyperparameters can be tuned depending on the context (large number of speakers, phone call, outdoor conversation, etc.), and there are different VAD and speaker embedding models that can be used depending on the characteristics of the audio. Therefore, the context needs to be reviewed carefully to optimize diarization.
- Speech-to-text models such as Whisper are general ASR models and thus lack accuracy with audio containing technical or context-specific vocabulary.

To sum up, the proposed pipeline, while promising and being a good demonstration, needs to be further fine-tuned with a context-specific dataset for a given business case. This also relates to the fact that language/voice models training relies on a large amount of data, and hence, providing specific examples will significantly help in improving results for given business cases.

5.3 End-to-end approach with real life data

The final demonstration web application provides an end-to-end pipeline to process conversational content and perform various tasks. In addition to the implemented tasks, one can easily use this work to implement a wide variety of tasks involving the processing of voices, text (from the transcription), or both at the same time, as demonstrated with the multimodal approach of SER. This offers a lot of flexibility, allowing the use of this proof of concept as a demonstration for various commercial purposes.

Our approach focuses on training and optimizing models for real-life data in the French language. The creation of a French dataset of real-life conversational content addresses the lack of such publicly available data. Indeed, most available datasets and, hence, pre-trained models are trained on English data. In addition, these models are only trained and tested on benchmark sets, which makes comparison with our approach quite difficult. However, it seems that fine-tuning pre-trained models on real-life data makes the resulting models more robust and more accurate for real-life applications and, hence, more suitable for commercial deployment. Then, the major challenge to improve the proposed method and achieve high-quality results on the SER problem is the data collection and labeling.

ASR-related tasks as well as NLP are known to be highly data-centric problems.

SOTA approach for many problems includes training neural networks on massive amounts of data. In the context of fine-tuning SER models with real-life data, one must be able to provide a very large dataset to expect high accuracy (or F1-score). In this thesis, the size of the final dataset can be considered quite small, and one must consider building a larger audio corpus to scale up the pipeline for commercial deployment. Our dataset contains only 1.5 hours of conversational content, and we expect that creating a 5- to 10-hour high-quality dataset could significantly improve results and even allow alternative approaches such as multitask (emotion and intensity) or multimodal (audio and text) to beat simple multilabel approaches [10, 42].

5.4 Labelling uncertainty and limits of the model

Data centric problems relates not only to the quantity of training data but also on their quality. Especially, SER tasks rely on highly subjective considerations from human emotional intelligence. Emotion expression and language are unique to humankind and the development of emotion and language related machine learning is often rightly criticized. Tech companies and research organizations hold a great responsibility for developing unbiased, accurate and transparent models and for tackling privacy and ethical concerns.

This starts by defining and implementing a precise protocol to build emotion dataset. Especially, in this project, the dataset was labeled by a single person which makes it inevitably biased. To mitigate this issue data must be labeled and reviewed by several persons. This is a sensitive issue since it is very difficult to assess to what extent such data can be biased, however going through multiple labeling (with several labelers) and statistically defines *unbiased* labels for the set should help to build datasets and trained model that are less biased.

Moreover, benchmark models (IEMOCAP, RAVDESS) are generally build with prior knowledge of targeted emotion : Actors are performing predefined emotions. On the one hand, this reduce the risks of mislabeled data and should somehow reduce bias on emotion interpretation. On the other hand, this lead to very stereotyped emotion expression which, as highlighted with our results, make the model less robust when it comes to real life data.

Overall, one must carefully think about limitations and drawbacks when developing emotion recognition models. Here, our approach focuses on French language, this induces a cultural and linguistic bias and makes the resulting model suitable for French conversational content. Furthermore, the reference trained model is a multilabel classifier (or multiclass) with 5 classes defining subspaces of the Valence-Arousal 2D space [38]. These generic classes are convenient, easily understandable and helps map emotions in a reduced number of categories. However, it makes the use of the model very shallow with a wide range of emotion for a single class (e.g. tension relates on both deep anger and simple unsatisfaction). To tackle this issue, we propose two possible solutions. The regression model outputs floating values on

Valence and Arousal scales which allows to handle the complete range of emotions in the Valence-Arousal space, however it is less directly understandable with no prior knowledge on the definition of this specific emotion space. Alternatively, one can use intensity label in combination to create a more complete mapping of emotion as described in 5.1.

Table 5.1: Mapping example based on combination of 5-class emotion and binary intensity prediction.

Intensity level	**Pleased**	**Relaxed**	**Neutral**	**Sadness**	**Tension**
Normal	Happy	Peaceful	Neutral	Depressed	Unsatisfaction
Strong	Excited	Content	Vehement	Crying	Anger

Increasing the granularity of the emotion space (i.e. having more emotion classes) is a way to improve interpretability of the model. Nevertheless, depending on the use case, it might be better to actually reduce granularity by mapping several classes to larger classes. For example, by reducing the 5-classes model to a 3-classes problem (Negative, Neutral, Positive emotion) could help building more accurate and interpretable model for certain purpose. For instance, in the context of call centers, providing the operator an estimation of the *Positivity / Negativity* of the current conversation with a client could be more helpful than very precise emotion prediction.

Finally, our multilabel approach tries to provide a versatile and flexible model. Hence, the model is meant as a tradeoff between the granularity of emotion prediction (up to 10 combination of tonality/intensity which can be combined to quantitative prediction) and good performances to build highly accurate and efficient prediction for coarser granularity, closer to binary classification.

5.5 Real time tool

The conversational analysis tool developed in this project might be used for several business applications, and the demonstration should showcase the possibilities offered by the proposed pipeline. However, in many expected use cases, such as for call center operators, one might want to have a tool able to evaluate the content of a conversation in real time. For now, models involved in the current pipeline require large hardware resources and are not able to perform the inference in real time. Specifically, consecutive steps of speaker diarization with Nemo and speech-to-text transcription with Whisper take around 10 seconds for 30 seconds of audio. Thus, further development needs to be considered to improve the pipeline, making it more efficient and able to take streaming audio as input and output real-time diarized transcription along with other tasks (emotion, topic, summary, etc).

Since ASR is a very active field of research, new models or alternatively new implementations of models might help to achieve reduced computing time. Especially,

a new implementation of Whisper with JAX (a deep learning framework similar to PyTorch) [55], recently released, makes the transcription of audio 70 times faster. Alternatively, when looking for high-speed computing, one might want to implement models and pipeline in C++ to highly increase performance. However, this requires a lot of work to re-implement the pipeline in a completely different language.

Overall, the main asset of our end-to-end approach might be its flexibility. Indeed, with a few efforts, one can update and adapt models to specific needs: reducing model size to increase performance, perform parallel computing inference if a large GPU is available, adding new models for specific tasks, etc...

6
Conclusion

This master thesis is the final outcome of a Research & Development project at La Javaness within the Unstructured Data Team. Based on previous research and proof of concept, the objective was to contribute to the development of a robust multitask speech analysis tool for French conversational content for real-life data. The project mainly focused on improving Speech Emotion Recognition model and leveraging state-of-the-art Speaker Diarization frameworks, but it also led to a large exploration of deep learning for speech processing. In the end, it aims to showcase the feasibility of potential commercial use.

Through a simple 5-class approach to the emotion classification problem, we show that fine-tuning a wav2vec2 model encoder with a simple dense network as a classification head is very effective in predicting emotions in conversational content. Especially, when inferring on casual conversational content, our model beats similar SOTA models trained on English research dataset while still achieving good results on the English benchmark set RAVDESS. Indeed, our own dataset contains a large plurality of speakers expressing a wide range of emotions which allows us to effectively fine-tune the model and make it robust enough to predict emotions in any context with good accuracy.

In addition, the proposed multitask and multimodal architectures seem very promising for context-specific applications (i.e potential business cases) assuming the availability of a large high-quality dataset. Actually, by leveraging the speech-to-text capability of Whisper, the multimodal approach is able to combine voice analysis along with purely linguistic features from the transcribed text.

Speech emotion recognition remains a highly data-centric task requiring very large dataset. Therefore, while the possibility of developing efficient speech emotion recognition with deep learning is highlighted through our results, the developed models still lack robustness to predict emotions in any context with any voices. Especially, commercially deployed models should be inclusive by being able to accurately detect non-stereotyped emotion expressions, including variety of accents, specific vocabulary and non normative social interactions.

The speaker diarization pipeline is a critical stage when processing conversational content. We propose a comparison of two current SOTA frameworks that come with complete flexible pipelines: pyannote and Nemo (Nvidia). Then, we explore Nemo's capability and try to optimize models' performance for two test cases and propose

optimized parameters in terms of computing time and low error rate.

The developed SER model and optimized speaker diarization pipeline are then implemented in a demonstration web application showcasing the capability of a complete multitask pipeline for conversational speech processing. The demonstration is designed with high flexibility, allowing leveraging all kinds of SOTA models for various tasks involving audio or text inputs.

Overall, results are very promising but still need to be improved with more high-quality data on specific use cases. This project demonstrates a wide range of possibilities for multitask speech processing for real-life conversation. Nevertheless, one must carefully keep in mind ethical and ecological concerns before potential scaling and deployment of such a tool.

Bibliography

[1] Yann LeCun, Y. Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–44, 05 2015.

[2] Ken H Davis, R Biddulph, and Stephen Balashek. Automatic recognition of spoken digits. *The Journal of the Acoustical Society of America*, 24(6):637–642, 1952.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[4] Xavier Amatriain. Transformer models: an introduction and catalog. *arXiv preprint arXiv:2302.07730*, 2023.

[5] Hanan Aldarmaki, Asad Ullah, Sreepratha Ram, and Nazar Zaki. Unsupervised automatic speech recognition: A review. *Speech Communication*, 139:76–91, 2022.

[6] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. *arXiv preprint arXiv:2212.04356*, 2022.

[7] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

[8] Tae Jin Park, Naoyuki Kanda, Dimitrios Dimitriadis, Kyu J. Han, Shinji Watanabe, and Shrikanth Narayanan. A review of speaker diarization: Recent advances with deep learning. *Computer Speech Language*, 72:101317, 2022.

[9] G. Zweig, O. Siohan, G. Saon, B. Ramabhadran, D. Povey, L. Mangu, and B. Kingsbury. Automated quality monitoring in the call center with asr and maximum entropy. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 1, pages I–I, 2006.

[10] Theo Deschamps-Berger, Lori Lamel, and Laurence Devillers. Investigating transformer encoders and fusion strategies for speech emotion recognition in emergency call center conversations. In *Companion Publication of the 2022 International Conference on Multimodal Interaction*, pages 144–153, 2022.

[11] Cecko Robert, Jamroży Jerzy, Jęśko Waldemar, Kuśmierek Ewa, Lange Marek, and Owsianny Mariusz. Automatic speech recognition and its application to media monitoring. *CMST*, 27(2):41–55, 2021.

[12] Tae youn Ahn and Sangmin-Michelle Lee. User experience of a mobile speaking application with automatic speech recognition for EFL learning. *British Journal of Educational Technology*, 47(4):778–786, September 2015.

[13] Philippe Gournay, Olivier Lahaie, and Roch Lefebvre. A canadian french emotional speech dataset, 2018.

[14] Dirk Helbing. *Societal, economic, ethical and legal challenges of the digital revolution: from big data to deep learning, artificial intelligence, and manipulative technologies*. Springer, 2019.

[15] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*, 2019.

[16] Taiba Majid Wani, Teddy Surya Gunawan, Syed Asif Ahmad Qadri, Mira Kartiwi, and Eliathamby Ambikairajah. A comprehensive review of speech emotion recognition systems. *IEEE Access*, 9:47795–47814, 2021.

[17] Hervé Bredin, Ruiqing Yin, Juan Manuel Coria, Gregory Gelly, Pavel Korshunov, Marvin Lavechin, Diego Fustes, Hadrien Titeux, Wassim Bouaziz, and Marie-Philippe Gill. pyannote.audio: neural building blocks for speaker diarization. In *ICASSP 2020, IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2020.

[18] Hervé Bredin and Antoine Laurent. End-to-end speaker segmentation for overlap-aware resegmentation. In *Proc. Interspeech 2021*, 2021.

[19] C.E. Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21, jan 1949.

[20] Maria Labied and Abdessamad Belangour. Automatic speech recognition features extraction techniques: A multi-criteria comparison. *International Journal of Advanced Computer Science and Applications*, 12, 01 2021.

[21] Bernhard Mehlig. *Machine Learning with Neural Networks*. Cambridge University Press, oct 2021.

[22] Iqbal H. Sarker. Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science*, 2(6), August 2021.

[23] Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object recognition with gradient-based learning. In *Shape, Contour and Grouping in Computer Vision*, pages 319–345. Springer Berlin Heidelberg, 1999.

[24] Taiwo Oladipupo Ayodele. Types of machine learning algorithms. *New advances in machine learning*, 3:19–48, 2010.

[25] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[26] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. 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)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[28] Peng Peng and Jiugen Wang. How to fine-tune deep neural networks in few-shot learning? *arXiv preprint arXiv:2012.00204*, 2020.

[29] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[30] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

[31] Hervé Bredin. pyannote.metrics: a toolkit for reproducible evaluation, diagnostic, and error analysis of speaker diarization systems. In *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association*, Stockholm, Sweden, August 2017.

[32] Gaofeng Cheng, Yifan Chen, Runyan Yang, Qingxuan Li, Zehui Yang, Lingxuan Ye, Pengyuan Zhang, Qingqing Zhang, Lei Xie, Yanmin Qian, et al. The conversational short-phrase speaker diarization (cssd) task: Dataset, evaluation metric and baselines. *arXiv preprint arXiv:2208.08042*, 2022.

[33] Tao Liu and Kai Yu. Ber: Balanced error rate for speaker diarization. *arXiv preprint arXiv:2211.04304*, 2022.

[34] Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[35] Abubakar Abid, Ali Abdalla, Ali Abid, Dawood Khan, Abdulrahman Alfozan, and James Zou. Gradio: Hassle-free sharing and testing of ml models in the wild. *arXiv preprint arXiv:1906.02569*, 2019.

[36] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[37] Rüdiger Wirth and Jochen Hipp. Crisp-dm: Towards a standard process model for data mining. In *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*, volume 1, pages 29–39. Manchester, 2000.

[38] James A Russell. A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161, 1980.

[39] yi-hsuan Yang and Homer Chen. Machine recognition of music emotion: A review. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3, 05 2012.

[40] Kai Sun, Jun-qing Yu, Yue Huang, and Xiaoqiang Hu. An improved valence-arousal emotion space for video affective content representation and recognition. pages 566 – 569, 08 2009.

[41] Oana Mitruț, Gabriela Moise, Livia Petrescu, Alin Moldoveanu, Marius Leordeanu, and Florica Moldoveanu. Emotion classification based on biophysical signals and machine learning techniques. *Symmetry*, 12:21, 12 2019.

[42] Pengcheng Yue, Leyuan Qu, Shukai Zheng, and Taihao Li. Multi-task learning for speech emotion and emotion intensity recognition. In *2022 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1232–1237, 2022.

[43] Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N. Chang, Sungbok Lee, and Shrikanth S. Narayanan. IEMOCAP: interactive emotional dyadic motion capture database. *Language Resources and Evaluation*, 42(4):335–359, nov 2008.

[44] Oleksii Kuchaiev, Jason Li, Huyen Nguyen, Oleksii Hrinchuk, Ryan Leary, Boris Ginsburg, Samuel Kriman, Stanislav Beliaev, Vitaly Lavrukhin, Jack Cook, et al. Nemo: a toolkit for building ai applications using neural modules. *arXiv preprint arXiv:1909.09577*, 2019.

[45] Tae Jin Park, Naoyuki Kanda, Dimitrios Dimitriadis, Kyu J Han, Shinji Watanabe, and Shrikanth Narayanan. A review of speaker diarization: Recent advances with deep learning. *Computer Speech & Language*, 72:101317, 2022.

[46] Yusuke Fujita, Naoyuki Kanda, Shota Horiguchi, Yawen Xue, Kenji Nagamatsu, and Shinji Watanabe. End-to-end neural speaker diarization with self-attention. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 296–303, 2019.

[47] Mirco Ravanelli and Yoshua Bengio. Speaker recognition from raw waveform with sincnet. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 1021–1028. IEEE, 2018.

[48] Fei Jia, Somshubra Majumdar, and Boris Ginsburg. Marblenet: Deep 1d time-channel separable convolutional neural network for voice activity detection. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6818–6822. IEEE, 2021.

[49] Brecht Desplanques, Jenthe Thienpondt, and Kris Demuynck. ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification. In *Interspeech 2020*. ISCA, oct 2020.

[50] Nithin Rao Koluguri, Taejin Park, and Boris Ginsburg. Titanet: Neural model for speaker representation with 1d depth-wise separable convolutions and global

context. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8102–8106. IEEE, 2022.

[51] Tae Jin Park, Nithin Rao Koluguri, Jagadeesh Balam, and Boris Ginsburg. Multi-scale speaker diarization with dynamic scale weighting. *arXiv preprint arXiv:2203.15974*, 2022.

[52] Audacity Team. Audacity® software - the name audacity® is a registered trademark.

[53] Bagus Tris Atmaja, Kiyoaki Shirai, and Masato Akagi. Speech emotion recognition using speech feature and word embedding. In *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 519–523, 2019.

[54] Steven R. Livingstone and Frank A. Russo. The ryerson audio-visual database of emotional speech and song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in north american english. *PLOS ONE*, 13(5):e0196391, May 2018.

[55] Roy Frostig, Matthew James Johnson, and Chris Leary. Compiling machine learning programs via high-level tracing. *Systems for Machine Learning*, 4(9), 2018.

A

Appendix - Complete tables of results

Table A.1: Table of all results with the first draft of the dataset on multiclass classification task

#	Base model	Nb of rows (train/val)	Total length	f1 micro	f1 macro	f1 on each class
1	wav2vec2	1109 / 278	1.3	0.32	0.28	0 - Pleased : 0.21 1 - Relaxed : 0.22 2 - Neutral : 0.43 3 - Sad : 0.17 4 - Fear : 0.22 5 - Tension : 0.42
2	whisper medium	1109 / 278	1.3	0.32	0.17	0 - Pleased : 0.10 1 - Relaxed : 0.20 2 - Neutral : 0.49 3 - Sad : 0.00 4 - Fear : 0.00 5 - Tension : 0.27
3	wav2vec2	1992 / 499	2.1	0.32	0.25	0 - Pleased : 0.18 1 - Relaxed : 0.24 2 - Neutral : 0.48 3 - Sad : 0.11 4 - Fear : 0.11 5 - Tension : 0.38
4	whisper medium	1992 / 499	2.1	0.33	0.18	0 - Pleased : 0.05 1 - Relaxed : 0.05 2 - Neutral : 0.48 3 - Sad : 0.09 4 - Fear : 0.09 5 - Tension : 0.32
5	wav2vec2	2126/532	2.3	0.41	0.28	0 - Pleased : 0.39 1 - Relaxed : 0.14 2 - Neutral : 0.59 3 - Sad : 0.04 4 - Fear : 0.13 5 - Tension : 0.40

Table A.2: Table of all results with the first draft of the dataset on binary classification of intensity

#	Base model	Nb of rows (train/val)	Nb of audio hours	f1	Accuracy
1	wav2vec2	1109 / 278	1.3	0.71	0.83
2	whisper medium	1109 / 278	1.3	0.59	0.80
4	wav2vec2	1992 / 499	2.1	0.71	0.86
5	whisper medium	1992 / 499	2.1	0.66	0.85
6	whisper large	1992 / 499	2.1	0.70	0.85

Table A.3: Table of all results with the cleaned dataset

#	Base model	Nb of rows(train/val)	Data processing techniques	f1 micro	f1 macro	f1 on each class
1	wav2vec2	590/148	None	0.49	0.33	0 - Pleased : 0.33 1 - Relaxed : 0.00 2 - Neutral : 0.68 3 - Sad : 0.25 4 - Tension : 0.38
2	wav2vec2	843/211	None	0.44	0.34	0 - Pleased : 0.36 1 - Relaxed : 0.14 2 - Neutral : 0.58 3 - Sad : 0.23 4 - Tension : 0.36
3	wav2vec2	1102/276	Training set augmentation with French Dataset	0.55	0.37	0 - Pleased : 0.48 1 - Relaxed : 0.15 2 - Neutral : 0.70 3 - Sad : 0.0 4 - Tension : 0.49
4	wav2vec2 Multilabel	1102/276	None	0.54	0.40	0 - Pleased : 0.29 1 - Relaxed : 0.26 2 - Neutral : 0.72 3 - Sad : 0.18 4 - Tension : 0.55
5	wav2vec2 Multilabel	1315/329	None	0.56	0.39	0 - Pleased : 0.24 1 - Relaxed : 0.15 2 - Neutral : 0.74 3 - Sad : 0.21 4 - Tension : 0.59
6	wav2vec2 Multilabel	1315/329	Training set augmentation with French Dataset and Call my agent	0.58	0.39	0 - Pleased : 0.35 1 - Relaxed : 0.12 2 - Neutral : 0.73 3 - Sad : 0.16 4 - Tension : 0.58
7	wav2vec2 Multilabel	1544/386	Training set augmentation with French Dataset and Call my agent	0.64	0.47	0 - Pleased : 0.29 1 - Relaxed : 0.36 2 - Neutral : 0.78 3 - Sad : 0.28 4 - Tension : 0.65
8	Whisper encoder Multilabel	1742/436	None	0.47	0.32	0 - Pleased : 0.19 1 - Relaxed : 0.19 2 - Neutral : 0.64 3 - Sad : 0.10 4 - Tension : 0.49
9	Multimodal classifier Bert + Wav2vec2	1742/436	None	0.60	0.43	0 - Pleased : 0.38 1 - Relaxed : 0.16 2 - Neutral : 0.76 3 - Sad : 0.27 4 - Tension : 0.59

B

Appendix - Demonstration application for speech analysis

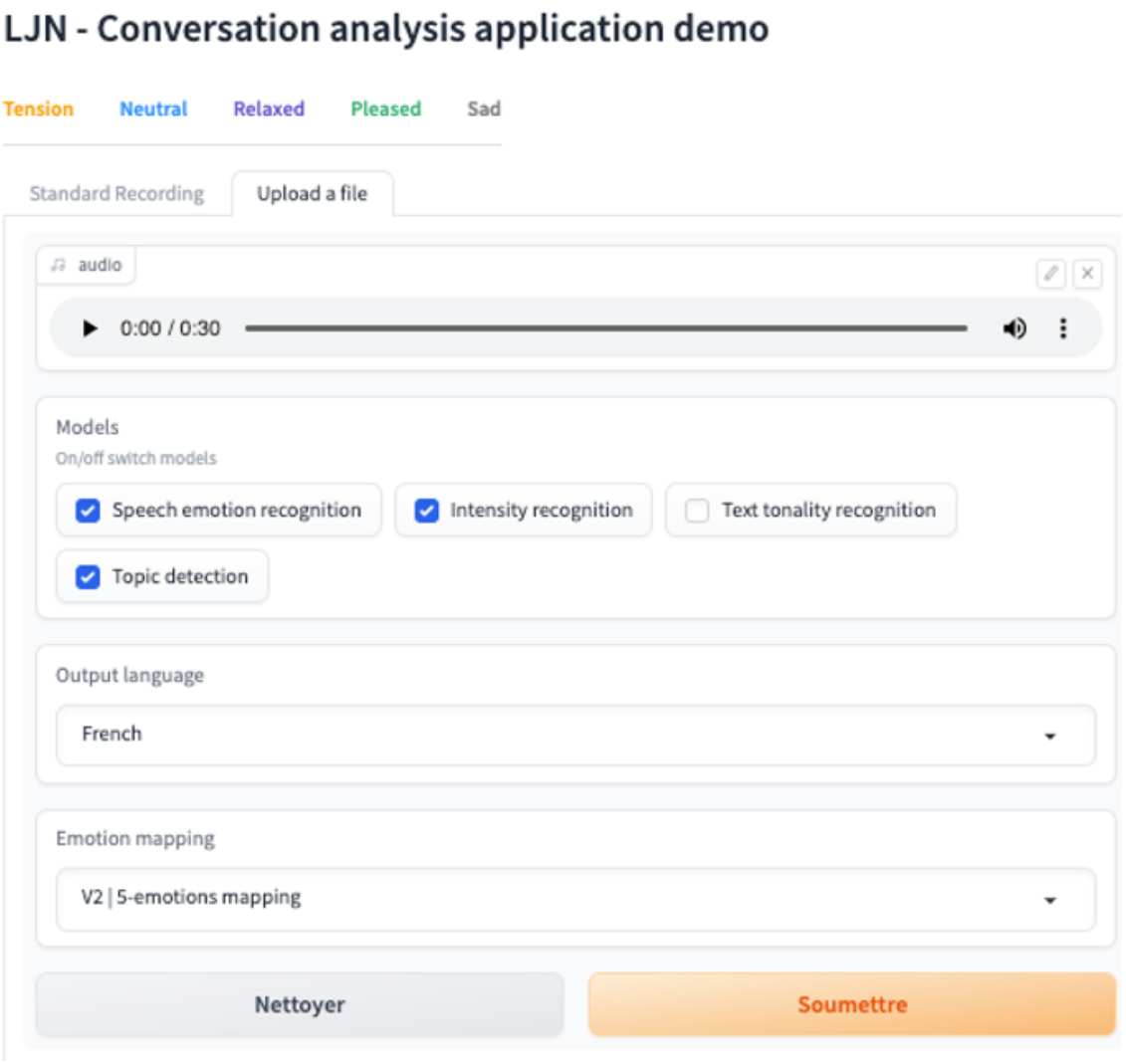

Figure B.1: Screen capture of the input panel of the demonstration web application.

Figure B.2: Screen capture of the output panel of the demonstration web application.

DEPARTMENT OF PHYSICS
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY