POLITECNICO DI TORINO

Master's Degree in Data Science and Engineering

Master's Degree Thesis

Computational Biases of Foundation Models for Speech Emotion Recognition: A Quantitative Analysis

Supervisors
Prof. RIZZO GIUSEPPE
Dott. D'ASARO FEDERICO
Dott. MÀRQUEZ VILLACIS JUAN JOSÈ
Dott.ssa FRISIELLO ANTONELLA

Candidate
ELENA DI FELICE

APRIL 2024

Summary

As Artificial Intelligent systems become more widely used in our daily lives, it's crucial to ensure not only their accuracy, but also their fairness.
In this study, I focused on assessing fairness and the possible presence of bias in systems that address the task of Speech Emotion Recognition (SER). Speech Emotion Recognition is the process of automatically detecting and understanding the emotional content conveyed through spoken language. It relies on analyzing acoustic features of the speech signal, independently of the actual linguistic content. The experiments were conducted using the only two datasets available in Italian for this task, *Emozionalmente* and *EMOVO*. I implemented the fairness metrics that are mostly used in literature (Disparate Impact, Statistical Parity, Average Odds and Equal opportunity) as well as two baselines to run the tests: a Support Vector Machine (SVM) model, considering two different methods to extract features (MFCC and MFMC), and a ResNet. Two sensitive attributes were considered for the analysis, based on the information about the subjects made available by the datasets: in the experiments carried out using *EMOVO*, only gender was considered, while in those using *Emozionalmente* I was also able to consider age. I then tested the fairness, using the same metrics, of WavLM, a new transformer based pre-trained model.
By comparing the results obtained, I was able to verify how different algorithms use the intrinsic information contained in the audios to obtain the labels, and by changing the distributions of subjects in the training datasets, I was able to verify whether and how the training data affect the output in terms of bias. Furthermore, by performing the experiments on a model that has better accuracy performance than the baselines, I was also able to draw conclusions about the dependence between bias and accuracy.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Knock Knock project

This thesis is part of the Knock Knock project (KK), which is being developed by the LINKS Foundation. The overall goal of the Knock Knock project is to develop and test an innovative method based on digital technologies for facilitating the placement phase of individuals with autism spectrum disorders in new environments, in order to help reduce the difficulties and negative experiences that are typically associated with this type of experience.
KK seeks to integrate, introduce, and experiment with a digital component in the existing process of including individuals with Autism Spectrum Disorder (ASD) at Il Margine Cooperative's new Day Activity Centre (DAC). Through digital technologies, the person will be able to experience virtual visits from his or her home (or from a place of choice) in which he or she will gradually make contact with the physical environments, the planned activities, and the social context he or she will find at the DAC. In addition to interactive systems, the project aims to develop and test an Artificial Intelligence (AI) system to recognize and label emotions experienced by participants in the experiment, with a focus on individuals on the Autism Spectrum and their families. In its first version, the AI system for emotion recognition is based on the analysis of para-verbal aspects of speech to recognize the six primary emotions outlined by Ekman [1] plus neutral state.
The purpose of this system is to contribute to the improvement of services for those who are affected by ASD, both directly and indirectly. In particular:

- Parents of people with ASD can find support in identifying the "threshold of emotional non-control", which is very subjective and can involve both positive and negative emotions. The priority need is to anticipate and mitigate behavioral crises;

- Individuals with ASD can benefit from learning to recognize or better distinguish their emotions;
- Professionals can use it as an educational tool and for monitoring, if the system will be designed to make the data it processes available and usable;
- Managers of services for families and people with DSA will be able to implement a unique and tailored transition management approach.

My thesis is part of this project, having as its goal to verify and quantify the presence of bias in the emotion recognition system. The idea is to develop a bias recognition framework that can be applied to all sensitive attributes, including being an ASD subject or not.

1.2 Emotion Recognition Task

Emotion recognition is the artificial intelligence task aimed at detecting human emotions from various sources, such as text, speech and facial expressions.
The use of technology to recognize emotions is a fairly recent practice in research, but it already has various applications. Some areas in which emotion recognizers are employed are for example call centers assessing customer satisfaction, e-learning systems, assistive robotics, security agencies, military organizations and many more [2, 3, 4]. Detecting emotions can be difficult due to their subjective nature, despite their numerous applications. There is no clear understanding of how to measure or categorize them.
Recognizing the emotion expressed given an input data is a particular example of the macro area of classification problems. Given a set of data to provide as input and a set of possible labels, we want to assign to each instance of the dataset one of the labels. Both machine learning and deep learning algorithms can be used to solve problems of this kind, as we will see in the in-depth discussion in chapter 2. Regardless of the classification model used, it has been seen that using different data modalities (text, audio, video) at the same time produces better results [5], however processing numerous forms of data requires technologies and resources that aren't always available. In this study I will focus on the task of Speech Emotion Recognition, which is the recognition of emotions using audio inputs.

1.2.1 Speech Emotion Recognition

Speech Emotion Recognition (SER) is the process of automatically detecting and understanding the emotional content conveyed through spoken language. It involves analyzing various acoustic features of the speech signal, such as pitch, intensity, rhythm, and spectral characteristics, to infer the underlying emotional state of the

speaker, independently of the actual linguistic content [6].
By ignoring language content, it is possible to create more broadly applicable models that can identify emotions in a variety of linguistic and cultural contexts. However, it's important to note that, even though semantic content is not employed directly in SER, prosody and other acoustic aspects may still be indirectly impacted by it [7]. The speaker's intonation, for example, can be influenced by word choice and the semantic context of those words, which can then have an impact on how emotionally charged the speech is perceived to be. For these reasons, in this study we decided to focus on the speech emotion recognition task using exclusively Italian language datasets, as you will see in section 4.

1.2.2 Emotion Classification

To properly implement a speech emotion recognition system, we must define and model emotions precisely. However, the definition of emotions remains a controversial topic in psychology. The range of emotions that human beings can experience is in fact very wide, and it is impractical to consider and classify them all. In the twentieth century, more than ninety definitions of emotion were proposed [8], and based on these definitions two models have become common in the speech emotion recognition task: *discrete emotional model* and *dimensional emotional model*.
Discrete emotion theory is based on Ekman's investigations of the six basic emotions: anger, disgust, fear, happiness, sadness and surprise. For this theory, all other emotions are obtained by the combination of the basic ones. We focus on these emotions because are the ones that appear to be universal across humanity [9]. But what differentiates basic emotions from other types of emotions? *A basic emotion should be discrete, have a fixed set of neural and bodily expressed components, and a fixed feeling or motivational component that has been selected for through longstanding interactions with ecologically valid stimuli* [1]. Most of the existing SER systems, including those used in this work, focus on these basic emotional categories, adding neutrality to represent states in which no emotion is being experienced.
Dimensional emotional model is an alternative model that uses a small number of latent dimensions to characterize emotions, such as valence, arousal, control, power (Figure 1.1). In this approach, emotions are related to one another in a structured way rather than being separate. Although it can be useful in categorizing a wider range of emotions, this approach has several disadvantages and is therefore not used very often: it is not intuitive, and specific training may be required to classify each emotion. In addition, some emotions, such as fear and anger, become identical, while others, like surprise, cannot be defined and may have a positive or negative valence depending on context.

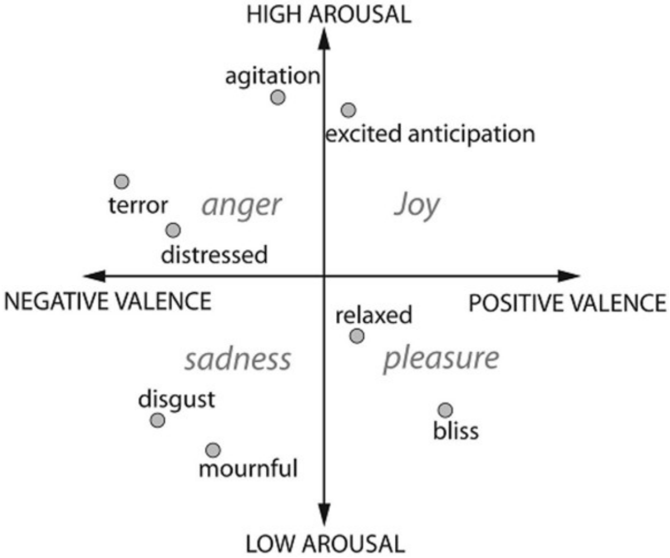

Figure 1.1: Dimensional emotional model with examples of emotions

1.2.3 Open problems and challenges

Emotions are complex and subjective experiences, which makes the emotion recognition task particularly challenging.
Since emotion expression varies widely across cultures and languages, it is difficult to create emotion recognition models that can universally interpret and respond to diverse emotional signals [7].
In addition, speech often conveys a mix of emotions simultaneously, introducing ambiguity into the recognition process.
Another problem relates to the contextual information surrounding the speech: in fact, social cues and the speaker's past experiences can help identify a particular emotion more accurately, but these data are not always available and it is not immediate to integrate them into recognition systems.
When, as in the context of this study, we are also dealing with data from ASD subjects, the difficulties increase [10, 11]. In fact, when it comes to emotions, it is said that ASD subjects typically do not express them in ways that regular people would be able to identify and comprehend. They either do not react emotionally at all or, on occasion, their emotional reactions may come across as excessive. In some cases, subjects are also non-verbal or voice is not the primary means by which they express emotions. It is therefore essential to recognize the limitations of the technologies used in order to improve them in the best way possible.

1.3 Computational Biases and Fairness

With the rise in popularity of AI systems that affect our daily lives, it is critically important to make sure they are not only accurate, but also designed to be as fair as possible. Nowadays, in fact, more and more decisions are controlled by AI algorithms, and the benefits of using automated decision-making systems are clear: algorithms are capable of handling and integrating much more data than a human may grasp, and also of performing complex computations much faster. One would also expect an automated algorithm to be more objective and fair than a human being, but unfortunately it is not always like this. These automated decision-making systems are used in various fields that significantly impact people's lives: systems that decide which individual will receive a job, a loan, bail or parole. For this reason, it is critically important to determine and improve the ethics of decisions made by these systems.
In the context of decision-making, when we talk about **fairness** we refer to the absence of any prejudice or favoritism toward an individual or group based on their inherent or acquired characteristics [12]. Thus, an unfair algorithm is one whose decisions are skewed toward a particular group of people. These biased predictions are usually a direct consequence of hidden biases in data or algorithms.

1.3.1 Causes of Unfairness and Types of Bias

We can identify different causes that lead to unfairness in machine learning [13, 14], and consequently we can talk about different types of biases [12].

- **Bias Encoded in Data**: human biases are frequently present in training data. These biases can arise from various sources, such as historical societal inequalities, human prejudices, or flawed data collection methods. Machine learning algorithms seek to fit data, and this inevitably will perpetuate existing biases. In this case we talk about *Historical biases* and *Measurement biases.*

- **Algorithmic Bias**: it refers to the inherent biases present in the design, development, or deployment of machine learning algorithms. These biases can lead to unfair or discriminatory outcomes, even if the training data itself is unbiased. Algorithmic bias can arise from various sources, including the choice of features, the optimization objectives, or the decision-making processes embedded within the algorithm.

- **Biases caused by "proxy" attributes for sensitive attributes**: sensitive attributes, such race, gender, and age, are often used to distinguish between privileged and unprivileged groups and should not be used in decision-making. Proxy attributes are non-sensitive attributes that can be exploited to derive sensitive attributes. If the dataset contains proxy attributes, the machine learning algorithm may make conclusions based on sensitive attributes while using supposedly legitimate attributes.

1.3.2 Measures of Algorithmic Fairness

Generally speaking, fairness is the absence of any prejudice or favoritism towards an individual or a group based on their intrinsic or acquired traits.
We can distinguish two types of fairness: group fairness and individual fairness. **Group fairness** criteria focuses on ensuring fairness for entire groups or categories of individuals, typically defined by sensitive attributes such as race, gender, or age. **Individual fairness**, on the other hand, focuses on ensuring fairness for individual individuals, irrespective of their group membership or demographic characteristics, following the principle that "similar individuals should receive similar treatments". In this work I will focus on assessing group fairness.
Having an intuitive definition of fairness is not enough because is important to be able to quantify the level of fairness/unfairness. In section 2 I will present the formal definitions of algorithmic fairness. The research emphasizes the trade-off between accuracy and fairness, with higher levels of fairness potentially compromising accuracy [15, 16, 17]. A fairness-aware algorithm aims to prioritize fairness without sacrificing accuracy or efficiency.

Chapter 2

Related Works

2.1 Speech Emotion Recognition

Speech Emotion Recognition is a speech processing and computational paralinguistics task that seeks to identify and categorize emotions communicated through spoken language. The purpose is to discern a speaker's emotional state based on their speech patterns, which include prosody, pitch, and rhythm. This is a topic that has been getting a lot of attention lately in literature, and there are several possible methods to approach it. The approaches differ in both the pre-processing and feature extraction methods and the classification models used, which range from machine learning methods to deep learning and transfer learning.

In [18] and [19], the authors identify and discuss several areas of SER, providing a full overview of existing literature for each, as well as highlighting the current challenges. The extraction of speech features is an essential step in SER. Acoustic aspects of speech are classified into two types: prosodic features and spectral features. Among the prosodic features, the most widely used in the literature are pitch [20, 21], zero-crossing rate (ZCR) [22] or voice quality features such as jitter and shimmer. For what concerns spectral features, the most used by researchers are MFCC [20, 22, 23] or linear prediction cepstral coefficient (LPCC) [23]. An interesting innovation was proposed by authors in [24]: they introduce the Mel Frequency Magnitude Coefficient (MFMC), very similar to MFCC but in this case magnitude spectrum is used instead of energy spectrum and the discrete cosine transform is not computed. The authors observe that MFMC recognizes emotions with better accuracy than the traditional spectral features.

[25] provides a comparison among the performance of popular machine-learning algorithms with several different feature sets, concluding that Support Vector Machine (SVM) is one of the best performing models. Other machine-learning classifiers commonly used in literature are Gaussian Mixture Model (GMM) [26]

and K-Nearest Neighbour (KNN) [27].
Speaking instead of deep neural network classifiers, [28] and [29] present an overview of the most used deep learning techniques, providing a comparison among their results on different datasets.
We currently have only two datasets available in Italian for the speech emotion recognition task (EMOVO [30] and Emozionalmente [31]), and most of the literature has used EMOVO to produce results on the Italian language. I will now report the most relevant works in this sense. In [32], authors propose a lightweight Convolutional Neural Network (LCNN) which extracts useful features automatically, and evaluate the model on several publicly available datasets, reaching an accuracy of 81% on EMOVO. In [33], the authors propose a SER system that uses log-Mels as an input to our Convolutional Recurrent Global Neural Network (CRGNN). A Convolutional Neural Network (CNN) is used to extract local invariant features using log-Mels, followed by a Recurrent Neural Network (RNN) to learn the temporal correlations between multiple time-step local invariant features. Finally, the most active features are selected using the Global Max Pooling technique. With this method they obtain an accuracy of 65% on EMOVO. In [34], an algorithm is proposed that uses deep learning to extract high-level features from raw data with great accuracy, regardless of language or speakers (male/female) of voice corpora. The reported emotion identification accuracy outperforms previous studies across languages and speakers.

2.2 Bias Detection

Preventing bias and discrimination is a long-standing issue in philosophy and psychology, and recently also in machine learning. However, in order to be able to fight against discrimination and achieve fairness, it's important to define what it means. Prior to computer science, philosophers and psychologists attempted to define fairness, but the lack of a uniform definition highlights the challenge of addressing this issue. Diverse cultures have diverse perspectives on fairness, making it challenging to define a universally accepted standard.
In [35], authors study the 50-year history of fairness definitions in the areas of education and machine-learning. They compare past and current notions of fairness along several dimensions, including the fairness criteria, the focus of the criteria, the relationship of fairness to individuals, groups and subgroups and the mathematical method for measuring fairness. They analyze the cultural and social contexts that have influenced these definitions and conclude that, in some cases, earlier definitions of fairness may be similar or identical to those in present machine learning research, while in other cases insights into what fairness means and how to measure it have largely gone overlooked.

In [36], authors collect the most prominent definitions of fairness in algorithmic classification problems and explains the rationale behind these definitions, demonstrating each of them on a single unifying case-study. Their research explains why the same case can be considered fair according to some definitions and unfair according to others.
In the next section, I will present the most prominent measures of algorithmic fairness in machine learning classification tasks [14].

2.2.1 Fairness Metrics

- **Disparate Impact Ratio**: this measure was designed to mathematically represent the legal notion of *disparate impact*. It requires a high ratio between the positive prediction rates of both groups. This ensures that the proportion of the positive predictions is similar across groups. Formally is defined as follows:

  $$\frac{P[\hat{Y} = 1 \mid S \neq 1]}{P[\hat{Y} = 1 \mid S = 1]} \geq 1 - \epsilon \tag{2.1}$$

  S represent the protected attribute, hence S=1 is the privileged group and S$\neq$1 is the unprivileged one. $\hat{Y}$=1 means that the prediction is positive. A higher value of this measure represents more similar rates across groups and therefore more fairness.

- **Statistical Parity Difference**: also known as Demographic Parity Difference, is very similar to disparate impact but we take the difference instead of the ratio. Formally is defined as follows:

  $$\mid P[\hat{Y} = 1 \mid S \neq 1] - P[\hat{Y} = 1 \mid S = 1] \mid \leq \epsilon \tag{2.2}$$

  A lower value of this measure represents better fairness. This metric ensure that the positive prediction is assigned to the two groups at a similar rate.

- **Average Odds Difference**: also known as Equalized Odds Difference, computes the difference between the false positive rates (FPRs) and the difference between the true positive rates (TPRs) of the two groups. Formally is defined as follows:

  $$\mid P[\hat{Y} = 1 \mid S = 1, Y = 0] - P[\hat{Y} = 1 \mid S \neq 1, Y = 0] \mid \leq \epsilon \tag{2.3}$$

  $$\mid P[\hat{Y} = 1 \mid S = 1, Y = 1] - P[\hat{Y} = 1 \mid S \neq 1, Y = 1] \mid \leq \epsilon \tag{2.4}$$

  Smaller differences between groups indicate better fairness. It is important to notice that since average odds relies on the actual ground truth (i.e., Y) it assumes that the base rates of the two groups are representative and were not obtained in a biased manner.

- **Equal Opportunity Difference**: this metric is similar to average odds but focuses on the true positive rates only. It requires true positive rates (TPRs) to be similar across groups. Formally is defined as follows:

  $$| P[\hat{Y} = 1 \mid S \neq 1, Y = 1] - P[\hat{Y} = 1 \mid S = 1, Y = 1] | \leq \epsilon \tag{2.5}$$

  Also in this case, a smaller difference indicate better fairness. It is important to note that following the equality in terms of only one type of error (e.g., true positives) will increase the disparity in terms of the other error.

2.2.2 Assessment Tools

Researchers have also recently introduced tools for assessing the amount of fairness in a system, in order to allow people working in the industry to develop fair machine learning applications in an easier way.

For example Aequitas [37] is an open source bias and fairness audit toolkit that allows users to test models for several bias and fairness metrics in relation to multiple population sub-groups. It generates reports for data scientists, machine learning researchers and policymakers in order to make informed decisions and prevent harm toward specific groups.

Another toolkit is IBM's AI Fairness 360 (AIF360) [38], that aims to integrate fairness research algorithms into industrial settings, establish a benchmark for evaluating algorithms, and foster collaboration among fairness researchers. The package provides fairness measurements, explanations, and techniques to reduce bias in datasets and models. The platform offers an interactive web experience for line-of-business users, comprehensive documentation, usage guidance, and industry-specific tutorials to help data scientists and practitioners choose the best tool for their needs.

Chapter 3

Methods

In this chapter, I will dive into the technical details of my work.
To begin, I will discuss the step-by-step approach of how I measured fairness, explaining the implementation of each metric in detail. Next, I will break down how we created visual plots to make our findings easy to understand. In the second part of the chapter I will explain the technical details related to the implementation of the baselines used to obtain the first results, namely SVM and ResNet. Lastly, I will discuss the features of WavLM, the pre-trained model actually used to carry out the project and obtain the best results.

3.1 Fairness

3.1.1 Metrics

The focus of this work is to study and quantify the biases that result from using speech emotion recognition systems. In order to do this, it was necessary to implement the metrics most commonly used in the literature, but adapting them to the specific context of this study.
The task we are dealing with is a speech emotion recognition task, that is, a classification task in which we can assign as a possible output label one of the six basic emotions of Ekman's theory: sadness, disgust, fear, anger, joy, surprise, plus neutrality for the intervals in which no emotion is expressed, for a total of seven possible labels. Usually, however, when we talk about bias and use metrics to assess fairness, we are always dealing with binary outputs. In fact, we always refer to *positive outcome* and *negative outcome* as two output alternatives for labels. For this reason, one of the most significant contributions of this study was to reformulate the metrics in such a way that they gained meaning in the multiclass setting in which we are working.

Below, whenever we refer to a sensitive attribute we are talking about any characteristic or personal trait of an individual that is considered private, protected, or potentially discriminatory. These attributes often include characteristics such as race, gender, sexual orientation, religion, disability status, age, socioeconomic status, ethnicity, and others. In our case, the sensitive attributes considered are gender and age. In order for these sensitive attributes to gain meaning in talking about bias, two groups are always identified: a *privileged group* and a *unprivileged group*. This means that the sensitive attribute must always take binary values: in the case of gender this is immediate, while in the case of age there is a need to identify a threshold to divide the population into young and old.
I will now illustrate each metric used in detail:

Statistical Parity Difference

This metric measures the difference in favorable outcomes between different demographic groups, to ensure that the model's predictions are not biased against any particular group based on their protected attributes such as race, gender, or age. This is the general definition, but it cannot be directly applied to our situation without being redefined, because we do not have a clear description of what a *favorable outcome* is. For this reason, we redefine statistical parity difference in such a way that a value is calculated for each possible emotion, i.e. label. In this way, we establish that we are talking about *favourable outcome* when the emotion we are considering at that moment is assigned.
In this setting, to compute the statistical parity difference for each label we follow these steps:

1. Creation of a dictionary to store the count of positive predictions for each label and sensitive attribute value;

2. Computation of the proportion of positive predictions for each label and sensitive attribute value, by dividing the count by the total number of instances for the sensitive attribute value we are considering;

3. Calculation of the actual statistical parity difference by subtracting the values corresponding to the same emotion for the two values of the sensitive attribute.

A Statistical Parity Difference value of 0 indicates that there is no difference in the rate of favorable outcomes between the privileged and unprivileged groups, indicating perfect fairness. We extend the concept of fairness to a range of -0.1 to 0.1. A negative value indicates that the privileged group is more likely to receive favorable outcomes than the unprivileged group, suggesting bias or discrimination. A positive value indicates that the unprivileged group is more likely to receive favorable outcomes than the privileged group, which might also indicate bias or discrimination, even though in the opposite direction.

Equal Opportunity Difference

This metric requires true positive rates (TPRs), meaning the probability of an individual with a positive outcome to have a positive prediction, to be similar across groups. It is computed as the difference of true positive rates between the unprivileged and the privileged groups. The true positive rate is the ratio of true positives to the total number of actual positives for a given group. This is the general definition that we need to rephrase to make it applicable to our setting. In our case we calculate an equal opportunity difference value for each label and we talk about *positive prediction* when the emotion we are considering at that moment is assigned.

In this setting, to compute equal opportunity difference for each label we follow these steps:

1. Creation of a dictionary to store predictions and true values separately with respect to the sensitive attribute;
2. Computation of the recall score (True Positive Rate) for each label separated by sensitive attribute value;
3. Calculation of the actual equal opportunity difference by subtracting the values corresponding to the same emotion for the two values of the sensitive attribute.

An Equal Opportunity Difference value of 0 indicates perfect fairness. We extend the concept of fairness to a range of -0.1 to 0.1. A negative value indicates that the privileged group is more likely to receive favorable outcomes than the unprivileged group, suggesting bias or discrimination. A positive value indicates that the unprivileged group is more likely to receive favorable outcomes than the privileged group, which might also indicate bias or discrimination, even though in the opposite direction.

Average Odds Difference

This metric, also known as **equalized odds**, is similar to equal opportunity but in addition to computing the difference between the true positive rates (TPR) of the two groups, also computes the difference between the false positive rates (FPR). The average between TPRs differences and FPRs differences is then calculated. Also in this case we obtain an average odds difference value for each label, considering a different emotion each time as a *positive prediction.* To compute average odds difference for each label we follow the following steps:

1. Creation of a dictionary to store predictions and true values separately with respect to the sensitive attribute;
2. Computation of the recall score (True Positive Rate) for each label separated by sensitive attribute value;
3. Calculation of TPRs difference for each label;
4. Computation of the False Positive Rate for each label separated by sensitive attribute value, by computing the confusion matrices for each label and sensitive attribute to extract false positive and true negative values;
5. Calculation of FPRs difference for each label;
6. Calculation of the actual average odds difference by computing the average between TPRs differences and FPRs differences.

An Average Odds Difference value of 0 indicates perfect fairness. We extend the concept of fairness to a range of -0.1 to 0.1. A negative value indicates that the privileged group is more likely to receive favorable outcomes than the unprivileged group, suggesting bias or discrimination. A positive value indicates that the unprivileged group is more likely to receive favorable outcomes than the privileged group, which might also indicate bias or discrimination, even though in the opposite direction.

Disparate Impact Ratio

This measure was designed to mathematically represent the legal notion of *disparate impact*. It is computed as the ratio between the positive prediction rates of both sensitive groups. This ensures that the proportion of the positive predictions is similar across groups. As above, we redefine this metric by computing a disparate impact ratio value for each label, and we talk about *positive prediction* when the emotion we are considering at that moment is assigned.
To compute the disparate impact ratio for each label we follow these steps:

1. Creation of a dictionary to store the count of positive predictions for each label and sensitive attribute value;
2. Computation of the proportion of positive predictions for each label and sensitive attribute value, by dividing the count by the total number of instances for the sensitive attribute value we are considering;
3. Calculation of the ratio obtained comparing the two groups.

A Disparate Impact Ratio value of 1.0 indicates perfect fairness. We extend the concept of fairness to a range of 0.8 to 1.25. A value less than 1 indicates that the privileged group is more likely to receive favorable outcomes than the unprivileged group, suggesting bias or discrimination. A value greater than 1 indicates that the unprivileged group is more likely to receive favorable outcomes than the privileged group, which might also indicate bias or discrimination, even though in the opposite direction.

The implementation of these metrics is available in appendix A.

3.1.2 Plots

To make the results obtained from the metrics easily usable, there is a need to display the values in an intuitive way. In fact, we get many numbers as output that are meaningless unless put in a context where their meaning is explained. For this reason, with the project team, we came up with a visualization of the results that would be as clear and immediate as possible.

Considering the data we are working with, I decided to visualize the results using a horizontal histogram. I placed emotion-related labels on the y-axis (*neutrality, surprise, disgust, joy, sadness, fear and anger*) and fairness-related information on the x-axis. Specifically, there are three labels on the x-axis: *perfect fairness, bias toward privileged group, and bias toward unprivileged group.* Perfect fairness corresponds to the value zero for the metrics Statistical Parity difference, Equal Opportunity difference and Average Odds difference, while it corresponds to the value 1 for the Disparate Impact metric. In both cases the label is positioned at the center of the x-axis. We then have on the left the label *bias towards privileged group* and on the right the label *bias towards unprivileged group.* Metrics values for each emotion are displayed at the end of the histogram column for clarity. As specified in the previous section, we do not refer only to perfect fairness, but extend the concept of fairness to a range to make it more applicable in practice. To identify it in the graph, I have highlighted the corresponding range. In this way if the histogram bar ends up in the highlighted range we know that we are respecting fairness otherwise in case it ends up outside we know that we are in the presence of bias and, thanks to the labels on the x-axis, we also have information about the direction of the bias.

The colors used to represent emotions with the histogram were chosen with reference to the model proposed by Ekman in the Atlas of Emotions [39]. Blue (RGB 60 175 175) is used to indicate surprise, green (RGB 100 153 65) is used for disgust, yellow (RGB 255 255 0) is used for joy, blue (RGB 64 106 173) is used for sadness, purple (RGB 91 57 136) is used for fear, red (RGB 160 61 62) is used for anger, and gray (RGB 229 229 229) is used for neutrality, as shown in figure 3.1.

The implementation of the plots is available in appendix B.

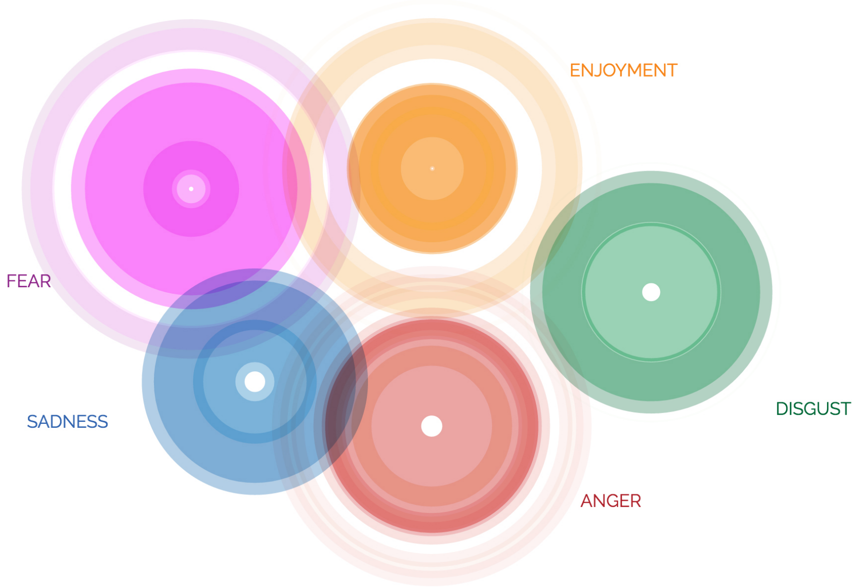

Figure 3.1: Emotion's colors according to the model proposed by Ekman in the Atlas of Emotions

3.2 Feature Extraction

When working with audio-type input data, the feature extraction step is necessary for several reasons:

- **Dimensionality reduction:** audio signals are tipically high-dimensional. They contain a huge volume of raw data, which can be computationally expensive to handle and may result in overfitting when training models. Feature extraction helps reduce this dimensionality by selecting a subset of informative features that capture relevant information about the speech signal;

- **Discriminative features:** not all parts of the audio signal are equally important for recognizing emotions. Feature extraction allows us to identify and extract features that are most discriminative for distinguishing between different emotional states. These features could include characteristics like pitch, intensity, spectral features, etc., which are known to be correlated with various emotions;

- **Noise robustness:** audio signals are often contaminated with noise from various sources such as background chatter, environmental sounds, or microphone interference. Feature extraction can help in extracting features that are robust to noise, thus improving the performance of emotion recognition systems in real-world environments;

- **Model efficiency:** extracting meaningful features from the audio signal allows for more efficient modeling. Models trained on well-selected features tend to generalize better to unseen data and require less computational resources during both training and inference phases;
- **Interpretability:** extracted features often have intuitive interpretations, making it easier to understand how the model is making predictions. This can be particularly important in applications where interpretability is necessary.

Overall, feature extraction plays a vital role in Speech Emotion Recognition by transforming raw audio signals into a more manageable and informative representation.
For my work, I extracted three types of features based on the model that I then used to solve the Speech Emotion Recognition task: MFCC, MFMC and Mel spectrograms. The details are explained below.

3.2.1 Mel-Frequency Cepstral Coefficient

The Mel-Frequency Cepstral Coefficient (MFCC) is the feature most commonly employed in automatic speech emotion identification systems. It improves speech accuracy by utilizing human auditory perception. MFCCs are frequently utilized as features in speech recognition and speaker identification systems because they effectively reflect the key aspects of speech signals while being robust to differences in pronunciation, accent, and noise. They encompass both the frequency content and the temporal dynamics of the signal in a compact representation, making them useful for a wide range of speech processing applications.
To calculate the MFCC, the following steps are required:

1. **Pre-emphasis:** the first step in the MFCC calculation is to apply a pre-emphasis filter to the signal. This filter is applied to emphasize high-frequency components over low-frequency ones. Voiced spectrum of speech has higher energy at low frequency than at high frequency so, to balance the spectrum of voiced sound, it is necessary to improve the energy at high frequencies. In this work, a pre-emphasis filter with coefficient 0.97 was used, as given in Equation 3.1;
$$H(z) = 1 - 0.97z^{-1} \tag{3.1}$$
2. **Framing and Windowing:** splitting signals into discrete frames allows for more stationary signals. Speech must be analyzed over a short period of time to ensure steady acoustic features. Therefore, the pre-emphasis filtered signal

is divided into short-time frames. It is critical to choose a frame duration carefully because if it is too long, the signal properties vary throughout the frame. On every frame an Hamming window is applied to smooth the signal and prevent the frames from discontinuities, as defined in Equation 3.2 (N in the equation is the length of the frame). In this work, a frame duration of 20 ms is used, with 50% window overlapping;

$$w[n] = 0.54 - 0.46 \cdot \cos\left(\frac{2\pi n}{N-1}\right) \tag{3.2}$$

3. **Fast Fourier Transform Magnitude Squared:** the windowed frames are then transformed into the frequency domain using the Fast Fourier Transform (FFT) algorithm. This converts the signal from the time domain to the frequency domain, providing a representation of the signal's frequency content. The Fourier transform is computed and then the magnitude squared is extracted. In this work, I computed the one-dimensional 512-point discrete Fourier transform and I kept only the first 257 coefficents;

4. **Mel Filterbank:** next step is applying the Mel Filterbank. The Mel scale is a nonlinear scale of pitch perception that better approximates the human auditory system's response to frequency. The Mel Filterbank consists of a set of triangular filters spaced evenly on the Mel scale. Each filter is centered at a specific frequency and covers a certain range of frequencies. The outputs of these filters represent the energy distribution across different frequency bands;

5. **Logarithmic Compression:** after passing through the Mel filterbank, the outputs are transformed using a logarithmic function. This step compresses the dynamic range of the filterbank outputs and makes the representation more aligned with human perception of sound intensity;

6. **Discrete Cosine Transform:** the final step involves applying the Discrete Cosine Transform (DCT) to the log filterbank energies. The DCT coefficients represent the spectrum of the signal in a compact form. Typically, only a subset of the resulting coefficients is retained, as they contain most of the relevant information about the spectral envelope of the signal. In this work, I performed the experiments considering three subsets of resulting coefficients: 12, 24 and 30.

3.2.2 Mel-Frequency Magnitude Coefficient

Even though MFCC is the most commonly employed feature, research community is yet to attain an optimal emotion recognition rate. This can be caused by different reasons:

- Short time energy is utilized to extract MFCC; however, energy is not a sufficient feature for large signal levels because it uses square functions;
- The transformation vector for the discrete cosine transform (DCT) contains all frequencies. If a frequency band is contaminated by any noise factor, it impacts all the MFCCs;
- If a voiced phoneme is adjacent to an unvoiced phoneme in a frame, the dominant voiced phoneme will remain in the frame while the unvoiced phoneme will disappear. This causes information loss in the speech.

These motivations led the authors in [24] to propose a new spectral feature, the Mel-Frequency Magnitude Coefficient (MFMC), which recognizes the emotions with better accuracy than the traditional spectral features. The MFMCs are obtained by modifying the MFCCs extraction process, to overcome the above mentioned problems.
In particular, MFMC is extracted the same way as that of MFCC with the exception of two steps: first, magnitude of fast Fourier transform is used instead of magnitude square. Second, the discrete cosine transform used in the MFCC extraction for the purpose of decorrelation is excluded. Figure 3.2 shows the two extraction processes.

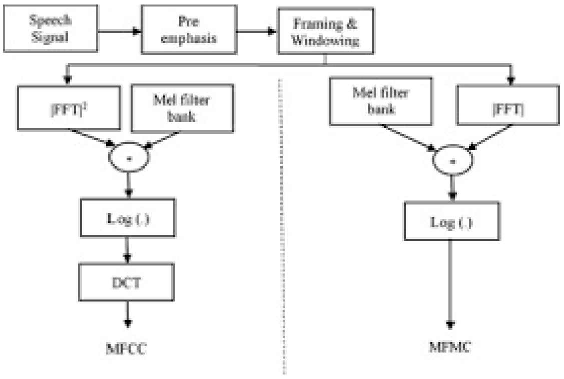

Figure 3.2: Extraction process of MFCC and MFMC

3.2.3 Mel Spectrograms

A Mel spectrogram, short for Mel-frequency spectrogram, is a visual representation of the frequency content of a signal over time, where the frequency axis is scaled according to the mel scale, a perceptual scale of pitches that approximates the human ear's response to different frequencies. This type of feature is particularly useful when we want to use a model for the speech emotion recognition task that needs an image-type input, such as ResNet.
To calculate the Mel spectrogram, the following steps are required:

1. **Preprocessing:** the input audio signal is typically divided into short overlapping windows using a technique like the Short-Time Fourier Transform (STFT). Each window represents a short segment of the signal. A windowing function (such as the Hamming window) is applied to each segment to reduce spectral leakage;

2. **Frequency Domain Representation:** for each windowed segment, the Fourier Transform is computed to convert the signal from the time domain to the frequency domain. This results in a representation of the signal in terms of its frequency components and their respective magnitudes;

3. **Mel Filterbank:** the Mel filterbank is a set of triangular filters that are spaced according to the mel scale. The mel scale is a nonlinear scale that approximates the human auditory system's perception of pitch. It is based on psychoacoustic experiments that measured the perceived distances between tones. The Mel filterbank is applied to the power spectrum obtained from the Fourier Transform. Each filter in the bank selectively extracts energy from specific frequency bands;

4. **Log Compression:** the energy within each mel filter's frequency band is summed, and then the logarithm of the sum is taken. This process compresses the energy values and emphasizes differences in low-energy regions while compressing high-energy regions. This logarithmic compression helps to mimic the logarithmic perception of loudness in the human auditory system;

5. **Spectrogram Visualization:** finally, the results of the log compression process are arranged over time to form the Mel spectrogram. Each column of the spectrogram represents a short time segment, and the rows correspond to different frequency bands defined by the Mel filterbank. The intensity of each pixel in the spectrogram represents the log energy of the signal within the corresponding time-frequency bin.

An example of Mel Spectrogram is shown in Figure 3.3.

3.3 Baselines

The implemented fairness metrics were tested on two baselines to verify that they worked properly. The baselines serve two purposes: to provide a more controllable and controlled setting in which to test the metrics, as well as to determine whether bias and accuracy are related. In fact, the literature agrees that the presence of bias is often a direct consequence of poor learning ability of the models.

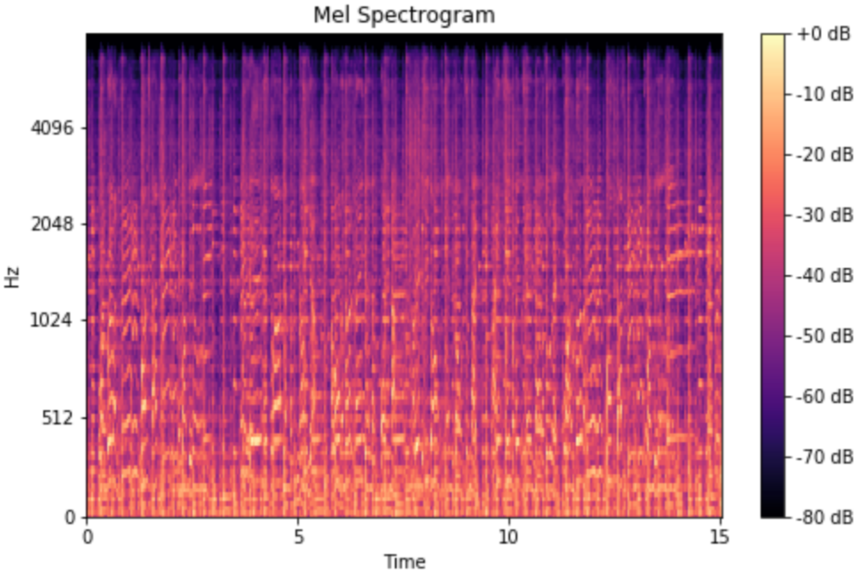

Figure 3.3: Example of Mel Spectrogram

3.3.1 Support Vector Machine

The first baseline I implemented is a classification model using Support Vector Machine (SVM). As input, I provided two types of data: I conducted experiments both using MFCCs and using MFCMs.
Support Vector Machine is a supervised machine learning algorithm used for classification tasks. It works by finding the hyperplane in an N-dimensional space (N being the number of features) that best separates different classes in the feature space. But there are numerous hyperplanes that might be used in order to divide the two classes of data points apart. The objective is to find a plane that has the maximum margin, i.e the maximum distance between data points of both classes, so that future data points can be classified with more confidence.
SVM does not inherently allow multiclass classification in its most basic form, it supports binary classification and separating data points into two classes. But the algorithm can be extended to the multiclass case by using the same principle after breaking down the multiclassification problem into multiple binary classification problems. In this work, we need to apply multiclass SVM.
Briefly, SVM works like this:

1. **Data Representation:** SVM begins with a set of labeled training data, where each data point belongs to one class. Each data point is represented as a feature vector in a high-dimensional space;

2. **Handling Non-linear Data:** In cases where the data is not linearly separable,

SVM can still find an optimal separating hyperplane by using kernel functions. These functions implicitly map the input vectors into a higher-dimensional space, where the data may be linearly separable. Common kernel functions include linear, polynomial, radial basis function (RBF), and sigmoid;

3. **Optimization:** SVM solves a convex optimization problem to find the optimal hyperplane. The most common formulation is the hinge loss function, which penalizes misclassifications. Techniques like gradient descent or quadratic programming are used to solve this optimization problem efficiently;

4. **Classification:** once the optimal hyperplane is found, SVM can classify new data points by determining which side of the hyperplane they fall on.

For my experiments, I used the *SVC* implemented function of *sklearn* with a linear kernel. More details about the experiments setting in Chapter 4.

3.3.2 ResNet

The second baseline I implemented, as opposed to the first, is a deep learning model. In particular, I decided to implement a ResNet.
ResNet, short for "Residual Network", is a deep neural network architecture introduced in [40]. The key innovation of ResNet is the use of residual blocks, which allow for the training of very deep neural networks (up to hundreds of layers) without suffering from the vanishing gradient problem. In a traditional neural network, each layer directly feeds into the next layer. However, in a residual block, the input to a layer is combined with the output of a previous layer, effectively creating a "shortcut connection" or a "skip connection." This allows gradients to flow more directly through the network during training, facilitating the training of very deep networks.
Because ResNet requires image-type data as input, I used Mel Spectrograms as features with this baseline. There are several variants of ResNet which differ in terms of depth (i.e., the number of layers), and in this work I implemented ResNet-18.
ResNet-18 has a total of 18 layers, including 16 convolutional layers and 2 fully connected layers. It has fewer parameters compared to deeper ResNet variants, making it more computationally efficient and easier to train on less powerful hardware. I will now provide a simplified overview of the implemented architecture:

1. **Input layer:** it accepts RGB images of size 224x224 pixels (Mel Spectrograms);

2. **Convolutional layer:** the network starts with a convolutional layer, with a kernel size of 7x7, a stride of 2, and a padding of 3, followed by batch

normalization and ReLU activation. This is followed by a 3x3 max-pooling layer with a stride of 2;

3. **Residual Blocks:** ResNet-18 consists of four sets of residual blocks. Each set contains two basic blocks, making a total of 8 basic blocks in the entire network. Each basic block consists of two convolutional layers with 3x3 filters, each followed by batch normalization and ReLU activation. The output of the second convolutional layer is added to the input of the block through a shortcut connection;

4. **Global Average Pooling:** after the last residual block, global average pooling is applied over the feature maps, resulting in a vector of features for each image;

5. **Fully Connected Layer:** finally, a fully connected layer followed by a softmax activation is used to produce the final class scores.

Figure 3.4 shows a diagram of the ResNet-18 architecture.

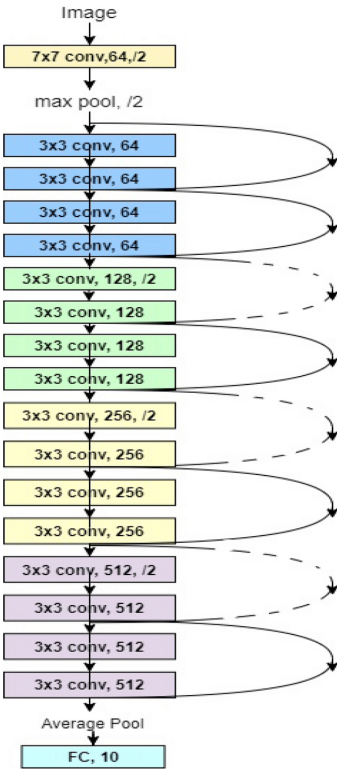

Figure 3.4: Diagram of ResNet-18 architecture

3.4 WavLM

As mentioned in the introduction, this thesis is part of the Knock Knock project developed by the LINKS Foundation. Therefore, in addition to testing the functions implemented for detecting bias on baselines, I also tested them on WavLM, which is the model used for the KK project. By comparing the results obtained from baselines and WavLM, I was also able to draw conclusions about the relationship between model bias and accuracy.
WavLM is a pre-trained model proposed in [41] to solve full-stack downstream speech tasks. It is an adaptable system that efficiently learns universal speech representations from large amounts of unlabeled speech data and can be applied to a variety of speech processing tasks. In pre-training, WavLM simultaneously learns denoising and masked speech prediction. In this way, it maintains the ability to model speech content through masked speech prediction while simultaneously enhancing its potential for non-ASR (Automatic Speech Recognition) tasks through speech denoising.

The model architecture uses the Transformer model as the backbone. It has a convolutional feature encoder and a Transformer encoder, as seen in Figure 3.5. Seven blocks of temporal convolution, followed by layer normalization and a GELU activation layer, make up the convolutional encoder. The temporal convolutions have 512 channels with strides (5,2,2,2,2,2,2) and kernel widths (10,3,3,3,3,2,2), resulting in each output representing about 25ms of audio strode by 20ms. The convolutional output representation x is masked as the Transformer input. The Transformer is equipped with a convolution-based relative position embedding layer with 128 kernel size and 16 groups at the bottom. In order to enhance the model, a relative position bias is utilized, which is encoded according to the Transformer self-attention mechanism's offset between the "key" and "query".

To fit the model to the speech emotion recognition task, these steps were followed:

1. an average pooling was performed on the final representations obtained from the context network;
2. the resulting vector was passed through a ReLU;
3. logits were obtained through a linear layer of dimension emb_dim x n_classes, where emb_dim is the final dimension of the vectors obtained from WavLM.

A schematic of the modified model is shown in Figure 3.6.

During the training performed to fit the model to our task, the weights of the context network and those of the added linear layer were finetuned. Instead, the feature extractor was freezed. Information regarding the datasets used to perform the training is in the Section 4.3.3.

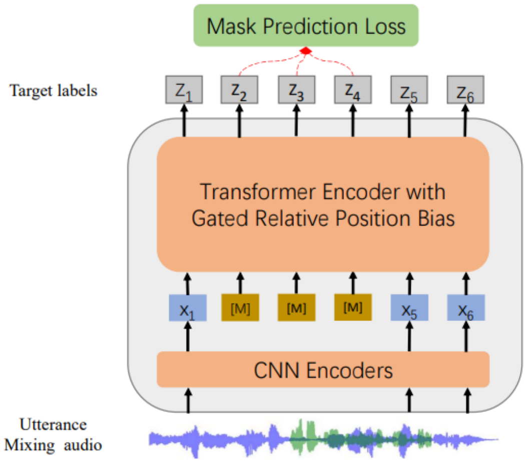

Figure 3.5: WavLM architecture

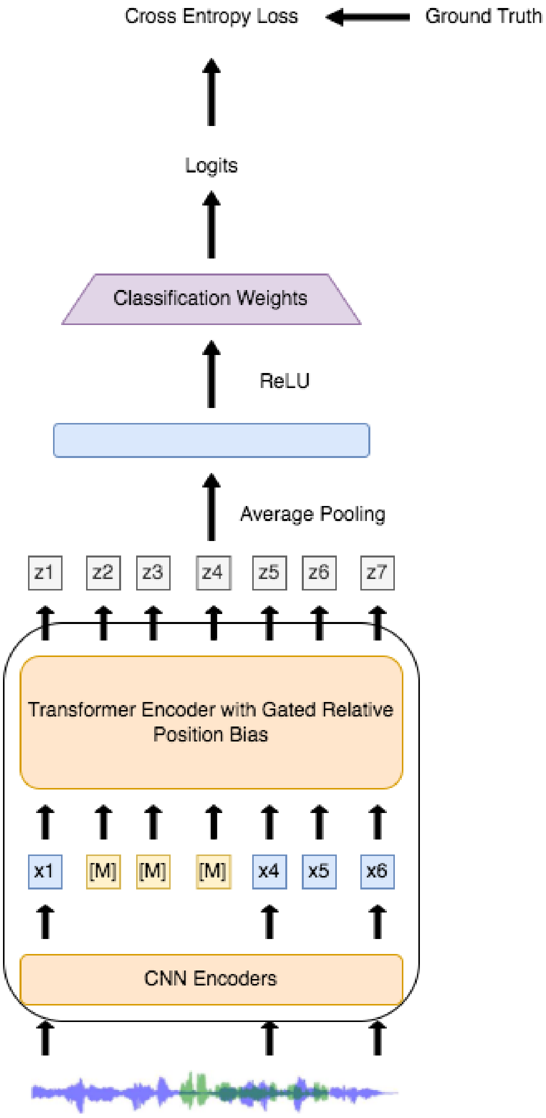

Figure 3.6: WavLM architecture modified to perform speech emotion rercognition task

Chapter 4

Experiments and Results

4.1 Datasets

To carry out the experiments, I used the only two datasets for the speech emotion recognition task available in the Italian language: **EMOVO** and **Emozionalmente**.
Below are the features of the two datasets in detail, along with a summary table (Table 4.1) highlighting similarities and differences on the most relevant features.

4.1.1 EMOVO

EMOVO [30] is the first database of emotional speech for the Italian language. It is a simulated dataset, built from the voices of six actors, three males and three females with proven expertise, with ages between 23 and 30 years old. The actors were asked to play 14 sentences simulating 6 emotional states (disgust, fear, anger, joy, surprise, sadness) plus the neutral state, without delivering explicit emotional indicators such as laughter or tears, which would distort the recognition test. These emotions are the well-known Big Six found in most of the literature related to emotional speech.
The 14 phrases played are the following:

1. Gli operai si alzano presto.

2. I vigili sono muniti di pistola.

3. La cascata fa molto rumore.

4. L'autunno prossimo Tony partirà per la Spagna nella prima metà di ottobre.

5. Ora prendo la felpa di là ed esco per fare una passeggiata.

6. Un attimo dopo s'è incamminato ... ed è inciampato.
7. Vorrei il numero telefonico del Signor Piatti.
8. La casa forte vuole col pane.
9. La forza trova il passo e l'aglio rosso.
10. Il gatto sta scorrendo nella pera.
11. Insalata pastasciutta coscia d'agnello limoncello.
12. Uno quarantatré dieci mille cinquantasette venti.
13. Sabato sera cosa farà?
14. Porti con te quella cosa?

Sentences 1-7, 13, 14 are semantically neutral, meaning that the semantic value of the content is emotionally neutral, while sentences 8-10 are nonsense. The authors decided to use both categories of sentences because if the former can pose a challenge to the actor to place them in the right emotional state, the latter involve the risk of being recited in a stereotypical manner, as the actor may not be able to "hear" as natural. In addition, for the purpose of spectral analysis, the following basic conditions were satisfied: presence in the sentences of all the phonemes of the Italian language and presence in every sentence of a fair balance between voiced and unvoiced consonant.
The recordings were made with professional equipment in the Fondazione Ugo Bordoni laboratories. The recordings were performed with a sampling frequency of 48 kHz, 16 bit stereo, wav format.
The database is composed of a total of 588 records. For each actor we have 98 sentences, corresponding to 14 sentences spoken in 6 emotional states plus the neutral one. This results in approximately 10 minutes per actor.

4.1.2 Emozionalmente

Emozionalmente is a crowd-sourced emotional speech corpus presented by authors in [31] in order to address the gap of Italian-language datasets in speech emotion recognition literature.
The dataset contains 6902 samples, recorded by 431 non-professional actors, all italian: 131 males, 299 females and 1 that listed themselves as "other", with an average age of 31 years old and a standard deviation of 12. Users were asked to play one or more sentences taken from a list with an emotion of their choice from the Big Six (disgust, fear, anger, joy, surprise, sadness plus the neutral state). The list containing the 18 phrases from which actors could choose is the following:

1. Gli operai si alzano presto.
2. La cascata fa molto rumore.
3. Vorrei il numero telefonico del Signor Piatti.
4. Non sapevo che fosse in città.
5. L'ho incontrato oggi dopo due anni.
6. Zia Marta ha detto che devo stare a casa sta sera.
7. Ho preso 6 nella verifica di matematica.
8. Tommaso ha detto che dovevo scegliere io cosa fare.
9. Il capo mi ha affidato un altro lavoro.
10. Tornerà a casa presto.
11. Vado in biblioteca.
12. È una notte stellata.
13. Oggi c'è una partita di basket.
14. È impegnato in una riunione.
15. È andato a scuola dopo pranzo.
16. Il cane ha riportato qui la palla.
17. Giovanni parte per Roma domani.
18. Ieri un gatto ha bevuto dalla tazza.

This sentences are constructed ad-hoc to be semantically neutral and easily readable with different emotional tones. They include everyday vocabulary and all phonemes of the Italian language. On top of that, three sentences from EMOVO were also included in the Emozionalmente sentence set.
Each actor performed on average 16 sentences, emotions are expressed uniformly (986 times each) and every sentence was verbalized 383 times on average. Recordings were generally obtained with non-professional equipment, for a total of 26297 seconds. They have 2 channels (stereo), a sample size of 16 bits, and a .wav format. 6839 audio recordings were obtained with a sampling rate of 48 kHz and 63 of them with 44.1 kHz, depending on the characteristics of the recording device.

In contrast to EMOVO, in Emozionalmente:

- The actors were not necessarily professionals, but a general audience without any restrictions on age or gender;
- Actors could perform as many sentence-emotion combinations as they wanted, but were not required to play all of them;
- Speech was not necessarily recorded with professional equipment, but using the instrumentation available to the actors;
- Recordings were not done in a noiseless laboratory, but anywhere the actors wished.

In Table 4.1 there is a comparison between the two datasets.

Table 4.1: Comparison between EMOVO and Emozionalmente

	Datasets	
	EMOVO	**Emozionalmente**
number of files	588	6902
number of actors	6 (3M, 3F)	431 (131M, 299F)
actors background	professional	non-professional
tipology of dataset	simulated	simulated
source	laboratory	crowd-sourcing
equipment	professional	non-professional
emotions	big six + neutral	big six + neutral
number of sentences	14	18
balanced	yes	no
sample size	16 bit stereo	16 bit stereo
format	wav	wav
sampling frequency	48 kHz	48 kHz and 44.1 kHz

4.2 Sensitive Attributes

The purpose of the experiments is to assess and quantify the presence of bias. To do this, the metrics outlined in Chapter 3 (statistical parity difference, equal opportunity difference, average odds difference and disparate impact ratio) were considered. In order to calculate the bias we need to consider it with respect to a sensitive attribute. When we refer to a sensitive attribute we are talking about any characteristic or personal trait of an individual that is considered private, protected, or potentially discriminatory. These attributes often include characteristics such as race, gender, age, religion, and others. In our case, the sensitive attributes considered are gender and age.

In particular, when we deal with the EMOVO dataset, we only consider gender as a sensitive attribute because we don't have in the dataset information about the age of the participants. There are only two genders present as options in the dataset (males and females), so I simply converted them to binary mode (females: 0 and males: 1).

In the experiments conducted using the Emozionalmente dataset, on the other hand, in addition to gender I was able to consider age as a sensitive attribute. Regarding gender, three options were available: female, male, and other. Since I need a binary sensitive attribute, I decided to eliminate the instances that presented "other" as gender and converted the remaining ones to binary (females: 0 and males: 1). The instances that presented "other" as a gender were very few so there was not much loss of information. With regard to age, however, the situation is different. In fact, age is a discrete but non-binary attribute and therefore there is a need to decide on a criterion to make it so. The idea is to divide into two groups, young and old, but in doing so there is a need to decide on a threshold for making this division. The decision criteria for threshold are both to maintain logical meaning, and therefore divide in such a way that the groups of young and old make sense to common sense, and to have a fairly balanced distribution. As a first step, I visualized the age distribution, as shown in Figure 4.1. As can be seen, the range of ages varies from 1 to 79, with a prevalence of individuals in their 20s and 30s.

I therefore decided to consider three possible thresholds:

- Threshold 1: **27**. This threshold was chosen because it is the one that divides the dataset in a more balanced way. In fact, in this way we have 480 *young* (0) and 458 *old* (1);

- Threshold 1: **30**. This threshold was chosen because, compared to 27, it is a more meaningful division according to common sense to divide into young and old. It should be noted, however, that by dividing in this way the groups are much more unbalanced: 641 *young* and 297 *old*;

- Threshold 1: **40**. This last threshold is, in my view, the most meaningful according to common sense but also the most unbalanced. In fact we have 792 *young* and 146 *old*.

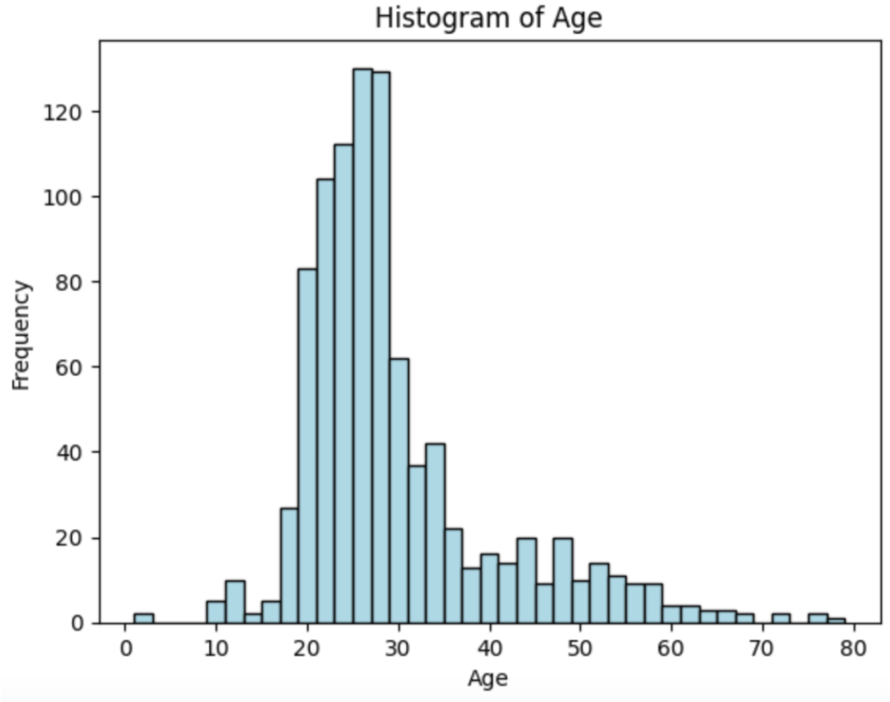

Figure 4.1: Age distribution

In the experiments I considered all three thresholds so as to see if and how the bias changes based on the division chosen.

4.3 Experiments

In this section I will illustrate the settings for all the conducted experiments, dividing them according to the model and dataset used.

4.3.1 SVM

The setting of experiments carried out using SVM as a model changes according to the database used.

EMOVO

EMOVO has a small amount of examples, so I decided to perform the experiments by applying k-fold cross-validation. In particular, I considered three possible values of k (5,10 and 15) to see if different configurations of the training dataset would go to change the bias. For each value of k, I calculated the results for each split for

accuracy and each of the fairness metrics. To then have final values, I averaged over all the results obtained from the different splits.
With SVM I considered both MFCC and MFMC as features, and for both I considered 12, 24 and 30 spectral coefficients. Considering then that I used 3 k-values for k-fold cross-validation for each experiment, for EMOVO with SVM I have a total of 18 experiments, summarized in Table 4.2

Feature	**Coefficients**	**K-fold Splits**		
MFCC	12	5	10	15
	24	5	10	15
	30	5	10	15
MFMC	12	5	10	15
	24	5	10	15
	30	5	10	15

Table 4.2: Summary of experiments configuration (SVM using EMOVO)

Emozionalmente

Emozionalmente is a dataset that contains enough examples to be able to do the classic train and test splits. Therefore, both when I considered gender and age as sensitive attributes, I divided the dataset into train and test splits and computed accuracy and values related to fairness metrics.
When I consider gender as a sensitive attribute, I do two different splits: in the first case, I take 70% of the dataset for training and 30% for testing, stratifying by emotion. Since Emozionalmente, however, is a very unbalanced dataset in terms of gender, visualizing the frequency immediately shows the imbalance, as seen in Figure 4.2.

For this reason, I decided to do another split, such that the training set was balanced, to see if a balanced training set contributed to better results in terms of bias. In this case we have a balanced training set but a very unbalanced test set, as can be seen in Figure 4.3.

So for the gender, considering the two features MFCCs and MFMCs with 12, 24 and 30 spectral coefficients each and doing experiments with both the unbalanced and balanced training set, I have a total of 12 experiments, summarized in Table 4.3.

When I consider age as a sensitive attribute, I do experiments considering all three thresholds. Therefore, considering the two features MFCCs and MFMCs with 12, 24 and 30 spectral coefficients each, I have a total of 18 experiments, summarized in Table 4.4.

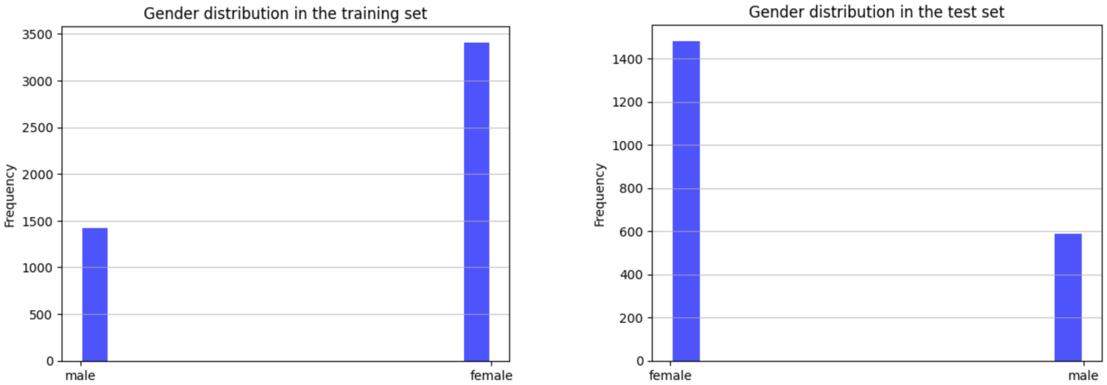

Figure 4.2: Gender frequency in unbalanced training and test split (Emozionalmente)

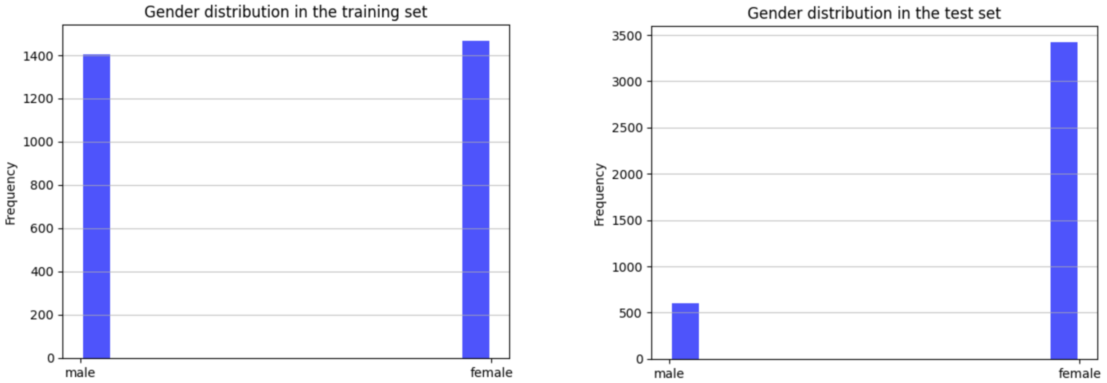

Figure 4.3: Gender frequency in balanced training and test split (Emozionalmente)

Feature	Coefficients	Dataset	
MFCC	12	unbalanced	balanced
	24	unbalanced	balanced
	30	unbalanced	balanced
MFMC	12	unbalanced	balanced
	24	unbalanced	balanced
	30	unbalanced	balanced

Table 4.3: Summary of experiments configuration (SVM using Emozionalmente and gender)

Feature	Coefficients	Age threshold		
MFCC	12	27	30	40
	24	27	30	40
	30	27	30	40
MFMC	12	27	30	40
	24	27	30	40
	30	27	30	40

Table 4.4: Summary of experiments configuration (SVM using Emozionalmente and age)

4.3.2 ResNet

Even in the case of experiments performed using ResNet as a model, the configurations change depending on the dataset used.

EMOVO

I performed three types of experiments using ResNet as the model and EMOVO as the dataset.
In the first scenario, I made the dataset split into training, validation, and test split without taking gender into account. To do the splits, I took 70% of the dataset to form the training set and of the remaining 30% I used 20% to create the validation set. However, since EMOVO is a balanced dataset in this respect (the actors are 4 males and 4 females and they all recited the same number of sentences) the splits are all balanced, as can be seen in Figure 4.4.

I then wanted to test whether the gender distribution in the training set had an impact on the final bias presented by the model. Therefore, I created two training sets with different distributions: in one case I created a training set with 70% males and 30% females and in another case a training set with 70% females and 30%

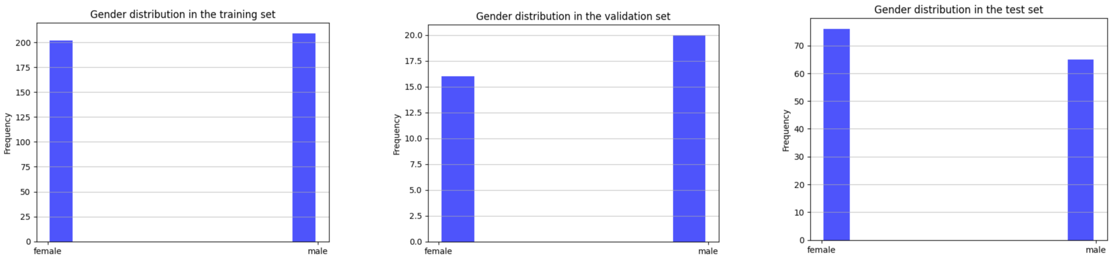

Figure 4.4: Gender frequency in training, validation and test balanced splits (EMOVO)

males. The validation and test sets were created accordingly with the remaining examples. The frequency of the two genders in the various sets is shown in Figures 4.5 and 4.6.

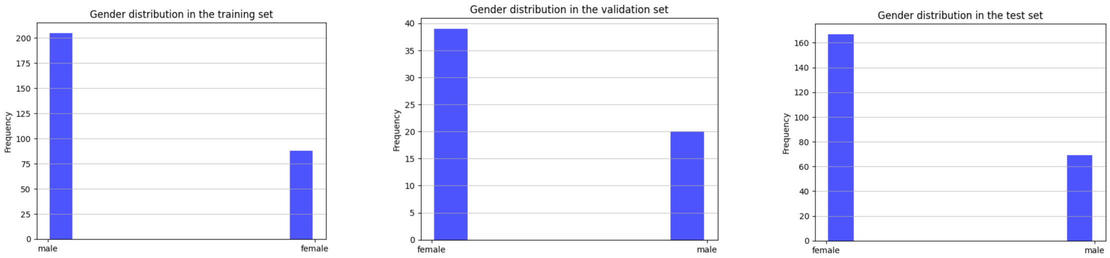

Figure 4.5: Gender frequency in training, validation and test when considering training set with 70% males (EMOVO)

In all experiments, the model settings remained the same: I trained the model for 20 epochs, as a loss function I used a Cross Entropy Loss, and as an optimizer Adam with a learning rate of 0.0001 and weight decay of 1e-4. I also used a learning rate scheduler to adaptively adjust the learning rate during training (ReduceLROnPlateau).
Having performed one experiment with each of the three training set configurations, I performed a total of 3 experiments using ResNet as the model and EMOVO as the dataset.

Emozionalmente

In the experiments carried out using Emozionalmente as a dataset and ResNet as a model, two cases must be distinguished according to the sensitive attribute considered.
When the sensitive attribute considered is gender, I performed two experiments: one with an unbalanced training set and one with a balanced training set. The

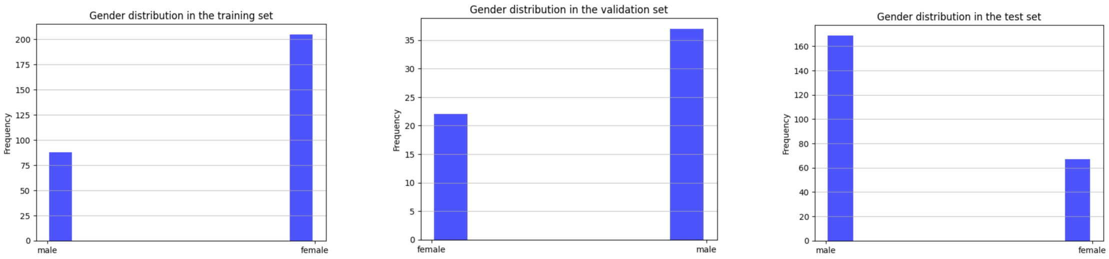

Figure 4.6: Gender frequency in training, validation and test when considering training set with 70% females (EMOVO)

unbalanced training set was obtained by dividing the original dataset without taking gender into account. However, being an unbalanced dataset in this sense, the divisions into train, validation, and test also reflect this. Gender frequencies in the various splits are shown in Figure 4.7. To obtain these splits, I considered 70% of the original dataset to create the training set and of the remaining 30% I used 20% to create the validation set.

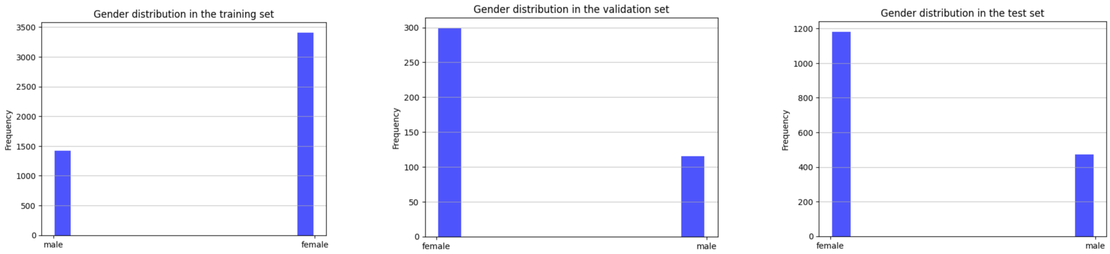

Figure 4.7: Gender frequency in training, validation and test when considering unbalanced training set (Emozionalmente)

To create a balanced training set, I had to take 70% of the males and 30% of the females from the original dataset. The validation and test splits were then created with the remaining data. The gender frequencies in the various splits are shown in Figure 4.8. Since the dataset is very unbalanced, in creating a balanced training set we could not consider too many examples and therefore the test split is more numerous.

In all experiments, the model settings remained the same: I trained the model for 20 epochs, as a loss function I used a Cross Entropy Loss, and as an optimizer Adam with a learning rate of 0.0001 and weight decay of 1e-4. I also used a learning rate scheduler to adaptively adjust the learning rate during training (ReduceLROnPlateau).

Having performed one experiment with each of the two training set configurations, I performed a total of 2 experiments using ResNet as the model, Emozionalmente

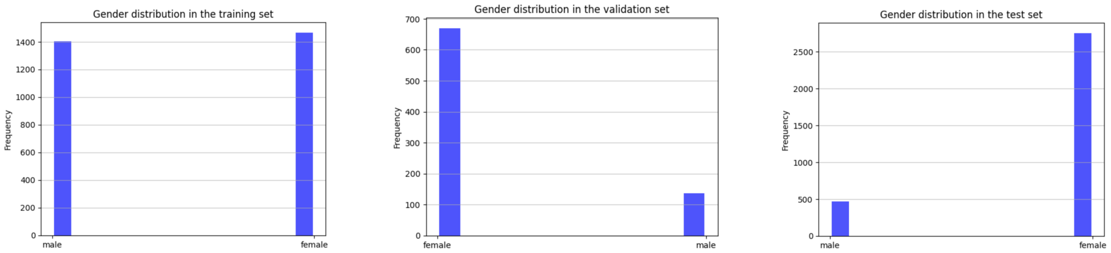

Figure 4.8: Gender frequency in training, validation and test when considering balanced training set (Emozionalmente)

as the dataset and gender as the sensitive attribute.

When the sensitive attribute considered is age, I performed an experiment considering each of the three thresholds, for a total of 3 experiments. The model settings in each of the experiments are the same as described above.

4.3.3 WavLM

Experiments using WavLM as a model were carried out using three different configurations of training and test splits.
In the first setup, an Emozionalmente split was used to fine-tune the model and the entire EMOVO dataset was used to test it. Emozionalmente was divided into a train split and a test split without regard to gender distribution. As we have already mentioned, since Emozionalmente is an unbalanced dataset in this sense, the training set obtained in this way is unbalanced. In this first configuration, having used EMOVO as the test dataset, it was possible only to consider gender as a sensitive attribute.
In the second configuration, the training split obtained from Emozionalmente was used for fine tuning, and the test set also obtained from Emozionalmente was used for testing. Because this time the test is done on Emozionalmente, I was able to consider both gender and age as sensitive attributes. Specifically, I conducted experiments considering each of the three thresholds considered for age.
In the last setup, I used the balanced training split shown in Figure 4.8 as the training set, and as a test set the split obtained as a result by putting together validation and test split of the image. This way I can test whether a balanced training set positively affects bias, although in order to get a balanced split I had to consider fewer examples than the original training set.
In total, I performed 6 experiments using WavLM as the model.

4.4 Results

The objectives of the experiments were as follows: to assess the extent of bias in the models considered, to evaluate any correlation between bias and accuracy, and to assess whether a different training set distribution affects bias.

4.4.1 Model Bias

For each of the experiments described in Section 4.3, I calculated the accuracy and bias results using the four fairness metrics described in Chapter 3: Statistical Parity Difference, Equal Opportunity Difference, Average Odds Difference and Disparate Impact Ratio. I produced a plot for each metric and each experiment. The plots produced can be found in the Appendix C. In Figure 4.9 you can see an example related to the experiment performed using the EMOVO dataset, the SVM model with MFCC feature with 24 spectral coefficients and 15 kfold splits.

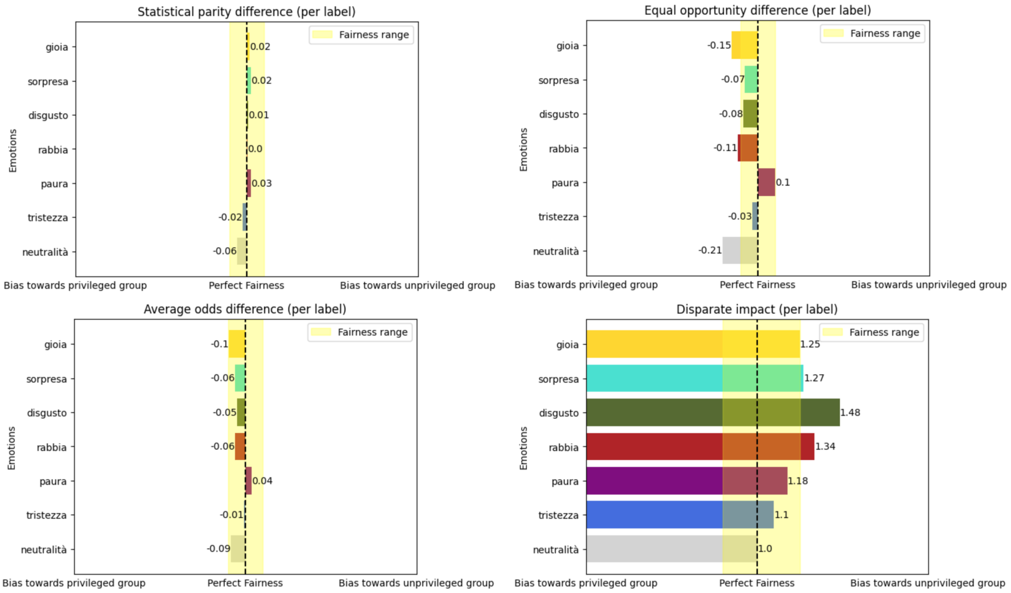

Figure 4.9: Fairness metric plot example

The plots are constructed in this way: on the x-axis are listed the emotions considered while on the y-axis we have an indication of the direction of the bias. The center of the x-axis (indicated by the dashed line) and the yellow-highlighted area around it indicate the values for which we consider the model fair. When the bar in the histogram breaks to the left of the yellow-highlighted area we are in the presence of bias toward what we call the *privileged group*, while when the bar breaks to the right of the yellow-highlighted area we are in the presence of bias toward what we call the *unprivileged group*.

In general, the models exhibit bias with respect to all fairness metrics considered. However, this is most likely related to the very low performance in accuracy, as we will see in the next section.

4.4.2 Accuracy and Bias

The relationship between accuracy and bias has been mentioned several times in the literature, and through my experiments I have also seen this: low accuracy indicates a poor ability of the model to generalize, and this can lead to a focus on data features that are related to sensitive attributes. Through the experiments, it was possible to see this trend either by changing the settings of the experiments while keeping the same model and dataset, or by testing the same dataset on different models with different performance.

Below I will provide an overview of the most significant results, dividing them according to the dataset used. To show the results I will use tables and not plots because it is more immediate that way to notice trends.

EMOVO

I will first discuss the results related to the experiments carried out using SVM as a model.

Results for experiments conducted using MFCCs as features are shown in Tables 4.5, 4.6 and 4.7. The numbers shown refer to the values obtained using the different fairness metrics considered. Cells highlighted in yellow indicate values that do not fall within the fairness range and are therefore considered indicators of bias. We can immediately see a trend of decreasing bias as accuracy increases. In fact, the model has higher accuracy as the spectral coefficients considered increase: the first table (Table 4.5, 12 spectral coefficients) refers to an experiment with 41% accuracy, the second table (Table 4.6, 24 spectral coefficients) refers to an experiment with 60% accuracy, and the third table (Table 4.7, 30 spectral coefficients) refers to an experiment with 64% accuracy. Looking at the amount of cells colored yellow, we can see that this decreases as accuracy increases.

On the other hand, when we consider the results obtained by considering MFMCs as features, we have much lower accuracies, and this is also reflected in fairness, as can be seen in Tables 4.8, 4.9 and 4.10. Table 4.8 refers to an experiment with accuracy of 33%, Table 4.9 to an experiment with accuracy of 41% and Table 4.10 to an experiment with accuracy of 43%. The performance obtained with these features is therefore much lower than that obtained with MFCCs, and from the amount of cells colored yellow, it is immediately apparent that the fairness is also

	Gioia	Sorpresa	Disgusto	Rabbia	Paura	Tristezza	Neutralità
Statistical Parity	0.09	0.1	-0.06	0.06	-0.11	-0.03	-0.06
Equal Opportunity	-0.03	0.1	0.03	0.03	0.01	-0.07	-0.03
Average Odds	-0.07	0.01	0.04	-0.01	0.06	-0.02	0.01
Disparate Impact	2.22	3.08	0.74	2.33	0.49	0.79	0.75

Table 4.5: MFCC, 5 splits, 12 coefficients

	Gioia	Sorpresa	Disgusto	Rabbia	Paura	Tristezza	Neutralità
Statistical Parity	0.01	0.06	-0.03	0.01	0	0	-0.04
Equal Opportunity	-0.11	0.1	-0.04	-0.08	0.07	-0.11	-0.08
Average Odds	-0.07	0.02	-0.01	-0.05	0.04	-0.06	-0.03
Disparate Impact	1.06	2.21	0.98	1.35	1.05	1.02	0.81

Table 4.6: MFCC, 5 splits, 24 coefficients

	Gioia	Sorpresa	Disgusto	Rabbia	Paura	Tristezza	Neutralità
Statistical Parity	0.03	0.02	0.01	0.01	-0.01	0	-0.05
Equal Opportunity	-0.19	-0.1	-0.04	-0.1	0.04	-0.05	-0.1
Average Odds	-0.12	-0.07	-0.03	-0.05	0.02	-0.03	-0.03
Disparate Impact	1.21	1.48	1.24	1.09	0.98	1.15	0.75

Table 4.7: MFCC, 5 splits, 30 coefficients

worse. This may confirm the theory that bias is related to accuracy.

	Gioia	Sorpresa	Disgusto	Rabbia	Paura	Tristezza	Neutralità
Statistical Parity	0.14	0.06	-0.03	-0.02	-0.06	-0.11	0.02
Equal Opportunity	0.32	0.13	0.09	-0.16	0.07	0.01	0.16
Average Odds	0.22	0.09	0.08	0.09	0.2	0.16	0.13
Disparate Impact	7.81	0.33	0.59	0.95	0.76	0.6	1.11

Table 4.8: MFMC, 5 splits, 12 coefficients

	Gioia	Sorpresa	Disgusto	Rabbia	Paura	Tristezza	Neutralità
Statistical Parity	0.12	0.08	-0.01	-0.05	-0.17	0	0.04
Equal Opportunity	0.35	0.1	0.07	-0.14	-0.17	0.23	0.34
Average Odds	0.22	0.14	0.13	0.16	0.18	0.16	0.22
Disparate Impact	4.42	3.04	0.94	0.91	0.28	0.98	1.24

Table 4.9: MFMC, 5 splits, 24 coefficients

Even considering the experiments conducted using ResNet and WavLM, the observed trend continues to be valid. In fact, in the experiment performed using ResNet I obtain an accuracy of 44%, lower than the best obtained with SVM (64%). With WavLM I obtain an accuracy of 64%. Tables 4.11 and Table 4.12 show the values for the fairness metrics in the respective experiments. The amounts of bias in WavLM are greater than those in SVM for the same accuracy, but the biases in the ResNet model are more.

Emozionalmente (Gender)

Even in the experiments carried out using Emozionalmente as the dataset and considering gender as a sensitive attribute, we find the same trend. In this case, the experiments performed using the SVM model have very low accuracy both using MFCCs and using MFMCs. In Table 4.13 and Table 4.14 we have the results for the fairness metrics in the two best experiments (30 spectral coefficients). We

	Gioia	Sorpresa	Disgusto	Rabbia	Paura	Tristezza	Neutralità
Statistical Parity	0.12	0.05	-0.03	-0.03	-0.15	0	0.03
Equal Opportunity	0.34	0.07	0.11	-0.13	-0.04	0.13	0.35
Average Odds	0.27	0.12	0.1	0.14	0.16	0.15	0.27
Disparate Impact	3.66	2.14	0.85	1.08	0.34	0.99	1.4

Table 4.10: MFMC, 5 splits, 30 coefficients

	Gioia	Sorpresa	Disgusto	Rabbia	Paura	Tristezza	Neutralità
Statistical Parity	0.02	-0.03	-0.2	0.12	0.13	-0.09	0.05
Equal Opportunity	0.1	0.13	-0.3	-0.05	0.46	-0.1	0.04
Average Odds	0.04	0.09	-0.06	-0.04	0.2	0	-0.03
Disparate Impact	1.14	0.71	0.25	2.08	2.2	0.5	1.47

Table 4.11: ResNet, balanced training set

	Gioia	Sorpresa	Disgusto	Rabbia	Paura	Tristezza	Neutralità
Statistical Parity	0.11	-0.11	-0.08	-0.01	0.05	0.02	0.02
Equal Opportunity	0.31	-0.26	-0.29	-0.02	0.1	0.12	0.12
Average Odds	0.12	-0.09	-0.12	-0.01	0.02	0.06	0.06
Disparate Impact	3.38	0.45	0.48	0.95	1.44	1.64	1.15

Table 4.12: WavLM, EMOVO test set

have accuracy of 32% and 31%, respectively, and indeed the amount of bias found is also broadly the same.

	Joy	**Surprise**	**Disgust**	**Anger**	**Fear**	**Sadness**	**Neutrality**
Statistical Parity	0.09	0.06	-0.02	-0.08	0.02	0	-0.11
Equal Opportunity	0.21	0.2	0.05	-0.12	0.22	0.06	-0.11
Average Odds	0.07	0.08	0.04	-0.03	0.1	0.02	0
Disparate Impact	2.35	1.59	0.83	0.64	1.59	1.12	0.59

Table 4.13: MFCC, 30 coefficients, unbalanced training set

	Joy	**Surprise**	**Disgust**	**Anger**	**Fear**	**Sadness**	**Neutrality**
Statistical Parity	0.09	0.07	-0.13	-0.01	0.07	0.02	-0.1
Equal Opportunity	0.28	0.19	-0.06	-0.07	0.31	0.03	0
Average Odds	0.11	0.07	0.04	-0.04	0.14	0.01	0.06
Disparate Impact	2.22	2.04	0.38	0.89	1.73	1.13	0.65

Table 4.14: MFMC, 30 coefficients, unbalanced training set

With this dataset, even using ResNet we have a very low accuracy of 33% and indeed the amount of bias found is high, as can be seen in Table 4.15. However, it immediately jumps out when looking at the Table 4.16 that by using a much better performing model (WavLM which achieves an accuracy of 90% with Emozionalmente), the biases are almost completely reduced.

4.4.3 Training Set Impact

Another thing this study aims to test is whether there is a correlation between amount and direction of bias and distribution of the sensitive attribute in the training set. This kind of study I did by focusing on the gender sensitive attribute. This is because gender is a "fixed" attribute, inherent in the subject, whereas with age the threshold is chosen arbitrarily.
Audios produced by males and females are not the same: in general, it can be seen

	Joy	**Surprise**	**Disgust**	**Anger**	**Fear**	**Sadness**	**Neutrality**
Statistical Parity	0.01	0.1	-0.07	-0.05	0.05	0	-0.05
Equal Opportunity	0.03	0.31	-0.07	-0.11	0.18	0.1	-0.03
Average Odds	0.02	0.12	0	-0.04	0.07	0.05	0.02
Disparate Impact	1.07	1.69	0.72	0.59	1.45	1.03	0.66

Table 4.15: ResNet, unbalanced training set

	Joy	**Surprise**	**Disgust**	**Anger**	**Fear**	**Sadness**	**Neutrality**
Statistical Parity	0.01	0	0.01	0.02	0	0	-0.04
Equal Opportunity	0.08	0.03	-0.08	0.13	-0.02	-0.02	-0.04
Average Odds	0.05	0.02	-0.04	0.07	-0.01	-0.02	-0.01
Disparate Impact	1.12	0.98	1.06	1.15	0.98	1.01	0.79

Table 4.16: WavLM, unbalanced training set

that female voices have a higher frequency and pitch. The models may therefore be more inherently able to recognize audio with certain physical characteristics. I therefore performed experiments by changing the distributions in the training set to see if and how this had an impact on the results obtained.

EMOVO

Using the EMOVO dataset, I could only do this test using ResNet as a model. In fact, in the experiments conducted using SVM I do not use a training set but use the cross validation technique with k-fold splits, and the dataset is too small to do finetuning on WavLM.
I tested two configurations of the training set for ResNet, in addition to the one with the balanced dataset: in one case I used a training set that had 70% males and in the other a training set that had 70%. It should be noted that as a result the testing set is also unbalanced, having used the remaining data to create it. This could make the accuracy and fairness metrics values not very reliable.
The first thing we notice is a drop in performance. When we use a balanced training set, the model gets an accuracy of 44.68%. In contrast, when we use the unbalanced datasets we get in the case of the dataset with 70% males an accuracy of 36% and in the case of the dataset with 70% females an accuracy of 38.55%. This may indicate a poor ability of the model to generalize the available data. In terms of bias, there are minimal changes in trend. For example, if we consider the Statistical Parity fairness metric, we can see in Figures 4.10, 4.11 and 4.12 that when we consider the results obtained with a balanced training set, the biases are directed equally to the left and right. In contrast, when we consider the male-dominated training set, the bias is reduced to zero. Using a female-dominated dataset we notice an increase in bias directed toward the "privileged" group, namely males.
We can therefore conclude that the model "recognizes" as different the audios produced by females and males, but the results on bias may be irrelevant because of very low accuracy.

Emozionalmente (Gender)

Emozionalmente, as seen from Figure 4.7, it is a female-dominated dataset. To test the impact of the training set on the results, I created a balanced training set, but as can be seen in Figure 4.8, this resulted in fewer training items and a very unbalanced test set. These elements could make the results obtained unreliable.
In the experiments carried out using SVM and ResNet as models, it can be seen in the Figures 4.13, 4.13, 4.15 and 4.16 how, when the balanced dataset is used to perform training, the overall amount of bias decreases. However, no clear trend in the direction of bias can be recognized. This could be related to the very low accuracy of the models, all around 30%. Indeed, the biases seem to be turned in

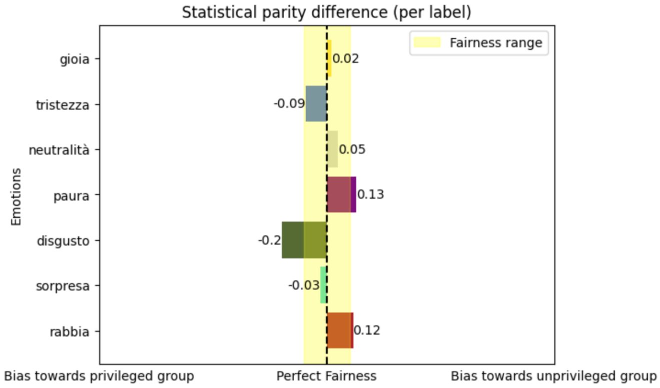

Figure 4.10: EMOVO, ResNet, balanced training set

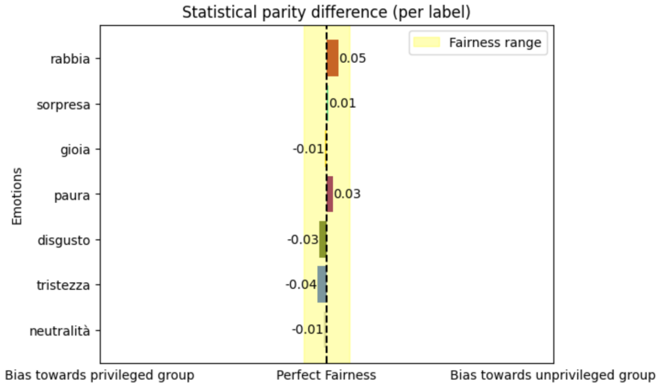

Figure 4.11: EMOVO, ResNet, 70% male training set

the direction of the unprivileged group, namely females. This could be related to an inferior ability of the models in general to work with audio with characteristics typical of the female voice. It should be noted, however, that when ResNet is used, there is an increase in accuracy of 6% from using the unbalanced dataset to using the balanced dataset for training. This could indicate that the use of balanced dataset helps the generalization capabilities of the model.

I performed two experiments using WavLM as a model: in one case I used the unbalanced female-dominated training set while in another case I used the balanced dataset. The first thing we notice is a drop in performance: in fact in the first case we have an accuracy of 90% while in the second one we have an accuracy of 65%. This could be caused by the smaller number of examples in the balanced training set. Regarding bias, we can see in Figures 4.17 and 4.18 that the values that fall outside the fairness range in the case where we use the balanced dataset are more, but this is probably related to the lower model accuracy. Regarding the direction of

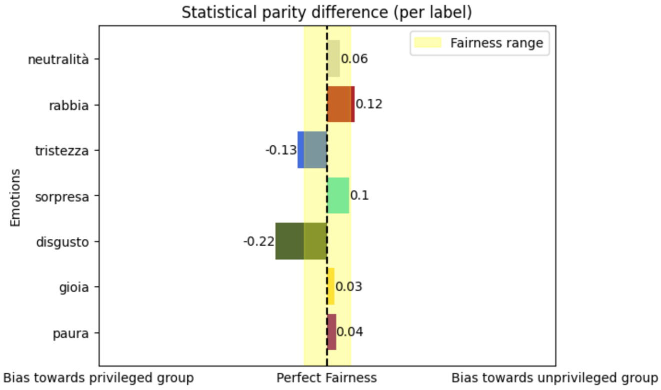

Figure 4.12: EMOVO, ResNet, 70% female training set

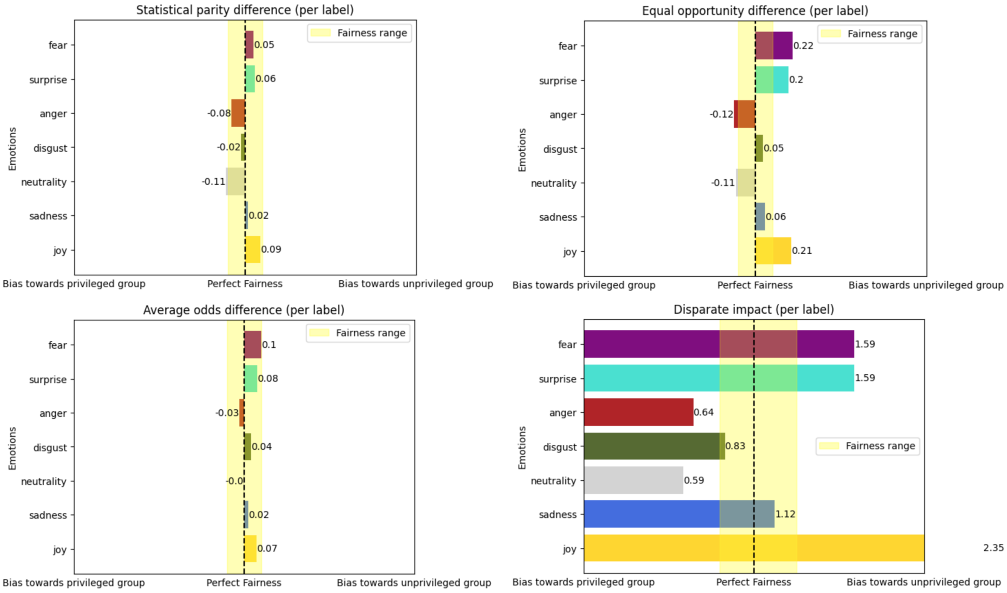

Figure 4.13: Emozionalmente, SVM, MFCC, 30 coefficients, unbalanced training set, gender (accuracy: 32.02%)

the biases, we again note that when we use the balanced dataset these increase in the direction of the unprivileged group, that is, females. The cause of this behavior could be the much higher number of examples related to female subjects in the testing set than those related to male subjects. Thus, we have no concrete results showing the impact of the training set on the amount and direction of bias.

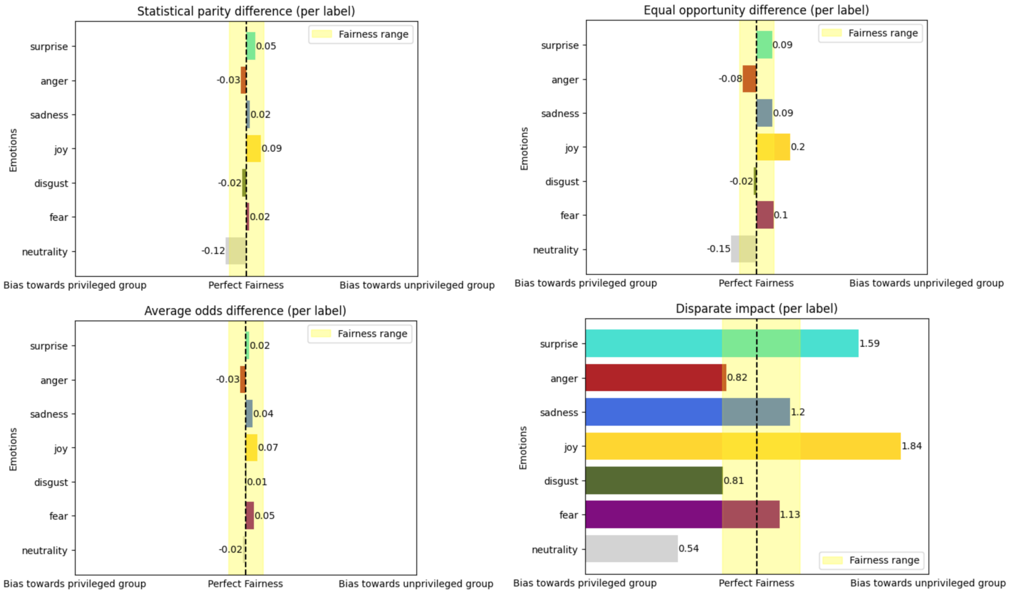

Figure 4.14: Emozionalmente, SVM, MFCC, 30 coefficients, balanced training set, gender (accuracy: 31.76%)

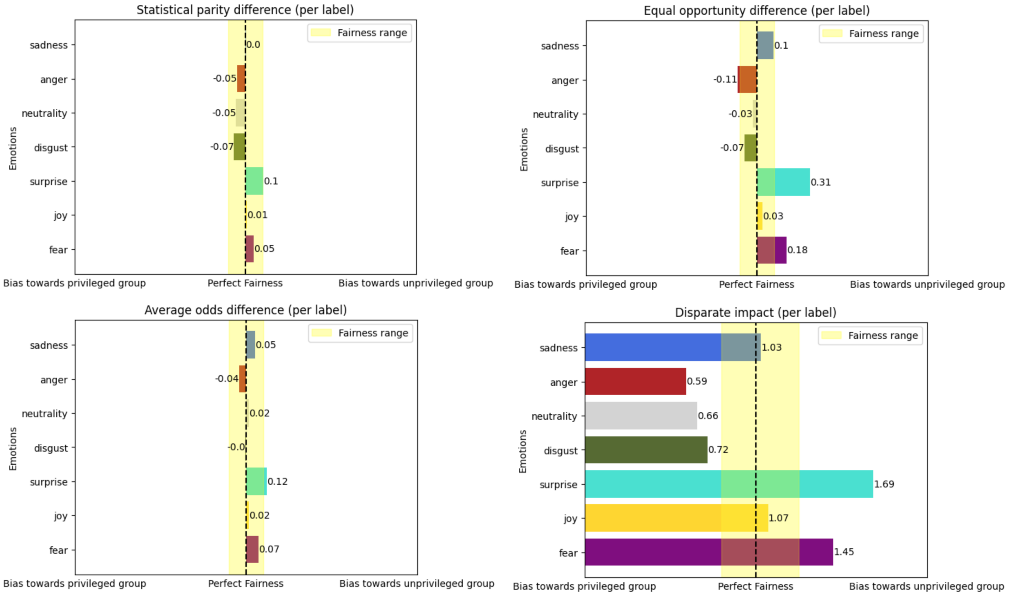

Figure 4.15: Emozionalmente, ResNet, unbalanced training set, gender (accuracy: 33.51%)

Emozionalmente (Age)

Regarding the sensitive age attribute, as mentioned earlier I considered three thresholds to make the division into "young" and "old": 27, 30 and 40. Using

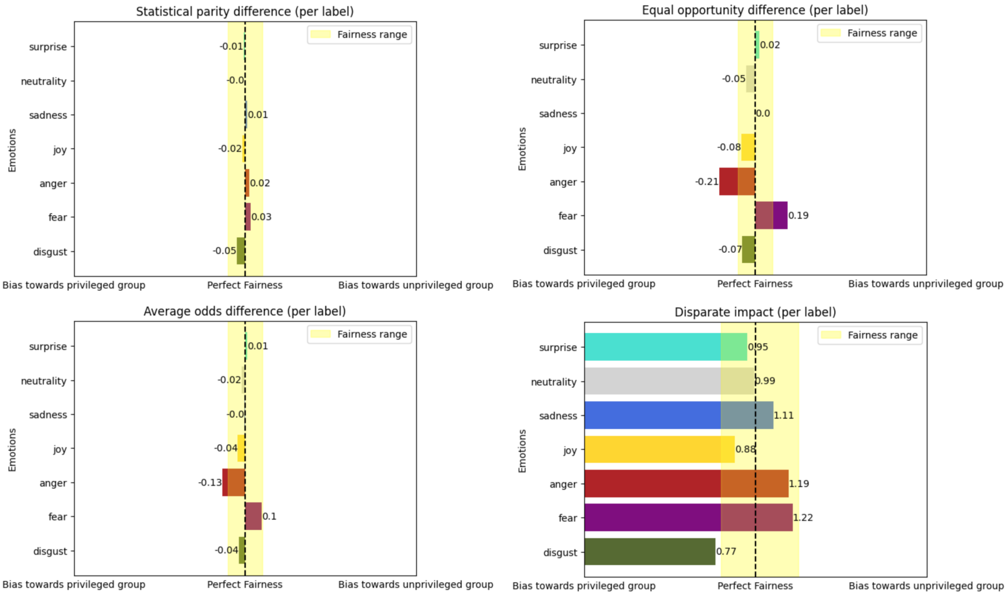

Figure 4.16: Emozionalmente, ResNet, balanced training set, gender (accuracy: 39.95%)

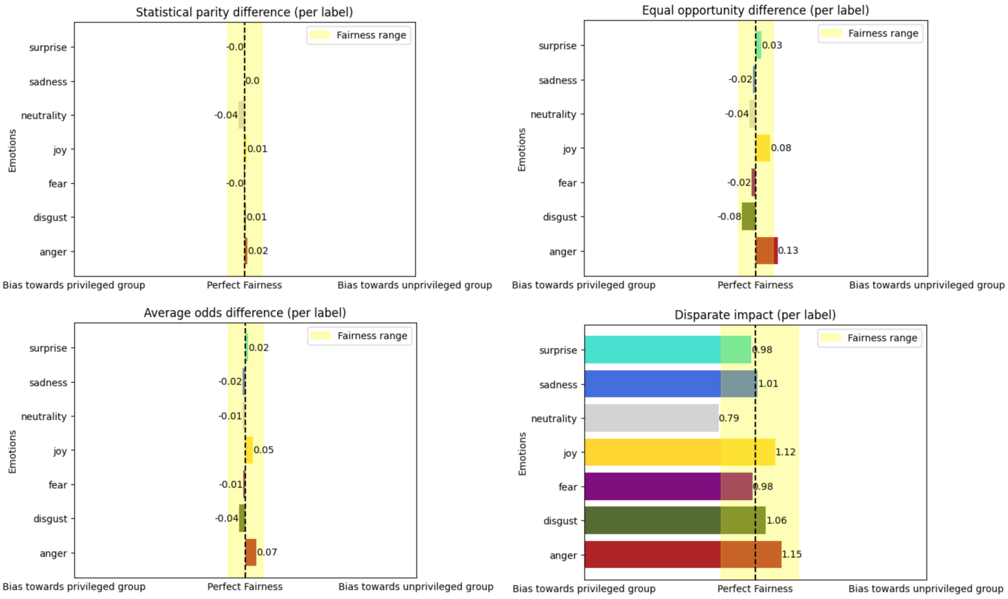

Figure 4.17: Emozionalmente, WavLM, unbalanced training set, gender (accuracy: 90.07%)

threshold 27 I have the most balanced division, while using threshold 40 I have the most unbalanced division. Looking at the results obtained we immediately notice

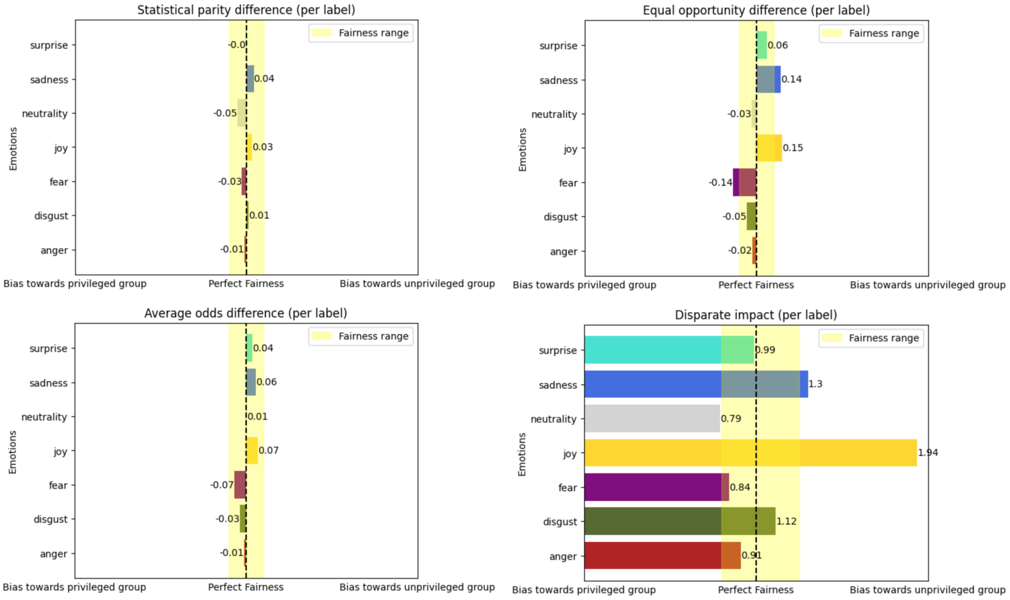

Figure 4.18: Emozionalmente, WavLM, balanced training set, gender (accuracy: 65.90%)

how, even having very low accuracy values, the biases are very small. This could be due to the fact that younger or older people do not have very different voice characteristics, as is the case between males and females, and therefore are treated equally by the model. Considering the three thresholds is equivalent to considering in the gender case balanced and unbalanced datasets: when we consider threshold 27 we are dealing with a balanced training set, while when we consider as threshold 30 or 40 we are dealing with unbalanced training sets, predominantly young people. Looking at the Tables 4.17, 4.18 and 4.19 it is immediately apparent how with unbalanced datasets the biases increase. However, there are no relevant results on the direction of the biases.

	Joy	Surprise	Disgust	Anger	Fear	Sadness	Neutrality
Statistical Parity	0.03	-0.01	0.01	-0.02	0.02	-0.01	-0.01
Equal Opportunity	0.01	-0.08	-0.02	-0.02	0.08	0.08	0
Average Odds	-0.01	-0.04	-0.02	0	0.04	0.04	0.01
Disparate Impact	1.21	0.9	1.1	0.87	1.23	1.23	0.93

Table 4.17: MFCC, 30 coefficients, age threshold 27

	Joy	Surprise	Disgust	Anger	Fear	Sadness	Neutrality
Statistical Parity	0.02	-0.04	0.03	0.03	0	-0.02	-0.01
Equal Opportunity	0.04	-0.17	0.08	0.01	-0.05	0.08	0.04
Average Odds	0.02	-0.08	0.03	-0.01	-0.02	0.06	0.03
Disparate Impact	1.15	0.75	1.27	1.18	0.98	0.87	0.94

Table 4.18: MFCC, 30 coefficients, age threshold 30

	Joy	Surprise	Disgust	Anger	Fear	Sadness	Neutrality
Statistical Parity	0.01	-0.06	0.01	0.01	-0.01	0.02	0.02
Equal Opportunity	0.07	-0.16	0.03	-0.01	-0.01	0.08	0.06
Average Odds	0.03	-0.06	0.01	-0.01	0	0.03	0.03
Disparate Impact	1.1	0.66	1.03	1.18	0.94	1.15	1.12

Table 4.19: MFCC, 30 coefficients, age threshold 40

Chapter 5

Conclusion

The focus of this work was to study and quantify bias in Speech Emotion Recognition systems. This was done by implementing four fairness metrics (Statistical Parity Difference, Equal Opportunity Difference, Average Odds Difference and Disparate Impact Ratio) by expanding their definition to apply to a multiclass scenario. Two baselines, SVM and ResNet, were implemented to carry out the tests. Also a pre-trained model with significantly better performance was used to perform experiments, WavLM. The tests performed had three purposes: to assess and quantify the bias of the models considered, to check whether there was a link between accuracy and bias, and to analyze the impact that the distribution of sensitive attributes in the training set has on the results.

The results obtained suggest a strong link between accuracy and bias obtained: as accuracy increases, there is a sharp decrease in bias. This is probably related to the model's ability to generalize the available data. Precisely because of this strong link to accuracy, it was not possible to identify clear trends suggesting a link between the distribution of the sensitive attribute in the training set and the amount and direction of bias obtained. In addition to this, the data available to create balanced or unbalanced datasets were limited, and as a result, the sets used for testing also had data distributions that may have influenced the results obtained.

Future work could focus on implementing baselines with more satisfactory accuracy results, but also on expanding the amount of data and datasets available in Italian for this task.

Appendix A

Fairness Metrics

```python
from collections import defaultdict
from sklearn.metrics import recall_score, multilabel_confusion_matrix

def statistical_parity_difference(y_true, y_pred, sensitive_attribute, labels):

    '''
      y_true: is the array containing the ground truth
      y_pred: is the array containing the predicted labels
      sensitive_attribute: is an array keeping track of the sensitive attribute corresponding to that instance (binary)
      NB: for the sensitive attribute we label as 0 the unprivileged group and 1 the privileged group
      labels: list with the possible output labels

    '''


    # Ensure the lengths of the input arrays are the same
    assert len(y_true) == len(y_pred) == len(sensitive_attribute), "Input arrays must have the same length"

    # Create a dictionary to store positive predictions for each label and sensitive attribute value
    positive_predictions = {0: {key: 0 for key in labels}, 1: {key: 0 for key in labels}}

    # Count positive predictions for each label and sensitive attribute value and store in the dictionary
    # positive_predictions = {"0": {"tristezza": 10, "gioia": 2, ecc...}, "1": {"tristezza": 4, "gioia": 5, ecc...}}
```

```
  for i, value in enumerate(sensitive_attribute):
      label = y_pred[i]
      positive_predictions[value][label] += 1

  # Calculate the proportion of positive predictions for each label and sensitive attribute value
  # proportions = {"0": {"tristezza": 0.3, "gioia": 0.1, ecc...}, "1": {"tristezza": 0.5, "gioia": 0.4, ecc...}}
  proportions = defaultdict(dict)
  for value, counts in positive_predictions.items():
      for label, count in counts.items():
          proportions[value][label] = count / sensitive_attribute.count(value)

  # Calculate statistical parity difference for each label
  # emotion_differences = {"tristezza": -0.1, "gioia": 0.4, ecc...}
  emotion_differences = {}
  for emotion in labels:
    difference = proportions[0].get(emotion, 0) - proportions[1].get(emotion, 0)
    emotion_differences[emotion] = difference

  return emotion_differences


############


def equal_opportunity_difference(y_true, y_pred, sensitive_attribute, labels):

  '''

    y_true: is the array containing the ground truth
    y_pred: is the array containing the predicted labels
    sensitive_attribute: is an array keeping track of the sensitive attribute corresponding to that instance (binary)
    NB: for the sensitive attribute we label as 0 the unprivileged group and 1 the privileged group
    labels: list with the possible output labels
    NB: in case we have a division by 0 we return 0

  '''


  # Ensure the lengths of the input arrays are the same
  assert len(y_true) == len(y_pred) == len(sensitive_attribute), "Input arrays must have the same length"

  # Create a dictionary to store y_pred and y_true separately with respect to the sensitive attribute
  sensitive_dict = {0: {'y_true': [], 'y_pred': []},
                    1: {'y_true': [], 'y_pred': []}}

  for i, value in enumerate(sensitive_attribute):
    true_label = y_true[i]
    predicted_label = y_pred[i]
    sensitive_dict[value]['y_true'].append(true_label)
    sensitive_dict[value]['y_pred'].append(predicted_label)

  # Compute recall score (True Positive Rate) for each label divided by sensitive attribute (obtain a list with the recall score for each label in labels)
  y_true_0 = sensitive_dict[0]['y_true']
  y_pred_0 = sensitive_dict[0]['y_pred']

  y_true_1 = sensitive_dict[1]['y_true']
  y_pred_1 = sensitive_dict[1]['y_pred']

  recall_0 = recall_score(y_true_0, y_pred_0, labels=labels, average=None, zero_division=0.0)
  recall_1 = recall_score(y_true_1, y_pred_1, labels=labels, average=None, zero_division=0.0)

  # Create a dictionary to store the recall score for each label divided by sensitive attribute
  # recall_dict = {"0": {"tristezza": 0.6, "gioia": 0.5, ecc...}, "1": {"tristezza": 0.4, "gioia": 0.3, ecc...}}
  recall_dict = {0: {}, 1: {}}

  for label, recall_score_value in zip(labels, recall_0):
    recall_dict[0][label] = recall_score_value

  for label, recall_score_value in zip(labels, recall_1):
    recall_dict[1][label] = recall_score_value

  # Calculate equal opportunity difference for each label
  # emotion_differences = {"tristezza": -0.1, "gioia": 0.4, ecc...}
  emotion_differences = {}
  for emotion in labels:
    difference = recall_dict[0].get(emotion, 0) - recall_dict[1].get(emotion, 0)
    emotion_differences[emotion] = difference

  return emotion_differences


############
```

```python


def average_odds_difference(y_true, y_pred, sensitive_attribute, labels):
  '''

    y_true: is the array containing the ground truth
    y_pred: is the array containing the predicted labels
    sensitive_attribute: is an array keeping track of the sensitive attribute corresponding to that instance (binary)
    NB: for the sensitive attribute we label as 0 the unprivileged group and 1 the privileged group
    labels: list with the possible output labels

  '''

  # Ensure the lengths of the input arrays are the same
  assert len(y_true) == len(y_pred) == len(sensitive_attribute), "Input arrays must have the same length"

  # Create a dictionary to store y_pred and y_true separately with respect to the sensitive attribute
  sensitive_dict = {0: {'y_true': [], 'y_pred': []},
                    1: {'y_true': [], 'y_pred': []}}

  for i, value in enumerate(sensitive_attribute):
    true_label = y_true[i]
    predicted_label = y_pred[i]
    sensitive_dict[value]['y_true'].append(true_label)
    sensitive_dict[value]['y_pred'].append(predicted_label)

  # Compute recall score (True Positive Rate) for each label divided by sensitive attribute (obtain a list with the recall score for each label in labels)
  y_true_0 = sensitive_dict[0]['y_true']
  y_pred_0 = sensitive_dict[0]['y_pred']

  y_true_1 = sensitive_dict[1]['y_true']
  y_pred_1 = sensitive_dict[1]['y_pred']

  recall_0 = recall_score(y_true_0, y_pred_0, labels=labels, average=None)
  recall_1 = recall_score(y_true_1, y_pred_1, labels=labels, average=None)

  # Create a dictionary to store the recall score (True Positive Rate) for each label divided by sensitive attribute
  # tprs = {"0": {"tristezza": 0.6, "gioia": 0.5, ecc...}, "1": {"tristezza": 0.4, "gioia": 0.3, ecc...}}
  tprs = {0: {}, 1: {}}

  for label, recall_score_value in zip(labels, recall_0):
    tprs[0][label] = recall_score_value

  for label, recall_score_value in zip(labels, recall_1):
    tprs[1][label] = recall_score_value

  # Calculate TPRs difference for each label
  # tprs_diff = {"tristezza": -0.1, "gioia": 0.4, ecc...}
  tprs_diff = {}
  for emotion in labels:
    difference = tprs[0].get(emotion, 0) - tprs[1].get(emotion, 0)
    tprs_diff[emotion] = difference

  # Store in a dictionary the False Positive Rate
  # First we need to compute the confusion matrices for each label and senstiive attribute to extract FP and TN values
  confusion_matrices_0 = multilabel_confusion_matrix(y_true_0, y_pred_0, labels=labels)
  tn_0, fp_0 = confusion_matrices_0[:, 0, 0], confusion_matrices_0[:, 0, 1]

  confusion_matrices_1 = multilabel_confusion_matrix(y_true_1, y_pred_1, labels=labels)
  tn_1, fp_1 = confusion_matrices_1[:, 0, 0], confusion_matrices_1[:, 0, 1]

  # Create a dictionary to store the False Positive Rate for each label divided by sensitive attribute
  # fprs = {"0": {"tristezza": 0.6, "gioia": 0.5, ecc...}, "1": {"tristezza": 0.4, "gioia": 0.3, ecc...}}
  fprs = {0: {}, 1: {}}

  for label, tn_value, fp_value in zip(labels, tn_0, fp_0):
    fpr = fp_value / (fp_value + tn_value)
    fprs[0][label] = fpr

  for label, tn_value, fp_value in zip(labels, tn_1, fp_1):
    fpr = fp_value / (fp_value + tn_value)
    fprs[1][label] = fpr

  # Calculate FPRs difference for each label
  # fprs_diff = {"tristezza": -0.1, "gioia": 0.4, ecc...}
  fprs_diff = {}
  for emotion in labels:
    difference = fprs[0].get(emotion, 0) - fprs[1].get(emotion, 0)
```

```python
    fprs_diff[emotion] = difference

  # Calculate the average odds difference by computing the average between TPRs differences and FPRs differences
  # avg_odds = {"tristezza": 0.2, "gioia": 0.3, ecc...}
  avg_odds = {}
  for emotion in labels:
    avg = (tprs_diff.get(emotion, 0) - fprs_diff.get(emotion, 0)) / 2
    avg_odds[emotion] = avg

  return avg_odds


############


def disparate_impact_ratio(y_true, y_pred, sensitive_attribute, labels):

  '''

    y_true: is the array containing the ground truth
    y_pred: is the array containing the predicted labels
    sensitive_attribute: is an array keeping track of the sensitive attribute corresponding to that instance (binary)
    NB: for the sensitive attribute we label as 0 the unprivileged group and 1 the privileged group
    labels: list with the possible output labels
    NB: in case we have a division by 0 we don't compute the ratio but return 0

  '''

  # Ensure the lengths of the input arrays are the same
  assert len(y_true) == len(y_pred) == len(sensitive_attribute), "Input arrays must have the same length"

  # Create a dictionary to store positive predictions for each label and sensitive attribute value
  positive_predictions = {0: {key: 0 for key in labels}, 1: {key: 0 for key in labels}}

  # Count positive predictions for each label and sensitive attribute value and store in the dictionary
  # positive_predictions = {"0": {"tristezza": 10, "gioia": 2, ecc...}, "1": {"tristezza": 4, "gioia": 5, ecc...}}
  for i, value in enumerate(sensitive_attribute):
      label = y_pred[i]
      positive_predictions[value][label] += 1


  # Calculate the proportion of positive predictions for each label and sensitive attribute value
  # proportions = {"0": {"tristezza": 0.3, "gioia": 0.1, ecc...}, "1": {"tristezza": 0.5, "gioia": 0.4, ecc...}}
  proportions = defaultdict(dict)
  for value, counts in positive_predictions.items():
      for label, count in counts.items():
          proportions[value][label] = count / sensitive_attribute.count(value)


  # Compute the ratio obtained comparing the two groups and store it in the dictionary
  # ratios = {"tristezza": 0.4, "gioia": 0.3, ecc...}
  ratios = {}
  for emotion in labels:
    if proportions[1].get(emotion, 0) == 0:
      ratio = 0
    else:
      ratio = proportions[0].get(emotion, 0) / proportions[1].get(emotion, 0)
    ratios[emotion] = ratio

  return ratios
```

Appendix B

Plots

```python

import matplotlib.pyplot as plt


def plot_statistical_parity(data, dataset):

    '''

        data: should be a dictionary

    '''

    emotions = list(data.keys())
    values = list(data.values())

    if dataset == 'emovo':
        emotion_colors = {
            'gioia': 'gold',
            'neutralità': 'lightgray',
            'rabbia': 'firebrick',
            'tristezza': 'royalblue',
            'disgusto': 'darkolivegreen',
            'sorpresa': 'turquoise',
            'paura': 'purple'}

    elif dataset == 'emozionalmente':
        emotion_colors = {
            'joy': 'gold',
            'neutrality': 'lightgray',
            'anger': 'firebrick',
            'sadness': 'royalblue',
            'disgust': 'darkolivegreen',
            'surprise': 'turquoise',
            'fear': 'purple'}


    fig, ax = plt.subplots()

    bars = ax.barh(emotions, values, color=[emotion_colors[emotion] for emotion in emotions])

    ax.axvline(0, color='black', linestyle='--')

    ax.axvspan(-0.1, 0.1, alpha=0.3, color='yellow', label='Fairness range')

    for bar, value in zip(bars, values):
        if value < 0:
            ax.text(value, bar.get_y() + bar.get_height() / 2, f'{round(value, 2)}', ha='right', va='center')
        else:
            ax.text(value, bar.get_y() + bar.get_height() / 2, f'{round(value, 2)}', ha='left', va='center')


    ax.text(0, -1, 'Perfect Fairness', ha='center', va='center', color='black')
    ax.text(1, -1, 'Bias towards unprivileged group', ha='center', va='center', color='black')
    ax.text(-1, -1, 'Bias towards privileged group', ha='center', va='center', color='black')
    # NB: A value of < 0 implies higher benefit for the privileged group and
    # a value > 0 implies higher benefit for the unprivileged group

    ax.set_ylabel('Emotions')

    ax.set_title('Statistical parity difference (per label)')

    ax.set_xticks([])

    ax.set_xlim(-1, 1)

    ax.legend()

    plt.show()


##############

```

```

def plot_equal_opportunity(data, dataset):

    '''

      data: should be a dictionary

    '''

    emotions = list(data.keys())
    values = list(data.values())


    if dataset == 'emovo':
        emotion_colors = {
            'gioia': 'gold',
            'neutralità': 'lightgray',
            'rabbia': 'firebrick',
            'tristezza': 'royalblue',
            'disgusto': 'darkolivegreen',
            'sorpresa': 'turquoise',
            'paura': 'purple'}

    elif dataset == 'emozionalmente':
        emotion_colors = {
            'joy': 'gold',
            'neutrality': 'lightgray',
            'anger': 'firebrick',
            'sadness': 'royalblue',
            'disgust': 'darkolivegreen',
            'surprise': 'turquoise',
            'fear': 'purple'}


    fig, ax = plt.subplots()

    bars = ax.barh(emotions, values, color=[emotion_colors[emotion] for emotion in emotions])

    ax.axvline(0, color='black', linestyle='--')

    ax.axvspan(-0.1, 0.1, alpha=0.3, color='yellow', label='Fairness range')

    for bar, value in zip(bars, values):
        if value < 0:
            ax.text(value, bar.get_y() + bar.get_height() / 2, f'{round(value, 2)}', ha='right', va='center')
        else:
```

```
        ax.text(value, bar.get_y() + bar.get_height() / 2, f'{round(value, 2)}', ha='left', va='center')


    ax.text(0, -1, 'Perfect Fairness', ha='center', va='center', color='black')
    ax.text(1, -1, 'Bias towards unprivileged group', ha='center', va='center', color='black')
    ax.text(-1, -1, 'Bias towards privileged group', ha='center', va='center', color='black')
    # NB: A value of < 0 implies higher benefit for the privileged group and
    # a value > 0 implies higher benefit for the unprivileged group

    ax.set_ylabel('Emotions')

    ax.set_title('Equal opportunity difference (per label)')

    ax.set_xticks([])

    ax.set_xlim(-1, 1)

    ax.legend()

    plt.show()


##############


def plot_average_odds(data, dataset):

    '''

      data: should be a dictionary

    '''

    emotions = list(data.keys())
    values = list(data.values())


    if dataset == 'emovo':
        emotion_colors = {
            'gioia': 'gold',
            'neutralità': 'lightgray',
            'rabbia': 'firebrick',
            'tristezza': 'royalblue',
            'disgusto': 'darkolivegreen',
            'sorpresa': 'turquoise',
            'paura': 'purple'}

    elif dataset == 'emozionalmente':
        emotion_colors = {
            'joy': 'gold',
            'neutrality': 'lightgray',
            'anger': 'firebrick',
            'sadness': 'royalblue',
            'disgust': 'darkolivegreen',
            'surprise': 'turquoise',
            'fear': 'purple'}


    fig, ax = plt.subplots()

    bars = ax.barh(emotions, values, color=[emotion_colors[emotion] for emotion in emotions])

    ax.axvline(0, color='black', linestyle='--')

    ax.axvspan(-0.1, 0.1, alpha=0.3, color='yellow', label='Fairness range')

    for bar, value in zip(bars, values):
        if value < 0:
            ax.text(value, bar.get_y() + bar.get_height() / 2, f'{round(value, 2)}', ha='right', va='center')
        else:
            ax.text(value, bar.get_y() + bar.get_height() / 2, f'{round(value, 2)}', ha='left', va='center')


    ax.text(0, -1, 'Perfect Fairness', ha='center', va='center', color='black')
    ax.text(1, -1, 'Bias towards unprivileged group', ha='center', va='center', color='black')
    ax.text(-1, -1, 'Bias towards privileged group', ha='center', va='center', color='black')
    # NB: A value of < 0 implies higher benefit for the privileged group and
    # a value > 0 implies higher benefit for the unprivileged group

    ax.set_ylabel('Emotions')

    ax.set_title('Average odds difference (per label)')

    ax.set_xticks([])

    ax.set_xlim(-1, 1)

    ax.legend()

    plt.show()


##############


def plot_disparate_impact(data, dataset):

    '''

        data: should be a dictionary

    '''

    emotions = list(data.keys())
    values = list(data.values())


    if dataset == 'emovo':
        emotion_colors = {
            'gioia': 'gold',
            'neutralità': 'lightgray',
            'rabbia': 'firebrick',
            'tristezza': 'royalblue',
            'disgusto': 'darkolivegreen',
            'sorpresa': 'turquoise',
            'paura': 'purple'}

    elif dataset == 'emozionalmente':
        emotion_colors = {
            'joy': 'gold',
            'neutrality': 'lightgray',
            'anger': 'firebrick',
            'sadness': 'royalblue',
            'disgust': 'darkolivegreen',
            'surprise': 'turquoise',
            'fear': 'purple'}

    fig, ax = plt.subplots()

    bars = ax.barh(emotions, values, color=[emotion_colors[emotion] for emotion in emotions])

    ax.axvline(1, color='black', linestyle='--')

    ax.axvspan(0.8, 1.25, alpha=0.3, color='yellow', label='Fairness range')

    for bar, value in zip(bars, values):
        if value < 0:
            ax.text(value, bar.get_y() + bar.get_height() / 2, f'{round(value, 2)}', ha='right', va='center')
        else:
            ax.text(value, bar.get_y() + bar.get_height() / 2, f'{round(value, 2)}', ha='left', va='center')


    ax.text(1, -1, 'Perfect Fairness', ha='center', va='center', color='black')
    ax.text(2, -1, 'Bias towards unprivileged group', ha='center', va='center', color='black')
    ax.text(0, -1, 'Bias towards privileged group', ha='center', va='center', color='black')
    # NB: A value < 1 implies higher benefit for the privileged group and
    # a value >1 implies a higher benefit for the unprivileged group

    ax.set_ylabel('Emotions')

    ax.set_title('Disparate impact (per label)')

    ax.set_xticks([])

    ax.set_xlim(0, 2)

    ax.legend()

    plt.show()
```

Appendix C

Fairness Metrics Plots

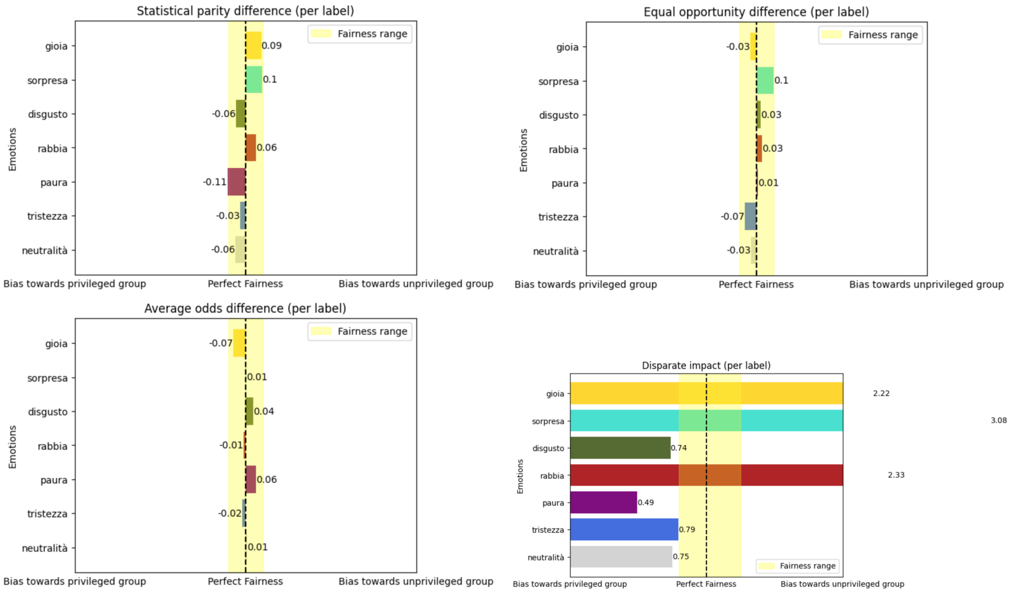

Figure C.1: EMOVO, SVM, MFCC, 12 coefficients, 5 kfold splits (mean accuracy: 41.14%)

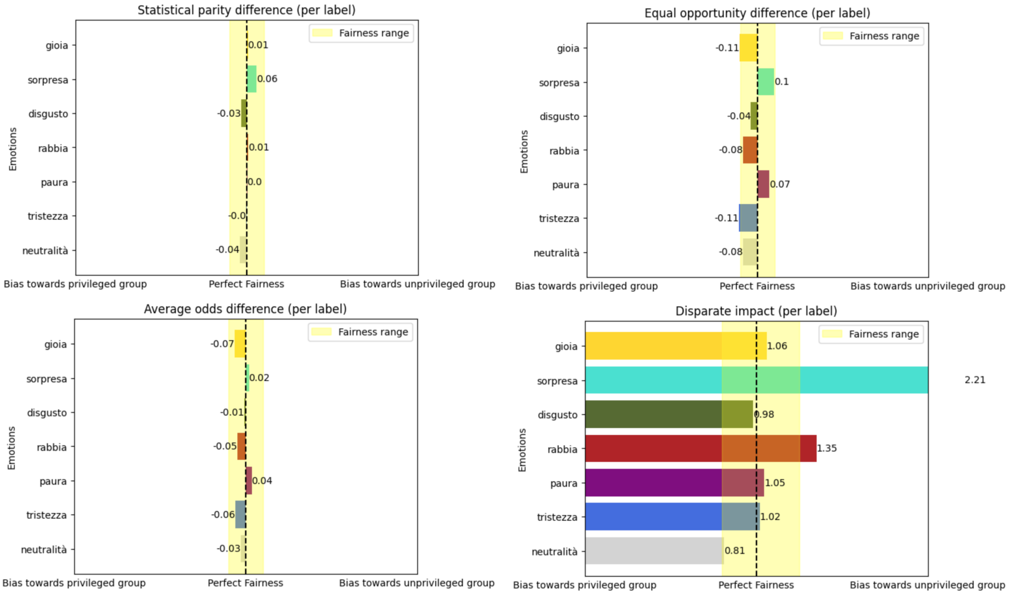

Figure C.2: EMOVO, SVM, MFCC, 24 coefficients, 5 kfold splits (mean accuracy: 60.19%)

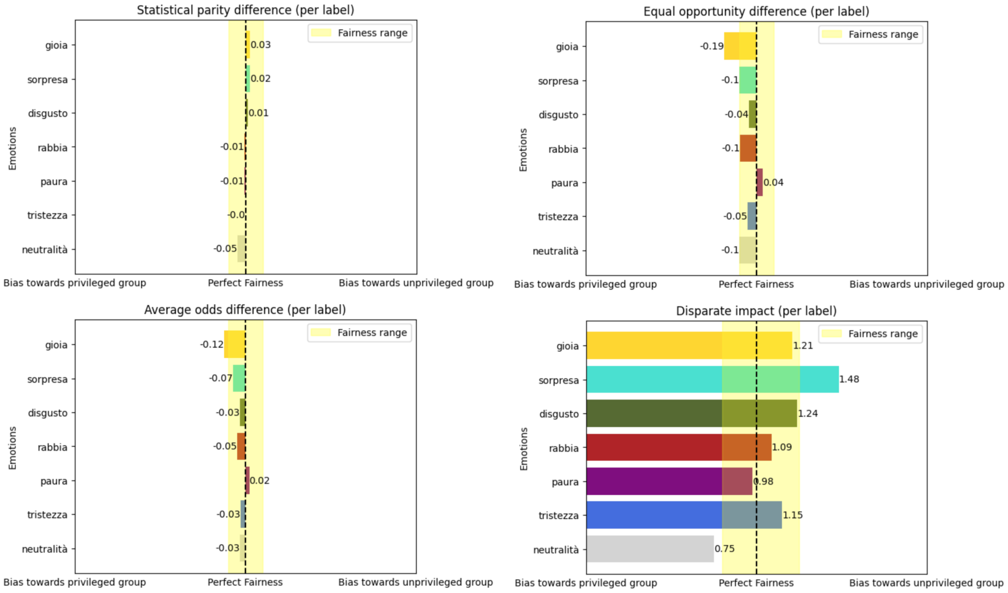

Figure C.3: EMOVO, SVM, MFCC, 30 coefficients, 5 kfold splits (mean accuracy: 64.44%)

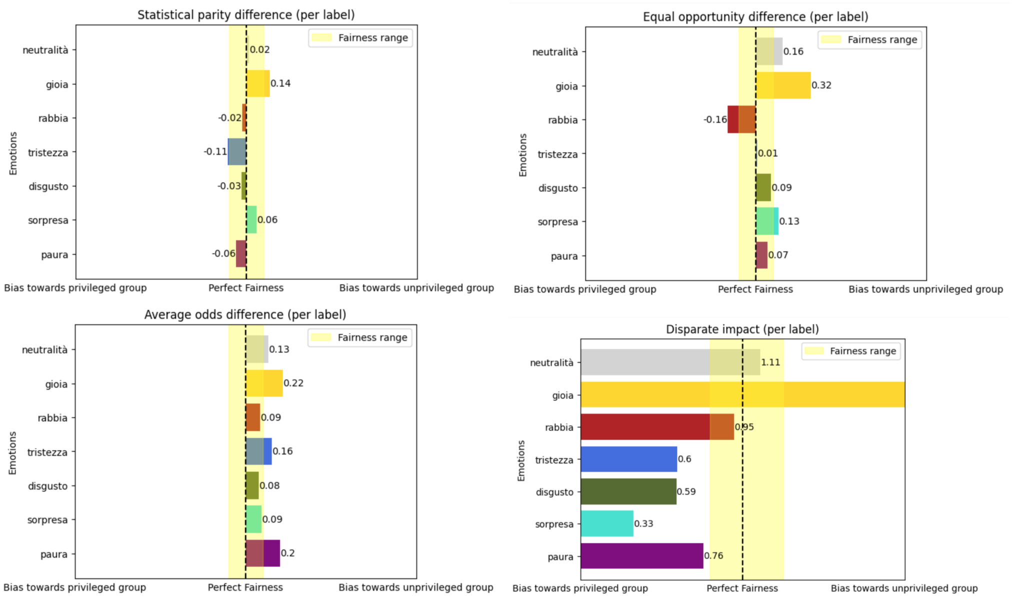

Figure C.4: EMOVO, SVM, MFMC, 12 coefficients, 5 kfold splits (mean accuracy: 33.33%)

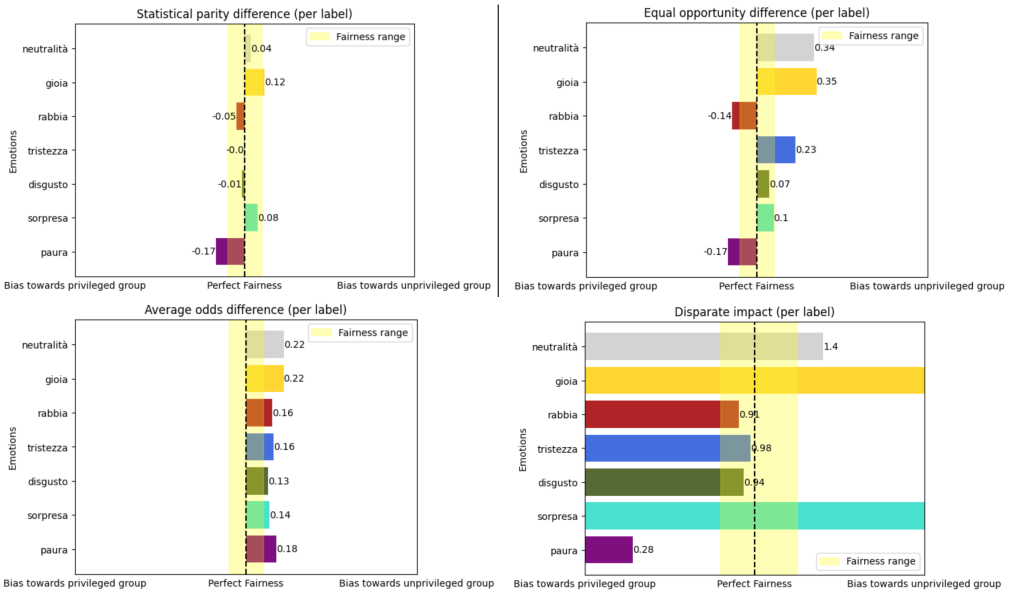

Figure C.5: EMOVO, SVM, MFMC, 24 coefficients, 5 kfold splits (mean accuracy: 41.14%)

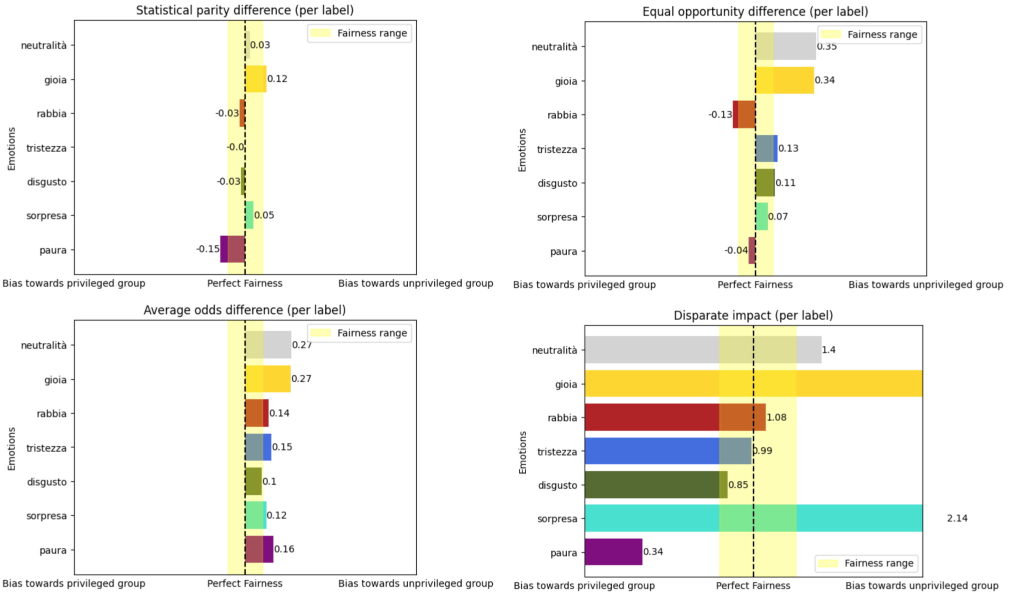

Figure C.6: EMOVO, SVM, MFMC, 30 coefficients, 5 kfold splits (mean accuracy: 43.86%)

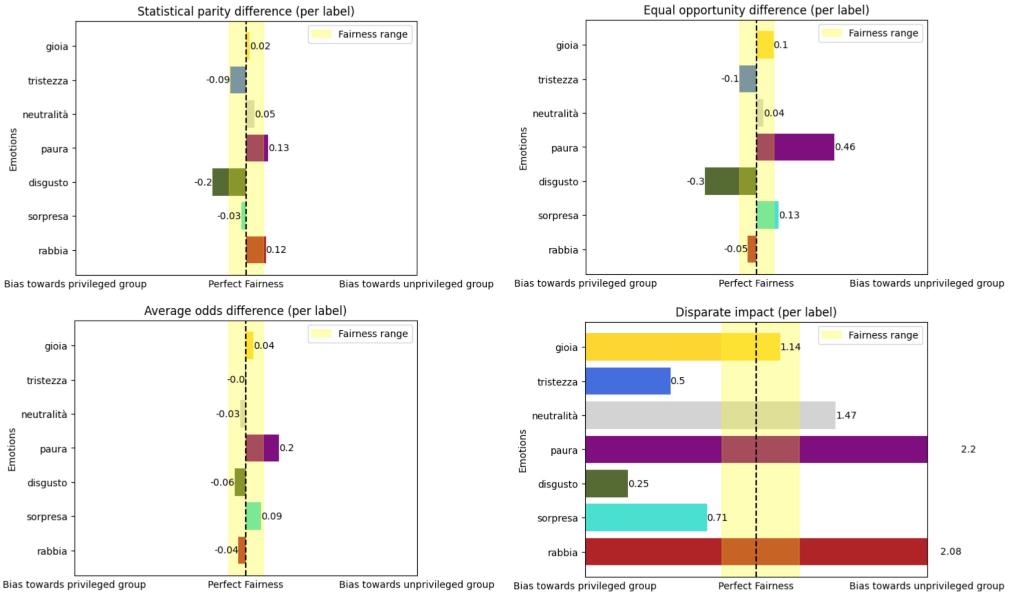

Figure C.7: EMOVO, ResNet, balanced training set (accuracy: 44.68%)

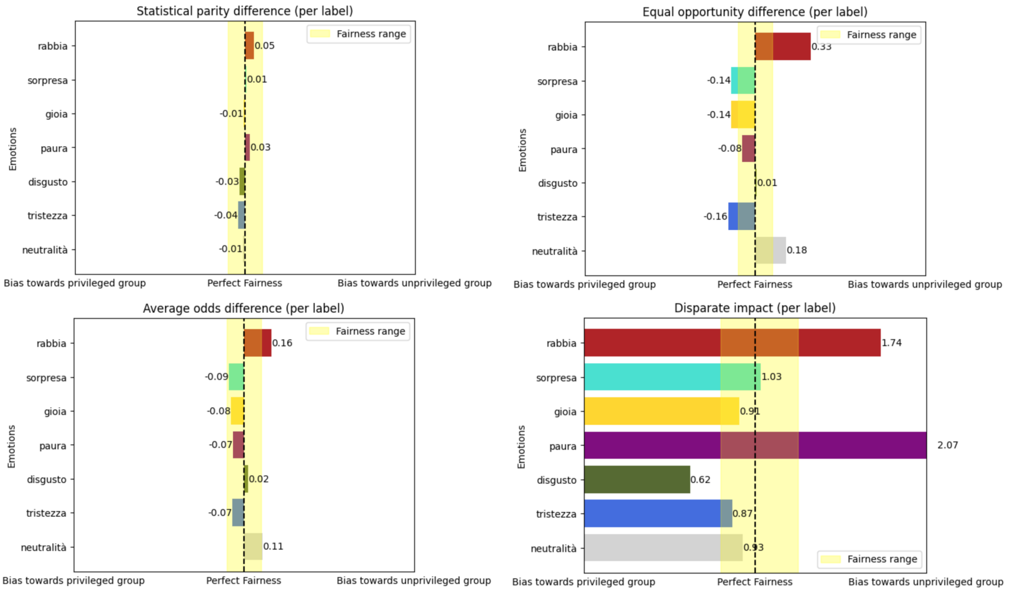

Figure C.8: EMOVO, ResNet, 70% male training set (accuracy: 36.01%)

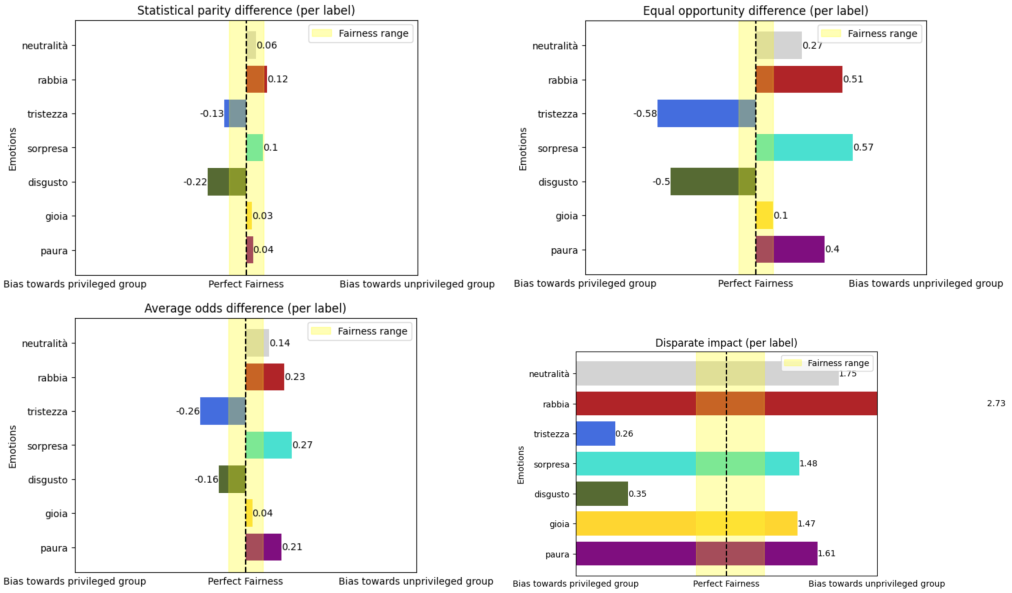

Figure C.9: EMOVO, ResNet, 70% female training set (accuracy: 38.55%)

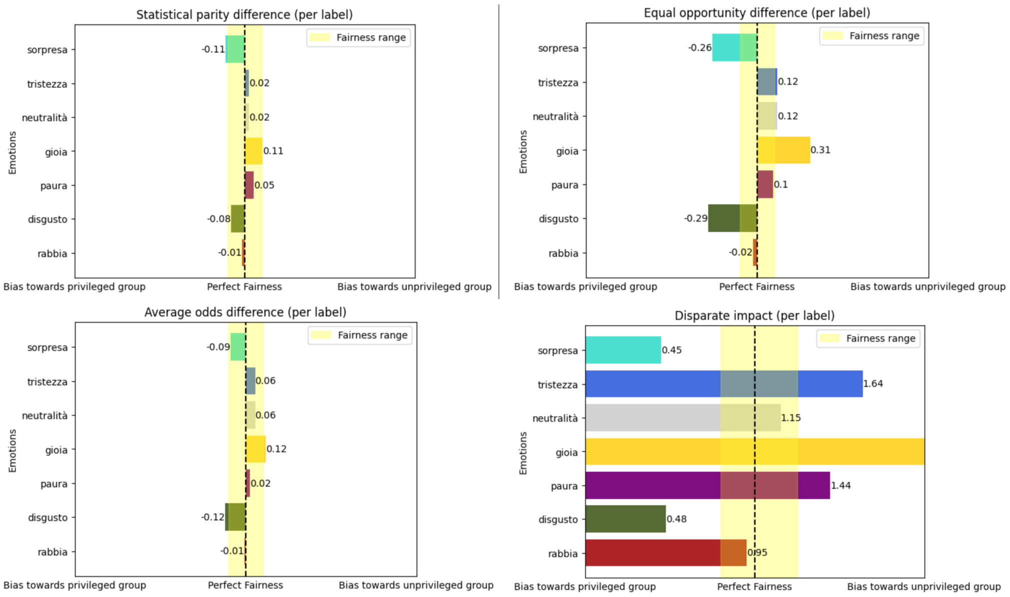

Figure C.10: WavLM, Emozionalmente unbalanced training set, EMOVO test set (accuracy: 64.11%)

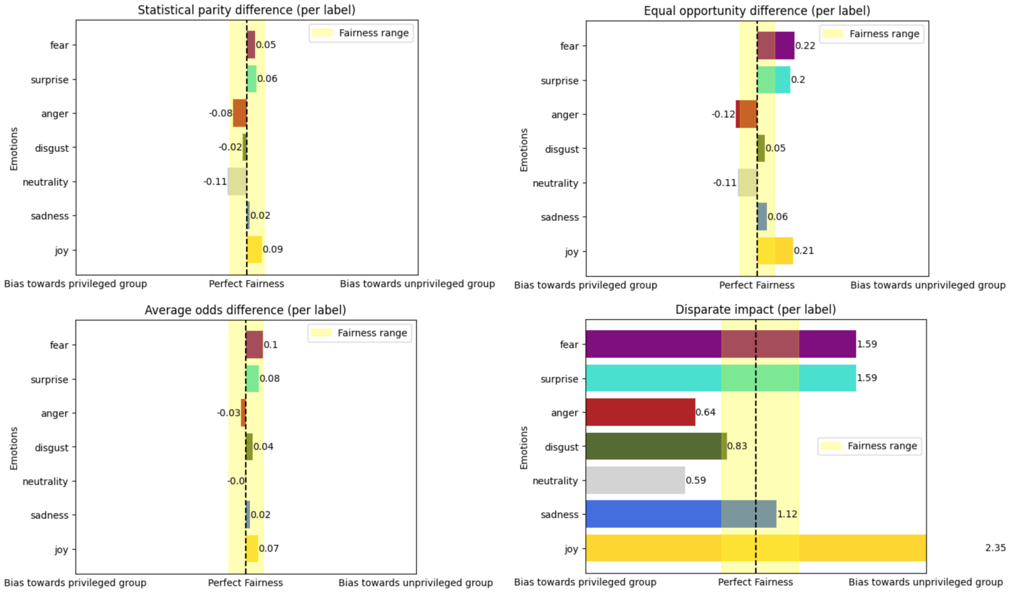

Figure C.11: Emozionalmente, SVM, MFCC, 30 coefficients, unbalanced training set, gender (accuracy: 32.02%)

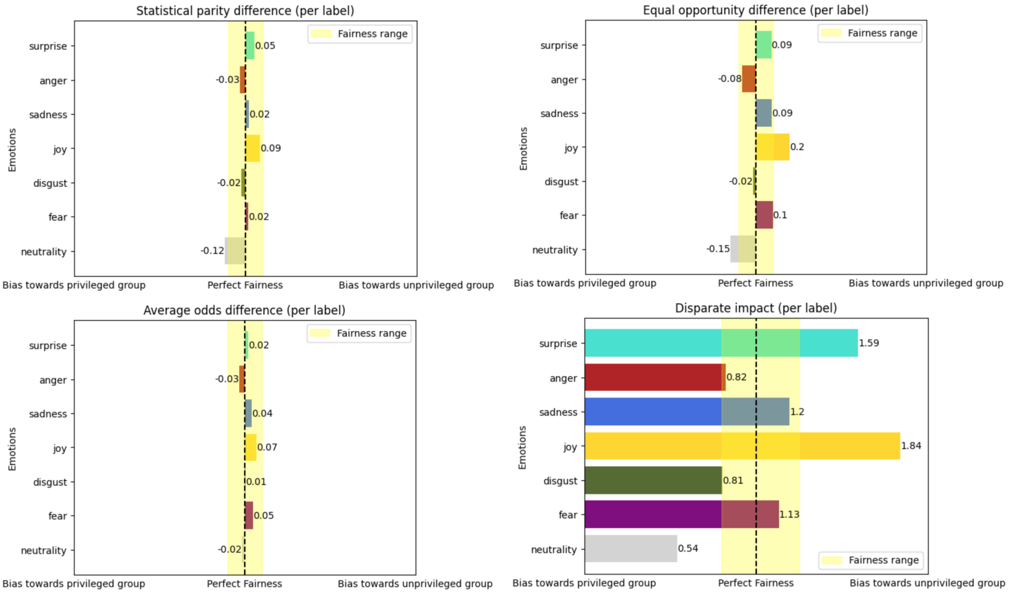

Figure C.12: Emozionalmente, SVM, MFCC, 30 coefficients, balanced training set, gender (accuracy: 31.76%)

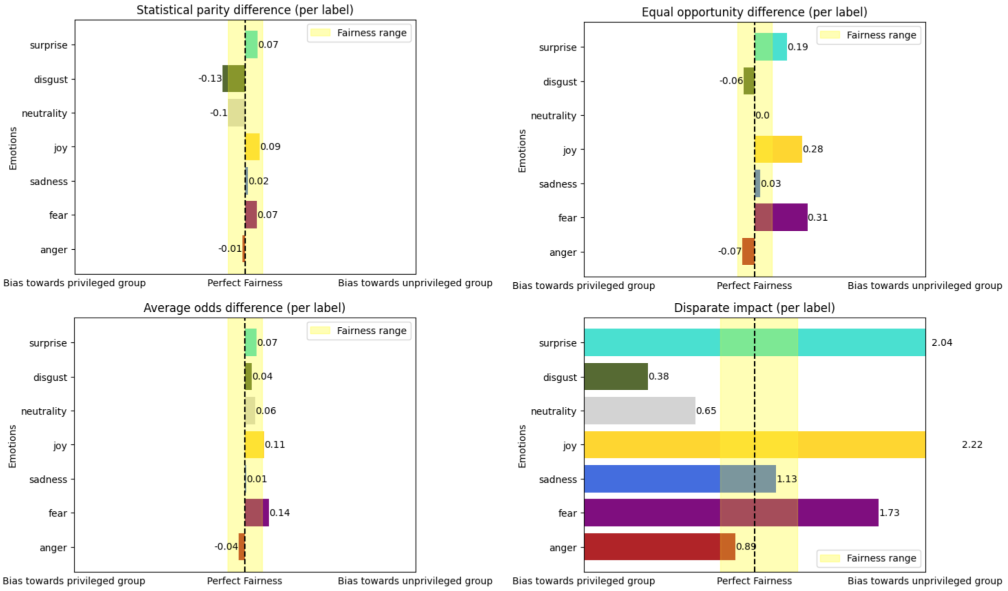

Figure C.13: Emozionalmente, SVM, MFMC, 30 coefficients, unbalanced training set, gender (accuracy: 31.54%)

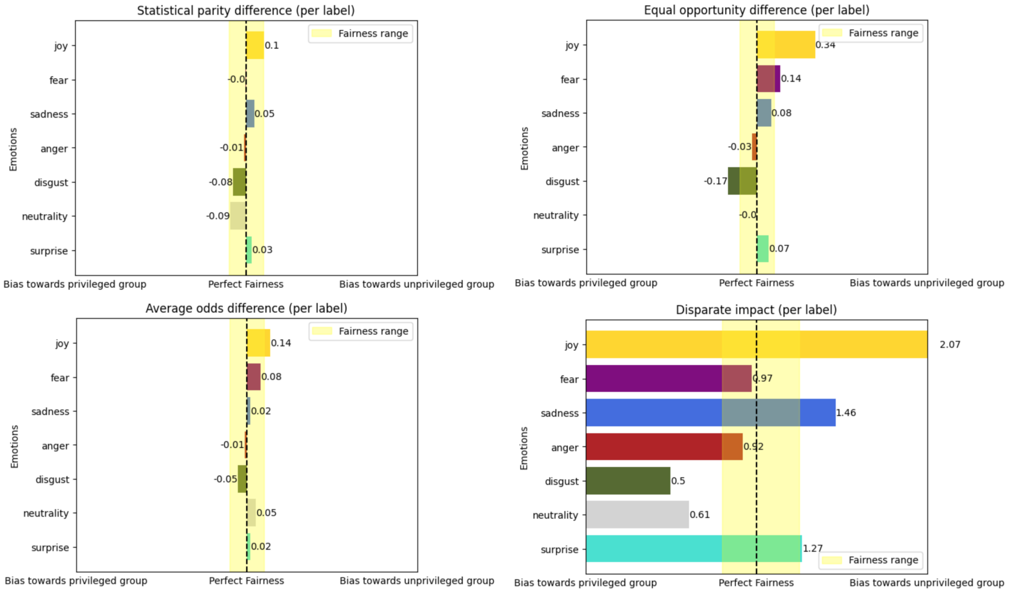

Figure C.14: Emozionalmente, SVM, MFMC, 30 coefficients, balanced training set, gender (accuracy: 30.12%)

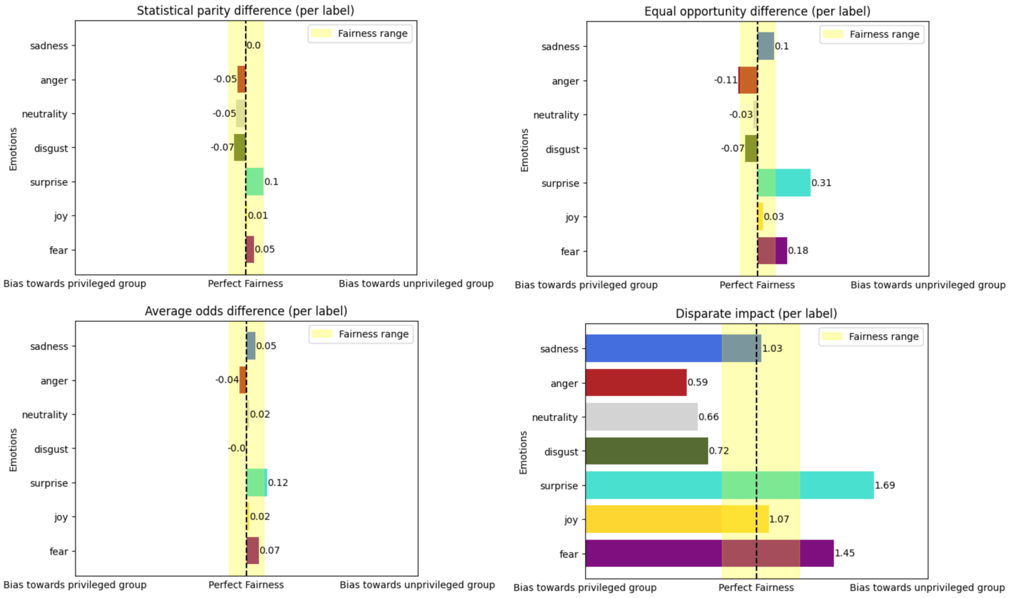

Figure C.15: Emozionalmente, ResNet, unbalanced training set, gender (accuracy: 33.51%)

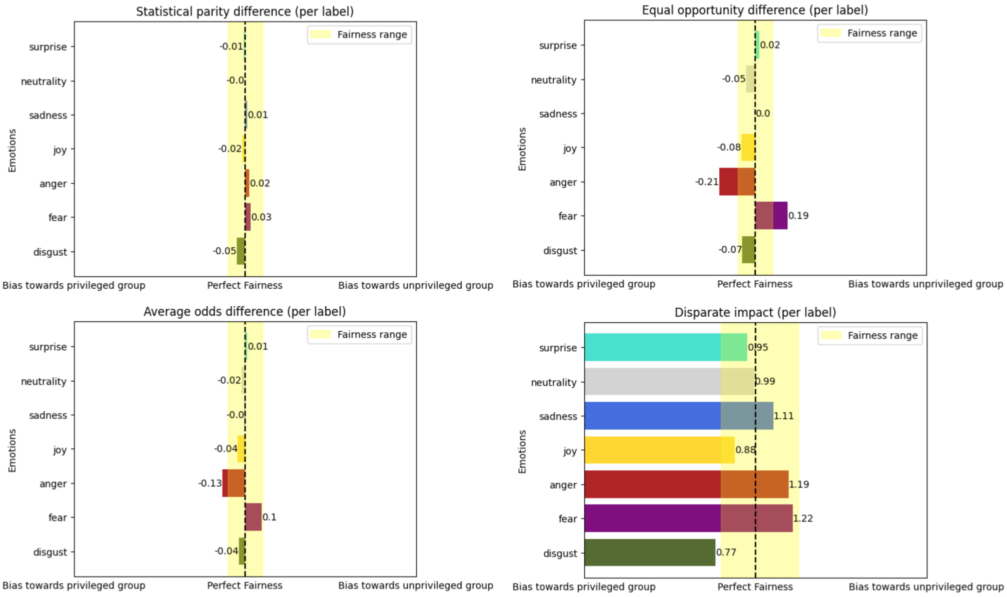

Figure C.16: Emozionalmente, ResNet, balanced training set, gender (accuracy: 39.95%)

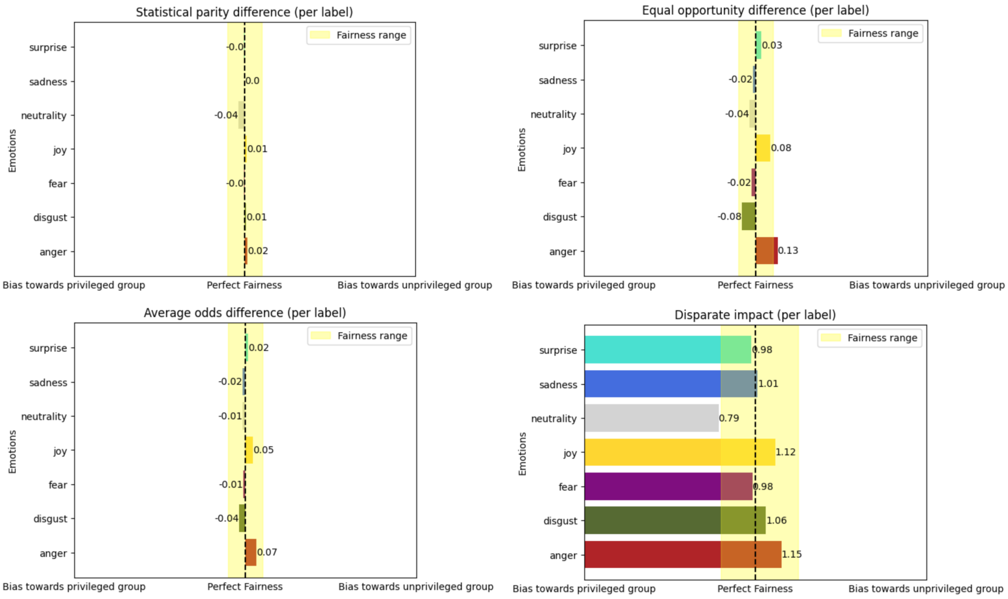

Figure C.17: Emozionalmente, WavLM, unbalanced training set, gender (accuracy: 90.07%)

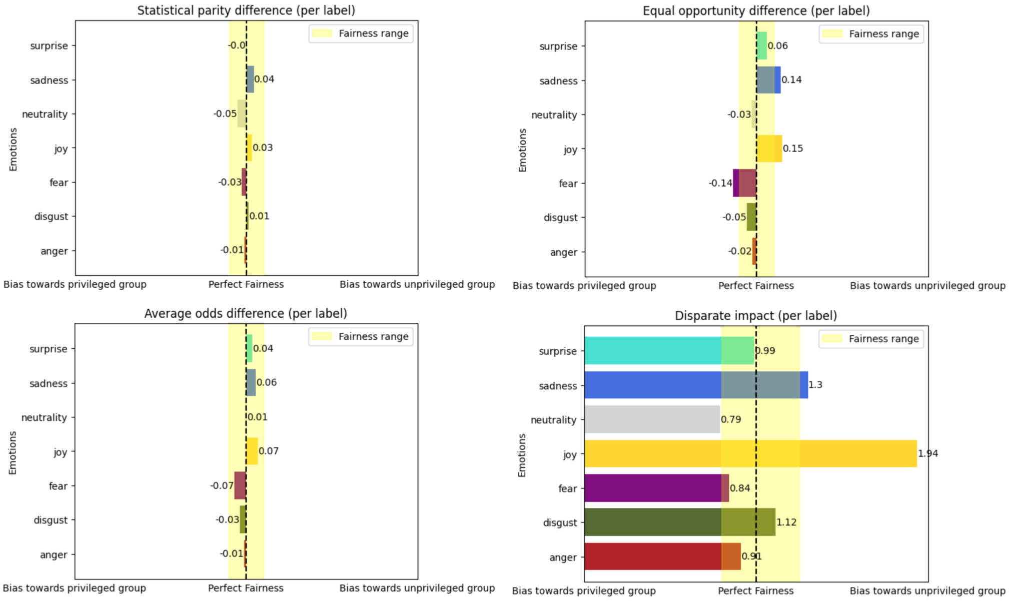

Figure C.18: Emozionalmente, WavLM, balanced training set, gender (accuracy: 65.90%)

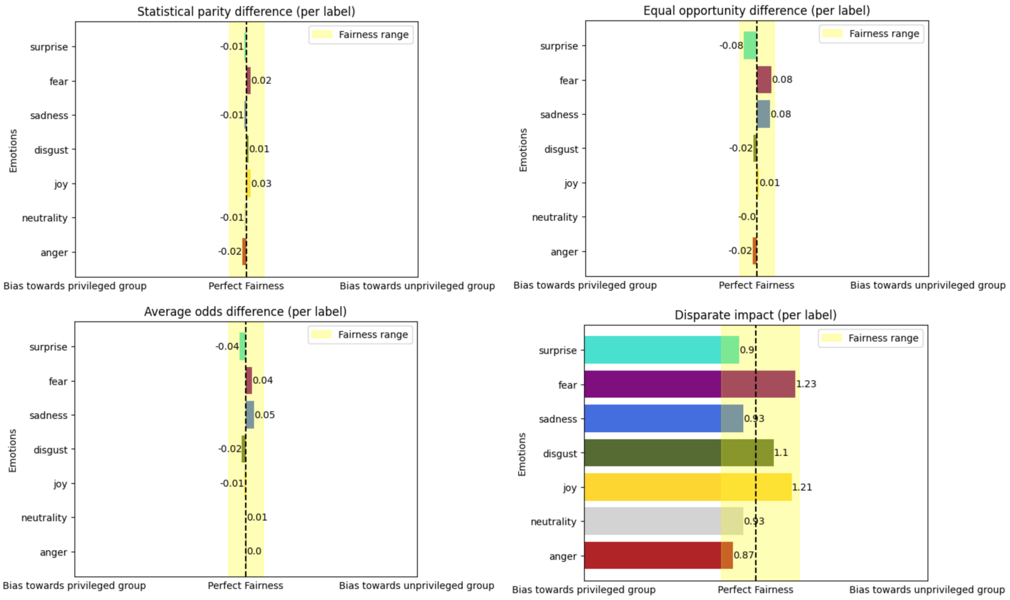

Figure C.19: Emozionalmente, SVM, MFCC, 30 coefficients, age threshold 27 (accuracy: 33.17%)

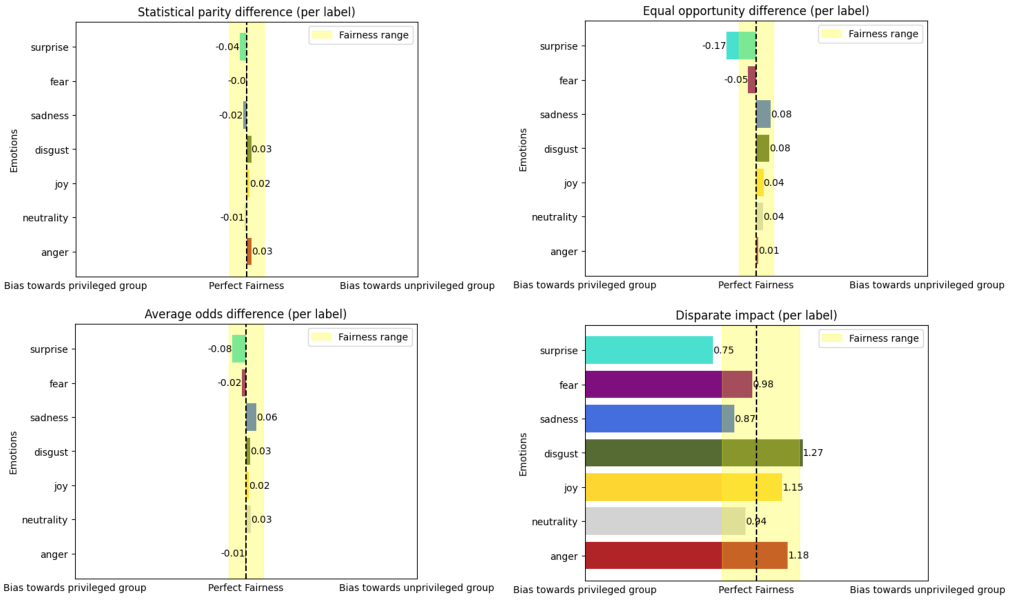

Figure C.20: Emozionalmente, SVM, MFCC, 30 coefficients, age threshold 30 (accuracy: 33.17%)

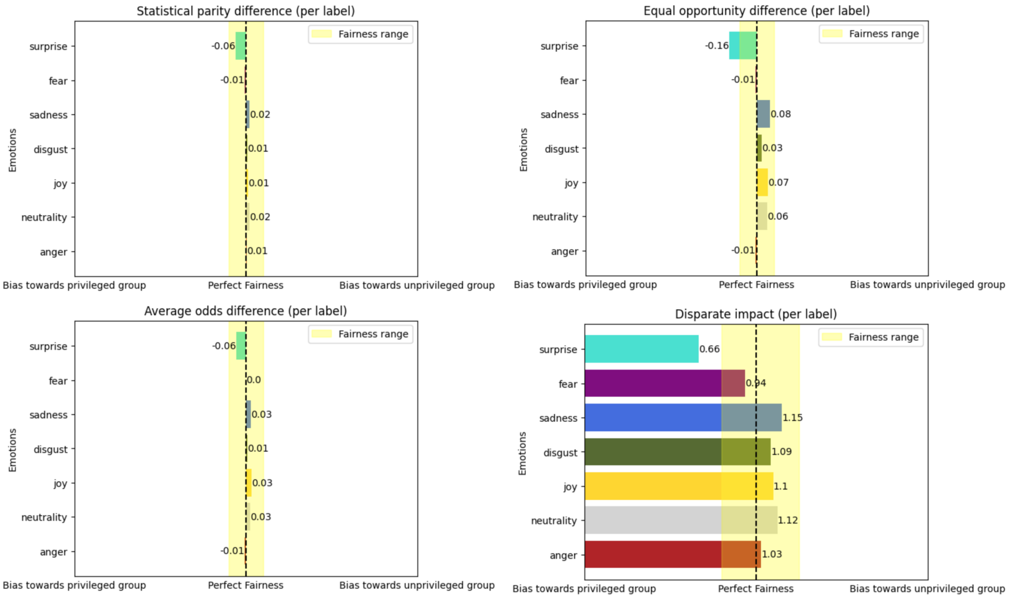

Figure C.21: Emozionalmente, SVM, MFCC, 30 coefficients, age threshold 40 (accuracy: 33.17%)

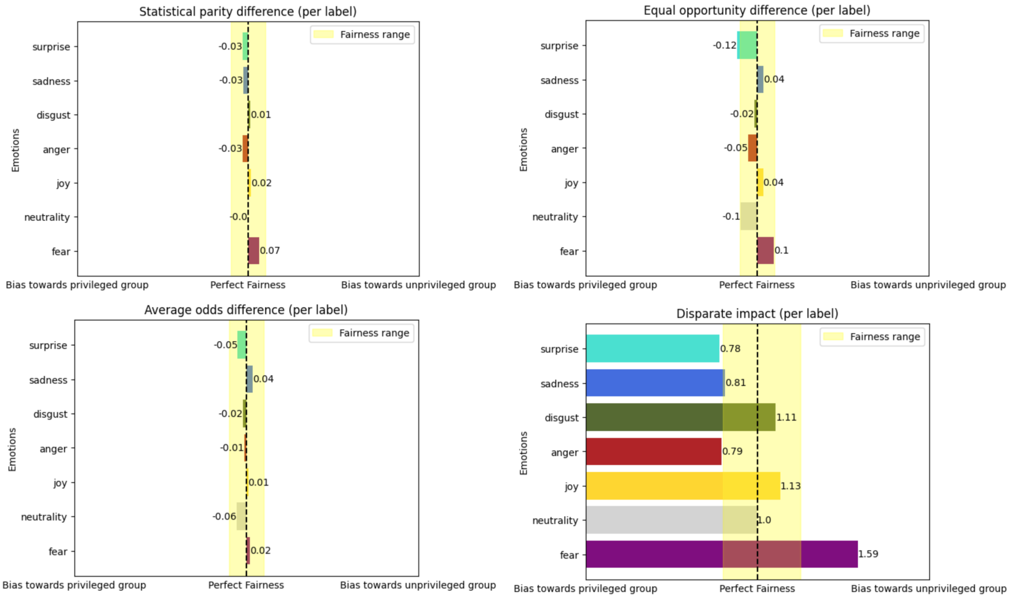

Figure C.22: Emozionalmente, SVM, MFMC, 30 coefficients, age threshold 27 (accuracy: 31.82%)

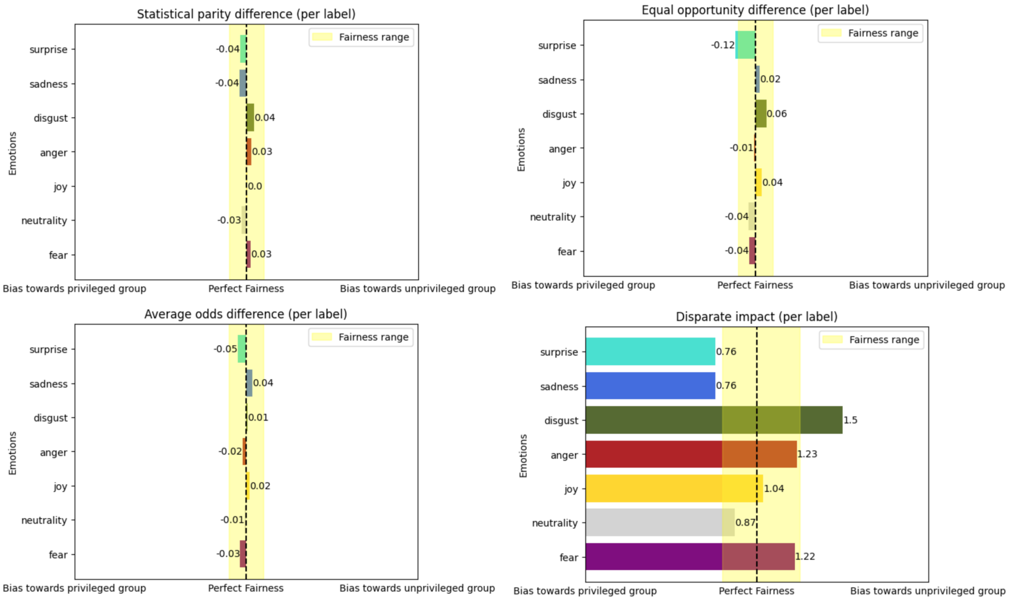

Figure C.23: Emozionalmente, SVM, MFMC, 30 coefficients, age threshold 30 (accuracy: 31.82%)

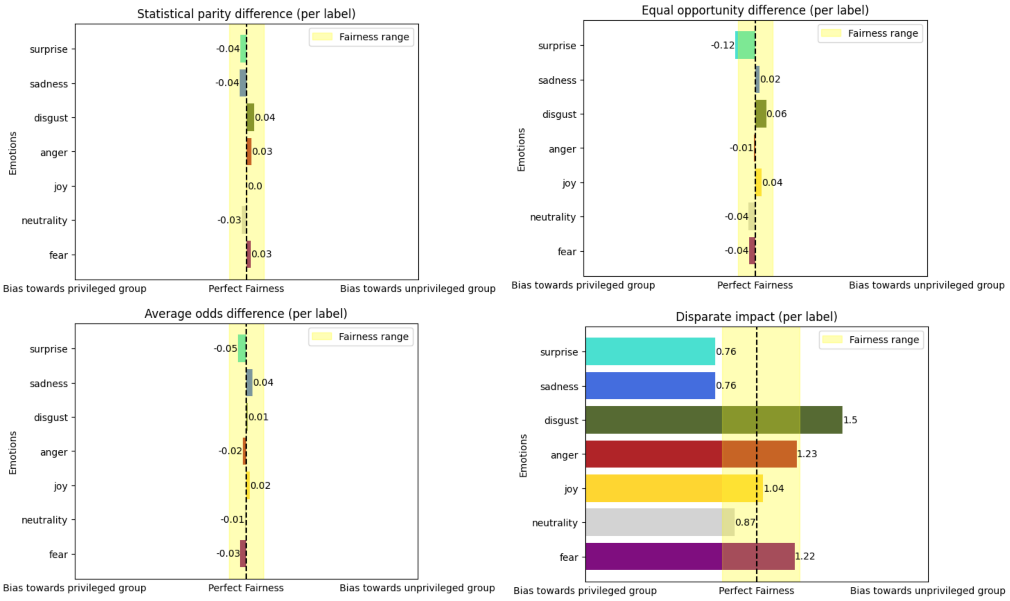

Figure C.24: Emozionalmente, SVM, MFMC, 30 coefficients, age threshold 40 (accuracy: 31.82%)

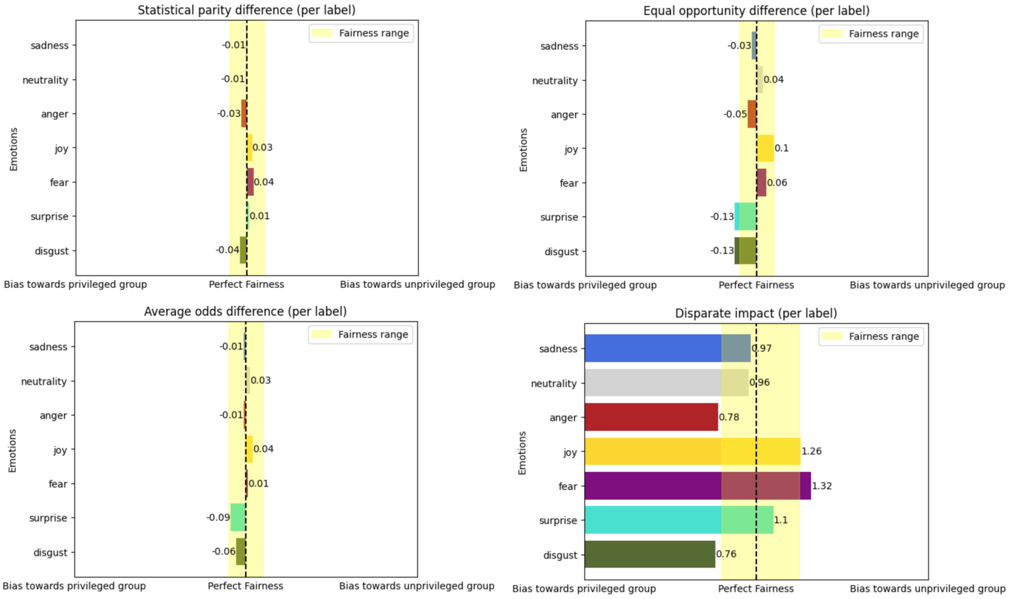

Figure C.25: Emozionalmente, ResNet, age threshold 27 (accuracy: 49.69%)

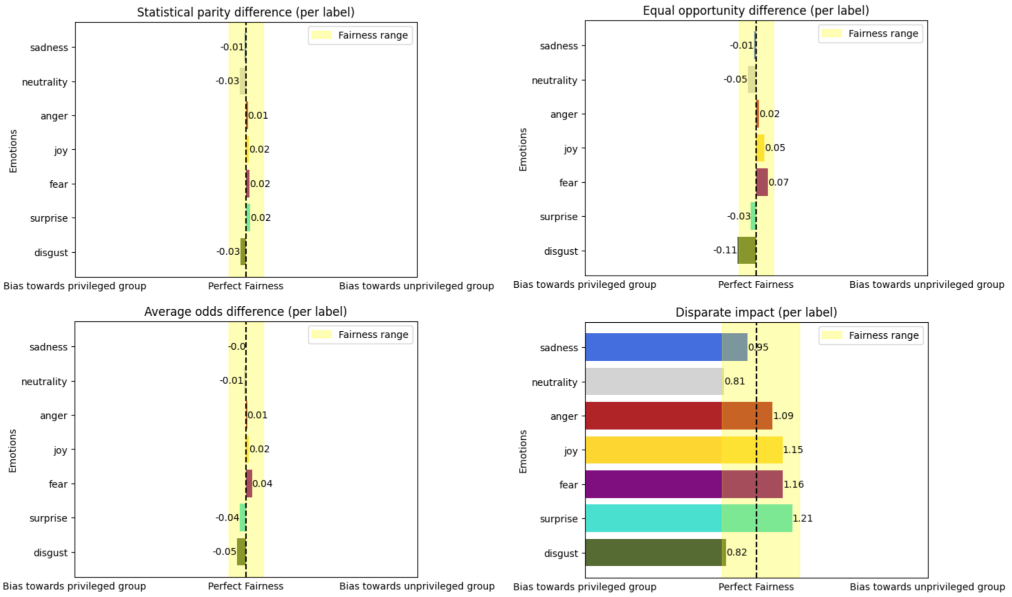

Figure C.26: Emozionalmente, ResNet, age threshold 30 (accuracy: 49.69%)

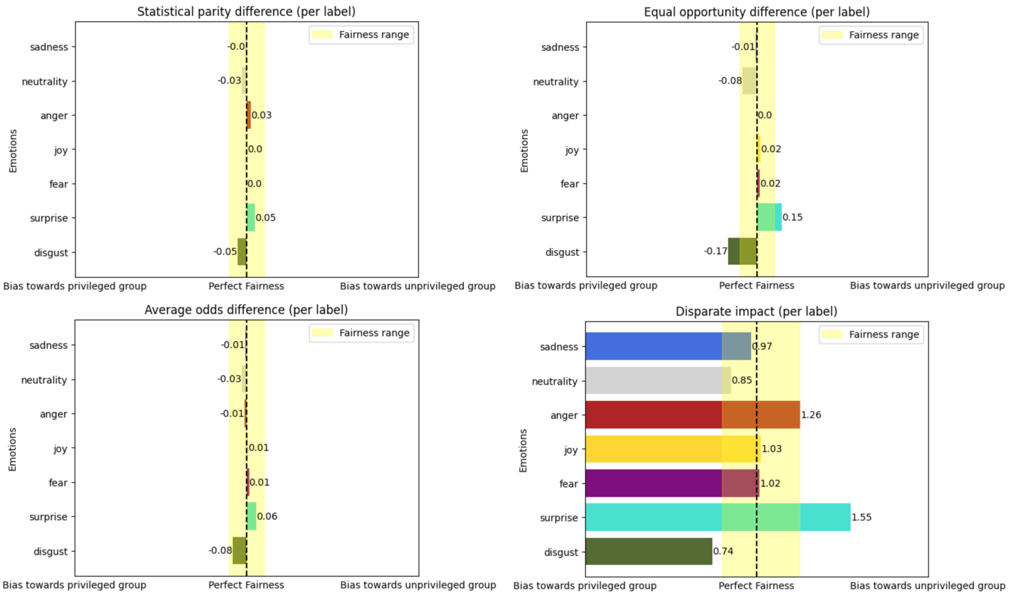

Figure C.27: Emozionalmente, ResNet, age threshold 40 (accuracy: 49.69%)

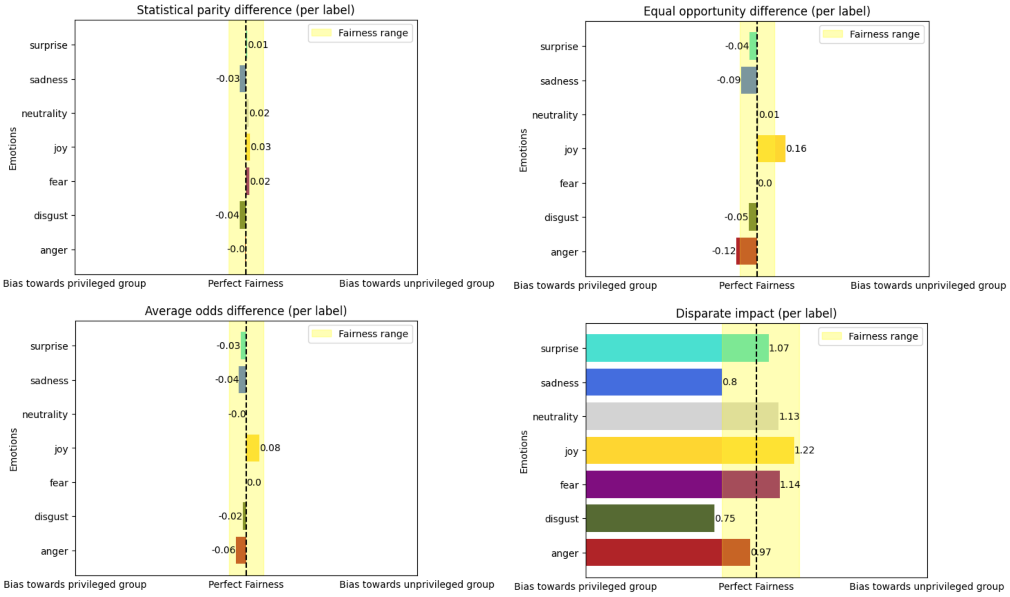

Figure C.28: Emozionalmente, WavLM, age threshold 27 (accuracy: 90%)

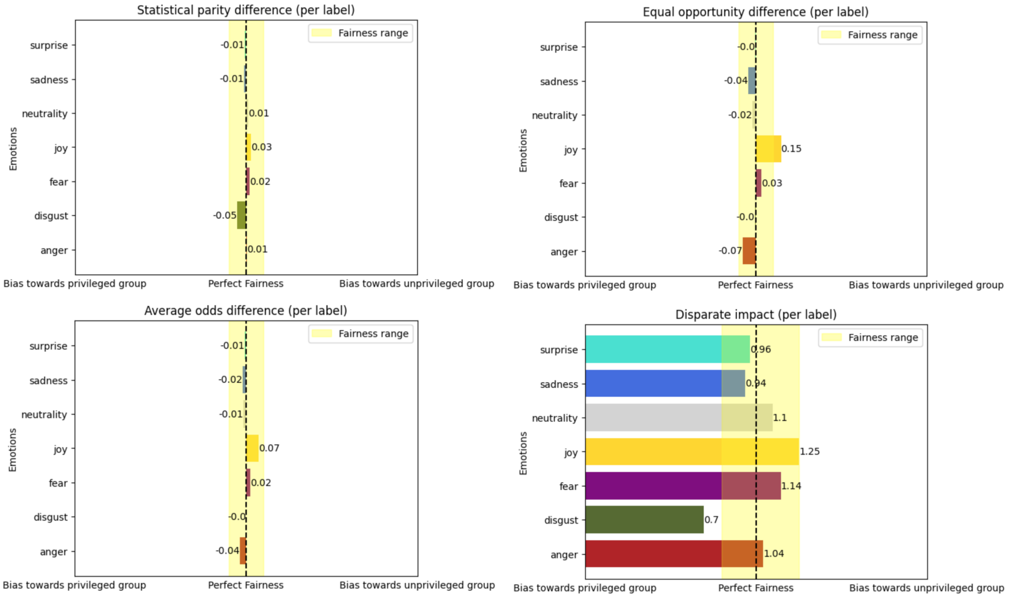

Figure C.29: Emozionalmente, WavLM, age threshold 30 (accuracy: 90%)

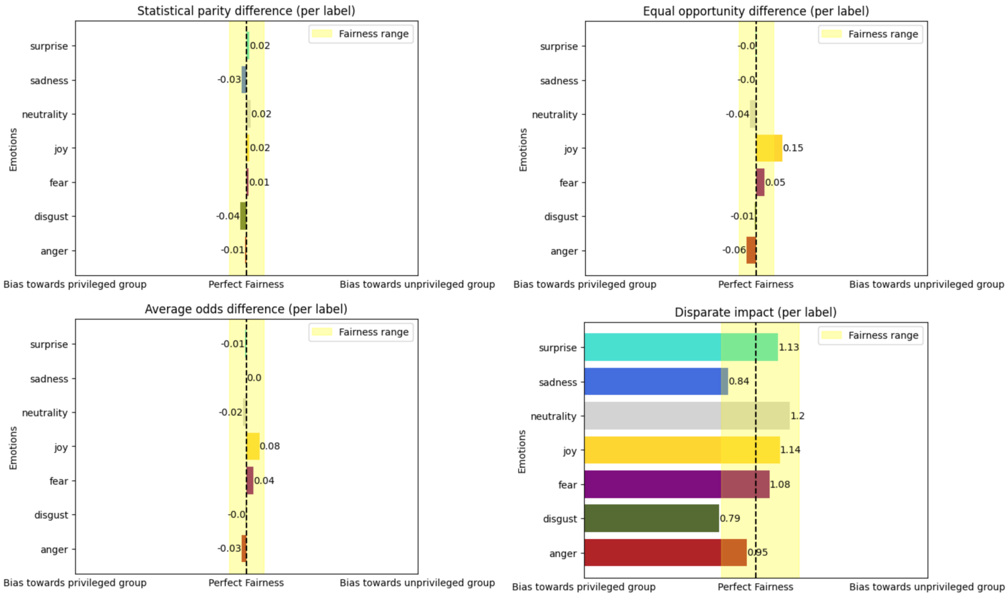

Figure C.30: Emozionalmente, WavLM, age threshold 40 (accuracy: 90%)

Bibliography

[1] Jessica L. Tracy and Daniel Randles. «Four Models of Basic Emotions: A Review of Ekman and Cordaro, Izard, Levenson, and Panksepp and Watt». In: *Emotion Review* 3.4 (2011), pp. 397–405. DOI: 10.1177/1754073911410747. eprint: https://doi.org/10.1177/1754073911410747. URL: https://doi.org/10.1177/1754073911410747 (cit. on pp. 1, 3).

[2] Laurence Vidrascu and Laurence Devillers. «Detection of real-life emotions in call centers». In: Sept. 2005, pp. 1841–1844. DOI: 10.21437/Interspeech.2005-582 (cit. on p. 2).

[3] Akputu K. Oryina and Abiodun O. Adedolapo. «Emotion Recognition for User Centred E-Learning». In: *2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*. Vol. 2. 2016, pp. 509–514. DOI: 10.1109/COMPSAC.2016.106 (cit. on p. 2).

[4] Xu Huahu, Gao Jue, and Yuan Jian. «Application of Speech Emotion Recognition in Intelligent Household Robot». In: *2010 International Conference on Artificial Intelligence and Computational Intelligence*. Vol. 1. 2010, pp. 537–541. DOI: 10.1109/AICI.2010.118 (cit. on p. 2).

[5] Imen Tayari, Nhan Thanh, and Chokri Ben Amar. «Multimodal Approach for Emotion Recognition Using a Formal Computational Model». In: *International Journal of Applied Evolutionary Computation* 4 (July 2013), pp. 11–25. DOI: 10.4018/jaec.2013070102 (cit. on p. 2).

[6] Javier de Lope and Manuel Graña. «An ongoing review of speech emotion recognition». In: *Neurocomputing* 528 (2023), pp. 1–11. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2023.01.002. URL: https://www.sciencedirect.com/science/article/pii/S0925231223000103 (cit. on p. 3).

[7] Rajesvary Rajoo and Ching Chee Aun. «Influences of languages in speech emotion recognition: A comparative study using Malay, English and Mandarin languages». In: *2016 IEEE Symposium on Computer Applications Industrial Electronics (ISCAIE)*. 2016, pp. 35–39. DOI: 10.1109/ISCAIE.2016.7575033 (cit. on pp. 3, 4).

[8] Robert Plutchik. «The Nature of Emotions: Human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice». In: *American Scientist* 89.4 (2001), pp. 344–350. ISSN: 00030996. URL: `http://www.jstor.org/stable/27857503` (visited on 02/21/2024) (cit. on p. 3).

[9] Paul Ekman. «Facial Expressions». In: *Handbook of Cognition and Emotion*. John Wiley Sons, Ltd, 1999. Chap. 16, pp. 301–320. ISBN: 9780470013496. DOI: `https://doi.org/10.1002/0470013494.ch16`. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/0470013494.ch16`. URL: `https://onlinelibrary.wiley.com/doi/abs/10.1002/0470013494.ch16` (cit. on p. 3).

[10] «Autism, Expression, and Understanding of Emotions: Literature Review». In: 16 (). DOI: `10.3991/ijoe.v16i02.11991`. URL: `https://online-journals.org/index.php/i-joe/article/view/11991` (cit. on p. 4).

[11] Mohammad Ariff Rashidan, Shahrul Na'im Sidek, Hazlina Md. Yusof, Madihah Khalid, Ahmad Aidil Arafat Dzulkarnain, Aimi Shazwani Ghazali, Sarah Afiqah Mohd Zabidi, and Faizanah Abdul Alim Sidique. «Technology-Assisted Emotion Recognition for Autism Spectrum Disorder (ASD) Children: A Systematic Literature Review». In: *IEEE Access* 9 (2021), pp. 33638–33653. DOI: `10.1109/ACCESS.2021.3060753` (cit. on p. 4).

[12] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. *A Survey on Bias and Fairness in Machine Learning*. 2022. arXiv: `1908.09635 [cs.LG]` (cit. on p. 5).

[13] Alexandra Chouldechova and Aaron Roth. *The Frontiers of Fairness in Machine Learning*. 2018. arXiv: `1810.08810 [cs.LG]` (cit. on p. 5).

[14] Dana Pessach and Erez Shmueli. *Algorithmic Fairness*. 2020. arXiv: `2001.09784 [cs.CY]` (cit. on pp. 5, 9).

[15] Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. *Inherent Trade-Offs in the Fair Determination of Risk Scores*. 2016. arXiv: `1609.05807 [cs.LG]` (cit. on p. 6).

[16] Aditya Krishna Menon and Robert C. Williamson. *The cost of fairness in classification*. 2017. arXiv: `1705.09055 [cs.LG]` (cit. on p. 6).

[17] Sorelle A. Friedler, Carlos Scheidegger, Suresh Venkatasubramanian, Sonam Choudhary, Evan P. Hamilton, and Derek Roth. *A comparative study of fairness-enhancing interventions in machine learning*. 2018. arXiv: `1802.04422 [stat.ML]` (cit. on p. 6).

[18] Mehmet Berkehan Akçay and Kaya Oğuz. «Speech emotion recognition: Emotional models, databases, features, preprocessing methods, supporting modalities, and classifiers». In: *Speech Communication* 116 (2020), pp. 56–76. ISSN: 0167-6393. DOI: https://doi.org/10.1016/j.specom.2019.12.001. URL: https://www.sciencedirect.com/science/article/pii/S0167639319302262 (cit. on p. 7).

[19] Moataz El Ayadi, Mohamed S. Kamel, and Fakhri Karray. «Survey on speech emotion recognition: Features, classification schemes, and databases». In: *Pattern Recognition* 44.3 (2011), pp. 572–587. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2010.09.020. URL: https://www.sciencedirect.com/science/article/pii/S0031320310004619 (cit. on p. 7).

[20] Yan Zhou, Heming Zhao, Xinyu Pan, and Li Shang. «Deception detecting from speech signal using relevance vector machine and non-linear dynamics features». In: *Neurocomputing* 151 (2015), pp. 1042–1052. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2014.04.083. URL: https://www.sciencedirect.com/science/article/pii/S0925231214013435 (cit. on p. 7).

[21] B. Yang and M. Lugger. «Emotion recognition from speech signals using new harmony features». In: *Signal Processing* 90.5 (2010). Special Section on Statistical Signal Array Processing, pp. 1415–1423. ISSN: 0165-1684. DOI: https://doi.org/10.1016/j.sigpro.2009.09.009. URL: https://www.sciencedirect.com/science/article/pii/S0165168409003843 (cit. on p. 7).

[22] Chien Shing Ooi, Kah Phooi Seng, Li-Minn Ang, and Li Wern Chew. «A new approach of audio emotion recognition». In: *Expert Systems with Applications* 41.13 (2014), pp. 5858–5869. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2014.03.026. URL: https://www.sciencedirect.com/science/article/pii/S0957417414001638 (cit. on p. 7).

[23] Tin Lay Nwe, Say Wei Foo, and Liyanage C De Silva. «Speech emotion recognition using hidden Markov models». In: *Speech Communication* 41.4 (2003), pp. 603–623. ISSN: 0167-6393. DOI: https://doi.org/10.1016/S0167-6393(03)00099-2. URL: https://www.sciencedirect.com/science/article/pii/S0167639303000992 (cit. on p. 7).

[24] J. Ancilin and A. Milton. «Improved speech emotion recognition with Mel frequency magnitude coefficient». In: *Applied Acoustics* 179 (2021), p. 108046. ISSN: 0003-682X. DOI: https://doi.org/10.1016/j.apacoust.2021.108046. URL: https://www.sciencedirect.com/science/article/pii/S0003682X21001390 (cit. on pp. 7, 19).

[25] Cem Doğdu, Thomas Kessler, Dana Schneider, Maha Shadaydeh, and Stefan R. Schweinberger. «A Comparison of Machine Learning Algorithms and Feature Sets for Automatic Vocal Emotion Recognition in Speech». In: *Sensors* 22.19 (2022). ISSN: 1424-8220. DOI: `10.3390/s22197561`. URL: `https://www.mdpi.com/1424-8220/22/19/7561` (cit. on p. 7).

[26] Martin Vondra and Robert Vích. «Recognition of Emotions in German Speech Using Gaussian Mixture Models». In: *Multimodal Signals: Cognitive and Algorithmic Issues*. Ed. by Anna Esposito, Amir Hussain, Maria Marinaro, and Raffaele Martone. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 256–263. ISBN: 978-3-642-00525-1 (cit. on p. 7).

[27] J. Umamaheswari and A. Akila. «An Enhanced Human Speech Emotion Recognition Using Hybrid of PRNN and KNN». In: *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*. 2019, pp. 177–183. DOI: `10.1109/COMITCon.2019.8862221` (cit. on p. 8).

[28] Babak Joze Abbaschian, Daniel Sierra-Sosa, and Adel Elmaghraby. «Deep Learning Techniques for Speech Emotion Recognition, from Databases to Models». In: *Sensors* 21.4 (2021). ISSN: 1424-8220. DOI: `10.3390/s21041249`. URL: `https://www.mdpi.com/1424-8220/21/4/1249` (cit. on p. 8).

[29] Ruhul Amin Khalil, Edward Jones, Mohammad Inayatullah Babar, Tariqullah Jan, Mohammad Haseeb Zafar, and Thamer Alhussain. «Speech Emotion Recognition Using Deep Learning Techniques: A Review». In: *IEEE Access* 7 (2019), pp. 117327–117345. DOI: `10.1109/ACCESS.2019.2936124` (cit. on p. 8).

[30] Giovanni Costantini, Iacopo Iaderola, Andrea Paoloni, and Massimiliano Todisco. «EMOVO Corpus: an Italian Emotional Speech Database». In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. Ed. by Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Reykjavik, Iceland: European Language Resources Association (ELRA), May 2014, pp. 3501–3504. URL: `http://www.lrec-conf.org/proceedings/lrec2014/pdf/591_Paper.pdf` (cit. on pp. 8, 27).

[31] Fabio Catania. «Speech Emotion Recognition in Italian Using Wav2Vec 2.0 and the Novel Crowdsourced Emotional Speech Corpus Emozionalmente». In: (May 2023). DOI: `10.36227/techrxiv.22821992.v1`. URL: `http://dx.doi.org/10.36227/techrxiv.22821992.v1` (cit. on pp. 8, 28).

[32] Youddha Singh and Shivani Goel. «A lightweight 2D CNN based approach for speaker-independent emotion recognition from speech with new Indian Emotional Speech Corpora». In: *Multimedia Tools and Applications* 82 (Feb. 2023), pp. 1–19. DOI: `10.1007/s11042-023-14577-w` (cit. on p. 8).

[33] Baraa Zayene, Chiraz Jlassi, and Najet Arous. «3D Convolutional Recurrent Global Neural Network for Speech Emotion Recognition». In: *2020 5th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*. 2020, pp. 1–5. DOI: `10.1109/ATSIP49331.2020.9231597` (cit. on p. 8).

[34] Youddha Singh and Shivani Goel. «An efficient algorithm for recognition of emotions from speaker and language independent speech using deep learning». In: *Multimedia Tools and Applications* 80 (Apr. 2021). DOI: `10.1007/s11042-020-10399-2` (cit. on p. 8).

[35] Ben Hutchinson and Margaret Mitchell. «50 Years of Test (Un)fairness: Lessons for Machine Learning». In: *Proceedings of the Conference on Fairness, Accountability, and Transparency*. FAT* '19. ACM, Jan. 2019. DOI: `10.1145/3287560.3287600`. URL: `http://dx.doi.org/10.1145/3287560.3287600` (cit. on p. 8).

[36] Sahil Verma and Julia Rubin. «Fairness definitions explained». In: *Proceedings of the International Workshop on Software Fairness*. FairWare '18. Gothenburg, Sweden: Association for Computing Machinery, 2018, pp. 1–7. ISBN: 9781450357463. DOI: `10.1145/3194770.3194776`. URL: `https://doi.org/10.1145/3194770.3194776` (cit. on p. 9).

[37] Pedro Saleiro, Benedict Kuester, Loren Hinkson, Jesse London, Abby Stevens, Ari Anisfeld, Kit T. Rodolfa, and Rayid Ghani. *Aequitas: A Bias and Fairness Audit Toolkit*. 2019. arXiv: `1811.05577 [cs.LG]` (cit. on p. 10).

[38] Rachel K. E. Bellamy et al. *AI Fairness 360: An Extensible Toolkit for Detecting, Understanding, and Mitigating Unwanted Algorithmic Bias*. 2018. arXiv: `1810.01943 [cs.AI]` (cit. on p. 10).

[39] Paul Ekman. *Atlas of Emotions*. https://atlasofemotions.org/ (cit. on p. 15).

[40] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. *Deep Residual Learning for Image Recognition*. 2015. arXiv: `1512.03385 [cs.CV]` (cit. on p. 22).

[41] Sanyuan Chen et al. «WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing». In: *IEEE Journal of Selected Topics in Signal Processing* 16.6 (Oct. 2022), pp. 1505–1518. ISSN: 1941-0484. DOI: `10.1109/jstsp.2022.3188113`. URL: `http://dx.doi.org/10.1109/JSTSP.2022.3188113` (cit. on p. 24).