POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Speech Fingerprinting and Matching for Content Retrieval

Master Thesis in
Music Engineering

Author: **Laura Colzani**

Student ID: 953563
Advisor: Prof. Paolo Bestagini
Co-advisors: Luca Cuccovillo, Artem Yaroshchuk
Academic Year: 2020-21

To my grandmother,
whose memory has always spurred me on to live and love
to the fullest extent of my potential.

Abstract

Audio fingerprinting and matching is a technology which is widely used in many contexts. From research to commercial applications, from forensics to all the situations in which sound content retrieval is vital, its benefits are acknowledged. The robustness of matching based on audio fingerprinting is so high that it can be effective in very noisy environments as well. Nearly everything has been explored if we talk about music field. What could be possible if we manage a way to implement audio fingerprinting for speech? As for music, also matching speech might be relevant in many scenarios: from countering disinformation by spotting mashups, to monitoring viral content, and many others.

With this premise, the goal of this work is to study, and eventually provide, a method for speech matching and recognition using a fingerprint. The technologies studied and proposed aim to be effective in different contexts, such as noisy environments or situations where background music can be heard as well. After a proper explanation of the state of the art, the proposed methods will be shown: two initial techniques, inspired directly by the state of the art and mainly based on digital signal processing, and two additional techniques studying how recent neural network solutions could help developing efficient techniques for speech fingerprint extractions.

Keywords: Speech fingerprinting, audio matching, deep neural networks

Abstract in lingua italiana

La tecnologia di audio fingerprinting e matching è ampiamente utilizzata in molti contesti. I suoi vantaggi sono riconosciuti: dalla ricerca alle applicazioni commerciali, dall'ambito forense a, più in generale, tutte le situazioni in cui il recupero di contenuti sonori è vitale. La robustezza di un audio matching condotto sfruttando delle fingerprint è così elevata che può essere efficace anche in ambienti molto rumorosi. In ambito musicale, quasi ogni ipotesi è stata esplorata. Ci si chiede dunque: cosa potrebbe essere possibile se riuscissimo a trovare una metodologia per implementare l'audio fingerprinting per il parlato? Come per la musica, anche uno speech matching potrebbe essere determinante in diversi scenari: dal contrastare la disinformazione individuando i mashup, al monitoraggio dei contenuti virali, e altro ancora.

Con questa premessa, l'obiettivo di questo lavoro è quello di studiare, ed ad ultimo fornire, un metodo per il matching e il riconoscimento del parlato utilizzando una fingerprint. Le tecnologie studiate e proposte mirano ad essere efficaci in diversi contesti, come ambienti rumorosi o situazioni in cui si può sentire anche della musica di sottofondo. Dopo un'adeguata spiegazione dello stato dell'arte, proporremo diversi metodi per approcciare il problema: due ispirati direttamente dallo stato dell'arte e basati principalmente sull'elaborazione digitale del segnale, e ulteriori due atti a studiare come i recenti progressi in ambito di deep learning possano aiutare a sviluppare tecniche efficienti per l'estrazione della fingerprint.

Parole chiave: Speech fingerprinting, audio matching, deep neural networks

Contents

Introduction

Audio matching is a vital topic for many applications. Its studies are widely exploited by applications in everyday life, and here the thought goes to Shazam, and by more professional environments [1].

In particular, audio forensics is a very sensitive topic in which any kind of content retrieval is crucial [2][3]: is the audio file corrupted? how is it corrupted? is the actual person speaking? can a specific source, i.e., company or website, broadcast a specific content? how can someone be sure the files are shared sticking to copyrights law? In this context indeed, the audio and, in general, multimedia field is increasingly predominant: in an environment where the vast majority of content (audio, images, video) is entrusted to the web, the most crucial challenges concern precisely the research and study of techniques that can provide the most truthful data and considerations on multimedia content.

In fact, many tasks can benefit from multimedia technologies, such as the reconstruction of events starting from audiovisual testimonies, or the search for possible violations of privacy and copyright on multimedia contents, and even the assessment of the integrity of the evidence, through the collection of information such as, for example, the recording device or the perceptual or acoustic audio contents. We have to remember that all these elements contribute to be able to determine, in the final analysis, whether the evidence has been corrupted.

More specifically, there are several factors that contribute to the corruption of an audio file: it may have been modified, by adding noise or correcting the pitch, or by applying non-linear transformations that make the reconstruction of the original more difficult; or it may be the result of a cut-and-paste operation of several audio excerpts, in order to obtain false evidence so as to attempt to modify the sentence or to divert investigations; not to mention the possibility that, in order to give false testimony, one may speak of actual imitations.

Such technologies, from the forensic field, can also be either used in other collateral contexts, such as fake news detection (which could be useful in social networks environment, where every kind of information can be found), or employed in everyday life: again, just

think of the case of Shazam, which allows to identify the title and author of a piece of music in a few seconds of listening to an unknown song, even in noisy conditions.

Audio fingerprinting is a very effective method used for recognition, identification and monitoring tasks. Its effectiveness lies in its properties: a fingerprint is compact and contains very relevant information about the file it describes; hence, its robustness also becomes a requirement that makes it so popular and appreciated [4]. Plus, in a small amount of time, all the significant information about a specific content is detected, with little chance of committing mistakes.

Certainly, however, one has to pay attention to some fundamental steps in the process of fingerprint extraction: which transformation to adopt in order to best represent the content in time and frequencies, how to select the most representative and peculiar features, how to condense everything into a compact and computationally manageable object, and how to quickly compare a new fingerprint to an already known database in order to draw conclusions as close to reality as possible.

From the study of the state of the art one can analyse different approaches to effectively solve such steps [5][6][7], but this is mainly when audio fingerprints are extracted from musical audio. Whether artist tracks or backgrounds, the literature offers interesting and valid insights in these contexts [8][9].

Our interest, however, lies on speech fingerprints: is there any possibility of devising extraction methods that can also be effective for speech itself? The main difficulties arise from the fact that speech present characteristics, in time and frequency, that are not enough to perform a proper discrimination and recognition with the state-of-the-art music approaches [10], not to mention the fact that music has characteristics which are missing in speech, such as rhythm, tonality, wider frequency range.

There are, however, some publications that show that, rather than relying on hand crafted feature extraction methods, the use of neural networks can be considered [11][12]: although there is not a consistent number of papers related to audio content matching and fingerprinting, there are nevertheless interesting results related to speech processing, especially in the field of speaker recognition, speaker diarization, and speech enhancement [13]. These same results can become sources of inspiration for possible work in this area.

With this in mind, the thesis consists of a project that was developed in two directions: firstly, it was intended to show that approaches for the musical context are not satisfactory; secondly, an approach involving the use of Deep Neural Networks (DNNs) was proposed, taking inspiration from the state-of-the-art attempts [14] and combining the basic concepts

behind neural networks themselves.

In the first approach we consider one of the most robust examples of audio fingerprints, proposed by Haitsma and Kalker in "A highly robust audio fingerprinting system" [7].

First, the fingerprint extraction phase is implemented, involving the transformation of the audio in order to observe its content in time and frequencies, the transformation phase to emphasise the most perceptually meaningful areas of the audio file, and the actual fingerprint processing (in this specific case, observing the energy values of appropriate sections of the previously transformed audio file). Then, the matching algorithm is implemented, between the fingerprints that constitute a query, about which hypothetically nothing is known, and the database of known fingerprints.

In the second approach, instead, we tried to incorporate deep learning in our work, in order to formalise a more effective algorithm of fingerprint extraction.

First of all a DNN is built, which should extract fingerprints on its own by learning how to model patterns inherent in the audio files. These same audio files are fed to the neural network as if they were images and, therefore, with information content, once again on both time and frequency. It is then trained according to specific loss definition, which in our case is Triplet Loss, very effective and popular in terms of image, especially face recognition [15][16]. The matching algorithm, on the other hand, stays nearly the same, allowing a more effective comparison between the two approaches.

For both cases, a dataset is used with original audio files, in English and Spanish and regarding different speakers, to which noise is applied. A distinction is made between:

- actual noises, such as background chatter or sounds produced by tools, e.g. electrical appliances, or other sounds such as running water;
- classical music in the background, in order to understand how effective the versions of the project are at focusing only on speech, neglecting the music as much as possible.

They were applied to the audio files according to different Signal-to-Noise Ratios (SNRs), ranging from a minimum of -10 dB to a maximum of 5 dB.

Moreover, in both cases we experimented the additional usage of Spleeter, a well known software for source separation in music, both for splitting vocals from the whole accompaniment and for splitting the most common instruments involved in contemporary music [17].

In order to evaluate the two versions of the project, the focus was on the accuracy met-

ric and the Equal Error Rate (EER) metric: the latter being decisive in calculating the decision threshold for a successful match between query and database. While the implementation of Haitsma and Kalker's proposal gives unsatisfactory results in terms of accuracy, especially if files have been affected by background music, the proposal involving the use of a neural network shows satisfactory results, with accuracy showing really good improvements, reaching percentages mostly around 90% especially with favourable SNRs.

This should be a good starting point in the development of new fingerprint extraction techniques: deep learning knowledge demonstrates to be a powerful tool for having precise and concise representation of audio files, with the hope that some other researches could take this direction and develop it way further. The proposed technology, in fact, could either replace or reinforce the most classical (and manual) techniques.

The thesis is articulated as follows:

- in Chapter 1, the state of the art is described, in order to achieve a better understanding of the concept related to fingerprints, how they are extracted and all the possible technologies in which the extraction itself can be performed; some more classic pipelines regarding handcrafted fingerprints will be shown, and some interesting papers on the use of neural networks will be reported as well; also the differences between music and speech in terms of features will be briefly pointed out;
- in Chapter 2, the reasoning that brought from an existing algorithm to the choice of exploring new approaches to the problem is showed: a state-of-the-art method will be clearly illustrated, and also a recap on the different possibilities of deep learning and their possible use will be exposed;
- in Chapter 3, starting from the formalization of the general workflow, the pipelines of our proposed approaches are explained;
- in Chapter 4, the experiments are described: from the construction of dataset, whose pipeline is included as well to have a clearer idea of the settings, to the brief description and reason to choose the metrics involved, and then concluding to the results, which are shown, commented and compared;
- in Chapter 5 we end the thesis with some brief final considerations and conclusions.

1 | State of the Art

In this chapter, the state of the art is explained. After a brief introduction with a more formal definition of audio fingerprinting and matching, we will report some interesting state-of-the-art proposals about both topics, concluding with our considerations toward the analysis of the state of the art itself.

1.1. Audio Fingerprinting

An audio fingerprint is defined as a compact content-based signature, which is able to summarise an audio recording in a meaningful way [4].

Audio fingerprinting therefore is a technology allowing the extraction of some relevant acoustic characteristics from a piece of audio content, which will eventually be processed and stored in a database. The goal is, whenever unidentified audio media is presented, to determine if it matches with the data already present in a database. Features of the input audio media are extracted, processed to obtain a fingerprint, and matched with fingerprints that already exist in the database. Also distorted versions of the same recording can be identified, if the design of fingerprints is effective enough, and that is a powerful characteristic when considering contexts of audio-matching with tricky scenarios, possibly involving every-day life.

Audio fingerprinting and matching differ from other audio content identification approaches, since the message is automatically deducted from the most significant perceptual components of sound, and is therefore a distinct entity from the actual signal. Other existing technologies embed some hidden messages within the audio and therefore require a completely different setup for being retrieved.

1.1.1. Properties

An audio fingerprint must satisfy many requirements [4]. Some of them can be prioritized in spite of other ones, according to the context and according to the goal of the specific task involving this technology.

Generally, the properties defining an audio fingerprint are:

- *Accuracy* The ability of reaching a satisfying percentage of correct identifications over the total number of identifications;
- *Reliability* The chance to assess, without little to no doubt, whether a query is present or not in the database. This leads to the avoidance of false positives. It is very important in terms of copyright enforcement;
- *Robustness* The ability of identifying an item regardless of compressions, distortions, interferences affecting that same item or the transmission channel. Other parameters to test robustness are pitching, equalization, background noise, conversions, lossy audio coding, and so on. It is a property working in synergy with the accuracy one;
- *Granularity* The ability to identify a known element of the database from very short excerpts of audio content, usually a few seconds long. This leads to having some more complexity to the research, since a comparison of audio in all possible alignments is needed;
- *Security* The possibility to contrast cracking or tampering solutions, which are designed to fool the fingerprint identification algorithms, especially for ensuring obligations of confidentiality;
- *Versatility* The ability to identify audio content regardless of the audio format, and to use the same database for different applications and scenarios;
- *Scalability* The possibility to perform with very large databases of titles or a large number of concurrent identifications, despite this could affect the properties of accuracy and complexity of the system;
- *Complexity* The computational cost of some steps in the process, such as fingerprint extraction, its size, complexity of the search algorithm, the complexity of the fingerprint comparison, the costs of adding new items to the database, and so on;
- *Fragility* The need, for some applications, to detect changes in the content, which clashes with the property of robustness. This should lead to robustness to content-preserving transformations only, excluding other distortions.

As in many engineering situations, it has to be kept in mind that not all properties allow the same gain in performance while improving a certain requirement. Whenever there is the need to modify the pipeline, for improving one property, it has to be remembered there will be a loss of the performance in some other.

In order to raise awareness about how to deal with the properties for an effective implementation of the technology, it is useful to stick to some guidelines [4], and keep in mind that a fingerprint should be

- A perceptual digest of the recording. It must retain as much relevant acoustic information as possible, allowing discrimination over a large number of fingerprints
- Invariant to distortions, due to the robustness property. As declared before, this constrain is relaxed only for some specific applications, generally content-integrity ones. Generally, audio content has to be recognized despite disturbances or the most common manipulations
- Compact. Since a very large number of fingerprints needs to be stored and compared, a shorter representation gives benefit in terms of complexity. However, this should not lead to loss in accuracy, reliability, or robustness
- Easily computable. If the extraction of a fingerprint is particularly time-consuming, complexity will affect the performances, especially when there is the need to build a significant database, and when the extraction should take place on embedded or portable devices with limited energy capacity.

1.1.2. Overview on General Algorithm

The general pipeline regarding fingerprint extraction and consequent audio matching is the one in Figure 1.1:

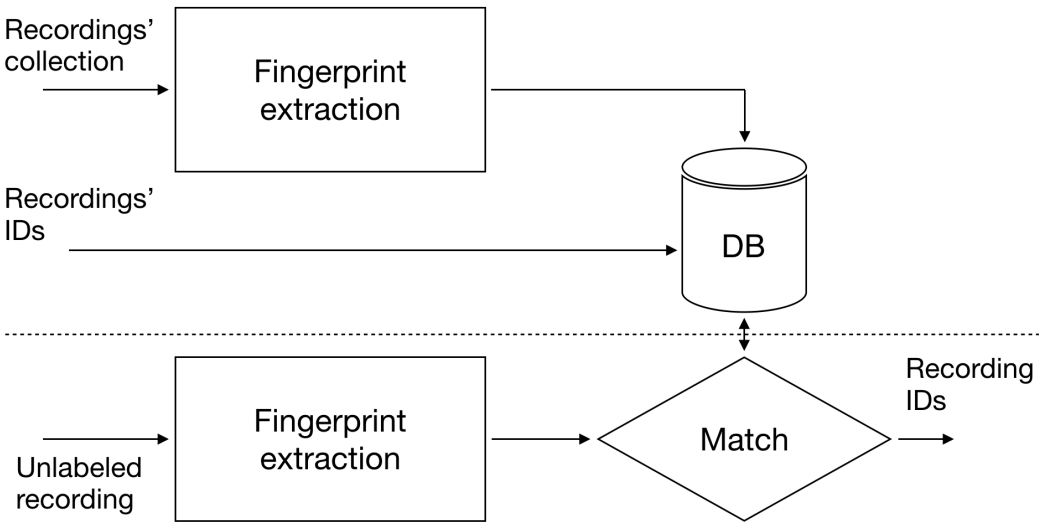


Figure 1.1: The whole pipeline of audio fingerprinting as described in [4]

Every step is now described very briefly:

- a collection of recordings is defined; each and everyone is labelled with a specific ID
- a set of unlabeled recordings is proposed as well
- for every recording of both the audio sets, a fingerprint is extracted: the ones belonging to the labelled set will become part of the actual DataBase;
- a rule for matching, or a set of rules, is proposed: each fingerprint can be interpreted as a query among a set of queries for the database, and if everything works correctly, the correct label, or recording ID in general, is assigned to the unknown recording.

It is necessary to spend some words more about the two most important steps of the pipeline, i.e., fingerprint extraction and matching against the database.

1.1.3. Fingerprint Extraction

The complete set of actions involved in fingerprint extraction is depicted in Figure 1.2. Two phases can be acknowledged: the front-end phase and the modeling phase

Front-End Phase

The front-end phase is a bit more complex, since the goal is to take audio files and then feed the fingerprint modeling block with a proper set of features. In Figure 1.2 the steps that compose the front-end phase itself are shown. We are going to detail each block mentioned.

Preprocessing: audio files are subjected to some necessary preliminary actions. For example, they are converted from analog to digital format, following a specific coding protocol, so that some parameters have to be defined (e.g., the sampling rate). In other scenarios, other approaches are used, for example pre-emphasis techniques, or also normalization techniques, in order to have control on the limits of the audio dynamic range.

Framing and Overlapping: starting from the assumption that, in intervals of some milliseconds, the signal can be considered as quasi-stationary, the signal is then divided through a framing process. Frame duration is chosen according to that same hypothesis of stationarity, and for all the frames the same window function is chosen, being careful that it will minimize the discontinuities from frame to frame.

The process of overlapping assures more robustness during the windowing process, so the more overlap, the more likely that nearly all the audio data will be kept. However, is has

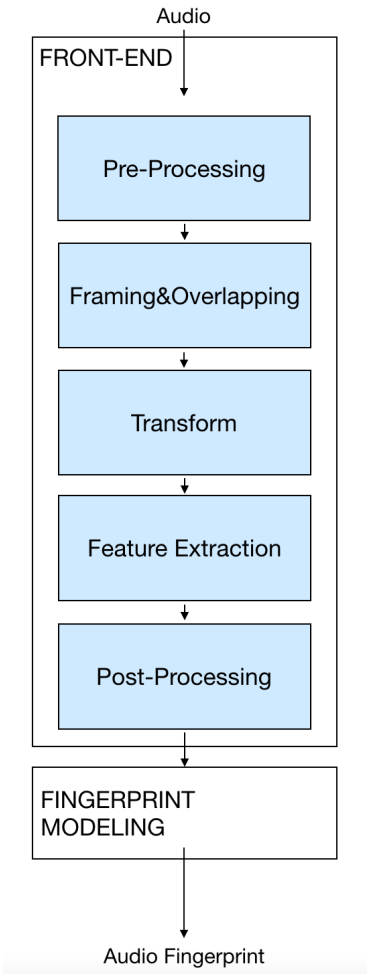


Figure 1.2: The whole pipeline of fingerprints extraction [4]

to be taken in account that this process takes time, since more frames will be loaded. A trade off between the property of robustness and the property of low computational cost always has to be considered

Transform: this sub-block has the goal to project audio information in a more significant space, so that redundancy is reduced. There is a very wide choice among all transformations, and the most suitable one depends on the scenario. For example, if Karhunen-Loève Transform (KLT) or Singular Value Decomposition (SVD) are chosen [18], there will be a gain in obtaining uncorrelated samples, but a loss in terms of computational complexity (and cost). Some other transforms do not have this issue, since they use fixed basis vectors, and some others involve a change from time to frequency domain, in order to have a gain in efficiency. Also power measures have been using effectively, remembering that it can be seen as a time-frequency distribution at the minimum term [9][19].

Among all the possibilities, the Discrete Fourer Transform (DFT), or from a computa-

tional point of view the Fast Fourier Transform (FFT), is still the most common: there have been other proposals, such as the Discrete Cosine Transform (DCT), the Haar Transform (HT) or the Walsh-Hadamard Transform (WHT) [20], but it has been shown [21] how DFT remains the most stable to shifting.

Feature Extraction: the process in which the final acoustic vectors are generated. Usually, perceptually meaningful parameters are taken into account, and this philosophy is applied in many fingerprints proposal: from Mel Frequency Cepstral Coefficientss (MFCCs), to Spectral Flatness Measure (SFM), usually computed for a specific sub-band in the spectrum, going on with some more articulated algorithms. Among the most significant ones, we would like to highlight [4]:

- the band representative vectors [22], presented by Papaodysseus et al., which are a list of indexes of bands with prominent tones, in the sense of peaks with significant amplitude;
- the hash string, proposed by Haitsma and Kalker in [7], defined as the sign of the energy band differences, starting from the signal divided in 33 bark-scaled bands;
- the frequency modulation analysis [23], declined in the geometric mean of he modulation frequency estimation of the energy of 19 bark-spaced band-filters;
- approaches involving a re-arrangement of features, like the one of Burges et al. proposed in [24], involving a modified Principal Component Analysis (PCA) called Oriented PCA, focused on non-orthogonal directions, taking some predefined distortions into account.

Post-Processing: since the feature extraction process usually brings to absolute measurements, not taking into account some more descriptive and robust characteristics of the audio file, an action or a sequence of actions can be taken into account; it is not so uncommon, for example, to consider the derivative or the second derivative alongside the extracted features [8][25][6][19]. This usually helps in gaining more robustness against linear time invariants, or slowly varying channel distortions, such as the ones in the equalization process.

Speaking of robustness, also quantization is taken into account: a very low resolution one [19][21][7] can also help in terms of complexity, when there is the need of storing a huge amount of fingerprints.

Quantization makes it possible to minimise the collision probability of two fingerprints in the matching phase [5].

Modelling Phase

Having completed the front-end process, and thus having obtained features from the audio files, in this phase the aim is to try to further reduce the size of the fingerprints; this process is the easier the more redundancies are observed, both in the audio itself and considering the database as a whole. It should not be forgotten that this phase is crucial since the searching and matching algorithms will be conditioned by the techniques adopted in this same phase.

Amongst the various examples, a fairly common one is to manipulate the vector sequences coming out of the front-end to condense them into a single vector, which can describe the whole file or at least a fragment of it. This approach is applied, for example:

- in the Etantrum proposal [26], which calculates this vector by exploiting the mean and variance of the energy of the audio, after it has been filtered with a bank of sixteen filters: the result is a fingerprint of 512 bits, which therefore satisfies the criterion of compactness;
- in the TRM Recognises Music (TRM) signature of the Musicbrainz [27] company, which includes several features among which the average zero crossing rate, the estimated beats per minute, the average spectrum, and produces a fingerprint which is indeed compact, and which allows for good computational efficiency.

However, there is nothing to prevent fingerprints from being defined as sequences, traces, or trajectories of features. In order to have a greater gain in terms of memory efficiency, it is not so unusual for such representations to be binary, or at least binary encoded, as happens in the proposals of Haitsma and Kalker [7] or Papaodysseus et al. [22].

The post-processing phase can be considered a continuation of the fingerprint extraction process: in [24], the application of the oriented PCA already allows a progressive reduction of statistical redundancy.

Other approaches exploit instead what could be defined as Global Redundancy, but they are effective and exploited especially in musical contents [6][8][19]: it is in fact not so impossible to find similar points in the same song, for instance because of a refrain.

1.1.4. Fingerprint Matching

Fingerprint Matching includes all processes required to ensure that, given an input query fingerprint and a database storing some fingerprints, there is the ability to determine whether the query itself is part of the database or not, and to determine which is the

matching fingerprint in the case the query belongs to the database. This task, of course, has to be performed so that the decision of the system is as close as possible to truth.

Two elements, in particular, are of high importance in the definition and effectiveness of a matching algorithm: similarity measure calculation, and searching algorithm.

Similarity Measure

Depending on the fingerprint extraction techniques, the techniques of similarity measures will also change.

If, for example, vector sequences have been extracted, the most efficient way to compare them will be some correlation metric, and for this task the Euclidean Distance (L2), or variants that take into account the possible difference in length of the sequences, is applied.

This can take the form of fairly common and efficient forms of classification: Sukittanon and Atlas propose a classification in [23], using a nearest neighbour algorithm, with a cross entropy estimation. In cases where the vectors have been quantized in post-processing, the Manhattan distance or the Hamming distance are more suitable [7].

There also have been proposals enhancing similarity with more than a linear relationship: the approach in [5] provides a metric called Exponential Pseudo Norm (EPN), which aims to emphasise more effectively both cases of similarity and cases of dissimilarity.

However, it must be remembered that sometimes, in the whole fingerprinting process, either we start from audio files from which the whole pipeline evolves, both for the database and for the query, or the database is already made up of fingerprints, and the whole process is undergone only by the query. In the latter case, especially in the proposals with codebooks [6][25], the vector sequences are associated to several of them, and for each one the error is computed and the codebook itself associated to the lowest error is selected.

Searching Methods

After having discussed the possible methods of matching, it is important to spend a few words on the searching methods. It is evidently impossible to employ a brute force algorithm, considering both the large quantity of fingerprints that could be stored in a database and the fact that the query is assumed to be unknown. If this method were to be adopted, the time needed to find the most suitable match would be proportional to the total number of fingerprints in the database. Therefore, the most common and effective search methods will be mentioned:

offline calculation of distances: this approach takes into account the assumption that, for possible queries, a match may already exist in the database. Therefore, in addition to having fingerprints, one will also keep track of the distance that exists between them, and the data structure will thus allow, once one is searching for a match online, to immediately discard the most striking non-matches. Without forgetting that, if the similarity measure stems from a calculation, or an algorithm, which has very specific properties, this approach allows the avoidance of false dismissals [28][29];

cheap similarity measure: this method serves purely for an initial screening, since the aim is to eliminate dissimilar fingerprints with the lowest possible computational cost. In order to avoid false dismissals, one must however remember that this measure must lower bound the more precise (and thus, expensive) similarity [29];

inverted file indexing: a method, proposed by Haitsma and Kalker in [7], involving the presence of an index of possible pieces of a fingerprint, pointing to positions of the audio files. So, on the assumption that there is exact match, a list of possible files and positions can be efficiently retrieved to search in an exhaustive fashion. Kurth et al. also present in [19] an index using code words instead of positions of audio files, and these code words are extracted from binary sequences representing the audio itself. This technique, due to the fact that takes into account that little to no errors are permitted in the words used to construct the index, could lead to false dismissals.

candidate pruning: the decision of keeping the best similarity score appearing during the search for a match, assuming it is nonsense to keep on with the search itself, if it is known that there is no room for improvement in the similarity scores anymore [28]. It is quite common [30] to exploit this method with tree structures, in order to avoid possible redundancies.

other approaches: theoretically, there are many other searching methods for fastening the matching process, also some more practical approaches. For example, the one proposed by Wang and Smith in [31], consisting in a smaller database with some fingerprints most likely to give a match to the query and in a bigger database, in which the search will go on if no match is found (with other searching methods, for example the inverted file indexing).

In the field of music, as can be seen from the various proposals mentioned in the discussion of a generic fingerprint extraction and audio matching pipeline, there have been several attempts with good results: one can therefore say with some confidence that we have a consolidated technology that can adequately fulfil its purpose.

But, as already mentioned, one wonders if there is a possibility to reach this level also for speech. In the next section, the state of the art of speech fingerprints will be addressed.

1.2. Speech Fingerprinting

In this section, we will talk about some significant attempts that have been made in speech fingerprinting: at first, a couple of examples with manual techniques will be presented; then, proposals involving both manual and deep neural network approaches will be shown (mixed approaches); and in the end, the attempts involving basically only deep learning methods. In the end of this section, we will also discuss the hypothesis on why it is more difficult to develop an effective algorithm in this context.

1.2.1. Manual Techniques

In this section we will illustrate some examples of fingerprint construction, starting from different methods of manual feature extraction.

The work we are going to present here is a two-phased experiment conducted by [32]. First of all, they create a binary fingerprint that is very much inspired by a well established method for music fingerprints, proposed in [7]: this is interesting because it shows the attempts in checking the state-of-the-art methods to see if they work with speech. Secondly, they propose a fingerprint with an algorithm of their own creation, involving the use of MFCCs, normalised energy, and the delta coefficients referred to normalised energy. In the end, the results are compared.

The experiment has been conducted on the dataset of the TRECVID 2009, with the goal of designing an effective technique for audio copy detection, which turned out to be the study of a proper fingerprint. In the beginning, as already anticipated, the idea was to consider a solid reference and readapt parts of the process. The authors thought that a binary fingerprint based on energy-difference could be that proper reference.

The extraction pipeline is the following:

- pre-processing: the audio signal is low-pass filtered at frequency $F = 4KHz$, then divided into 25 ms windows with and overlap of 10 ms; then, a pre-emphasis is applied, with factor 0.97; after that, the signal is windowed, with a Hanning window, and then the Fourier Transform can be computed
- transformation: considering the spectrum, only the frequencies between 300 Hz and 3000 Hz are taken into analysis, and the spectrum itself is divided into 16 mel-scale

bands

- actual fingerprint extraction: each band is filtered by a triangular window in order to compute the energy, and then the energy differences between the sub-bands will determine the fingerprint itself. We refer to the paper and quote the formula:
  If EB(n,m) represents the energy value of the nth frame at the mth sub- band, then the mth bit F (n, m) of the 15-bit fingerprint is given by $F(n,m) = 1$, if $EB(n,m) - EB(n,m+1) > 0$. Otherwise, $F(n,m) = 0$.

The subdivision in this small amount of bands, according to the authors, allows a higher likelihood of having all the bits matching.

While in last section we briefly said that binary fingerprints, like this same approach, are suitable for the use of distances like the Hamming one, Héritier et al. have another approach in mind for the matching technique, which involves hashing. Considering the fingerprint f_p for the k-th frame, then the function used is:

$$hash(f_p) = k \tag{1.1}$$

This means that for every possible fingerprint, there could be a match just by checking the hash value. If no match occurs, then the result k will be equal to -1. Note that the function is not a proper hash, since if any frames of the fingerprint is identical to another, the function returns a list of indexes.

The second approach they use for comparison is called Nearest-Neighbor Based Fingerprint (NNBF), and aims to be more efficient especially in the matching phase.

The reason behind this name lies in the fact that the authors wanted to put more emphasis on the proximity between frames than on the combination of features used to extrapolate the similarity measure and then evaluate the various proximities. However, we must not forget that proximity itself is certainly more effective and involves more efficiency in the matching phase depending on what is extracted earlier.

The steps taken into account for computing the similarity measure are the following:

- the pre-processing phase does not change
- same for the transformation, the choice of keeping and enhancing frequencies most heard by human auditory system is kept
- instead of energy difference, twelve cepstral coefficients are computed; energy is still taken into account, but just as a parameter and it is normalised to a certain

gain; alongside these parameters, also the delta coefficients, so the derivatives, are computed as well

In this scenario, the used distance is computed as follows: considering $q_k[1], \ldots, q_k[n]$ as the cepstral parameters of the k-th query frame and $t_j[1], \ldots, t_j[n]$ as the cepstral parameters of a j-th frame, to each test frame are assigned k values, defined as

$$d_{k,j} = \sum_{i=1}^{n} \mid q_k[n] - t_j[n] \mid \tag{1.2}$$

which is basically the definition of the Manhattan distance. Then, the test file is re-mapped to the query file according to which frame was the closest:

$$\hat{t}_j = q_{\hat{k}}, \text{ where } \hat{k} = \arg\min_k d_{k,j} \tag{1.3}$$

. Lastly, the hash in eq. (1.1) is computed on $\hat{t}_j$ instead than on the original test file, to avoid any not-found frame mapped to -1. This re-assignment to the closest frame in the query file explains how the name for NNBF was decided.

This fingerprint in itself is not so efficient from a computational point of view, but this issue is solved either using a binary tree search algorithm or, cutting straight to the point, using a GPU.

Despite this problem, the experiments conducted showed a better result: the evaluation criterion adopted was the minimal Normalised Detection Cost Rate (NDCR), computed as:

$$NDCR = P_{miss} + \beta \cdot FPR \tag{1.4}$$

where P_{miss} is the probability of a miss error, i.e. the False Negative Rate (FNR), FPR is the False Positive Rate (FPR), and β is a constant depending on test conditions.

The results are already quite satisfactory on different scenarios with different datasets in Table 1.1 and Table 1.2:

Transform	1	2	3	4	5	6	7
min NDRC	0.007	0.007	0.030	0.022	0.060	0.053	0.053
Threshold	1.26	1.04	1.30	1.88	0.55	0.76	0.52

Table 1.1: Minimal NDCR for FPR=0 for Haitsma and Kalker's fingerprints

Transform	1	2	3	4	5	6	7
min NDRC	0.007	0.000	0.007	0.007	0.022	0.000	0.030
Threshold	14	14	14	14	15	23	14

Table 1.2: Minimal NDCR for FPR=0 for NNBF

1.2.2. Mixed Techniques

Here, a couple of interesting approaches will be shown. It will be noticeable that, even though the feature extraction for construction the fingerprints is still mainly manual, there are some nice intuitions on the use of DNNs.

An interesting approach was proposed by Matrouk et al. in [12], where the extraction is still manual, but the matching phase is managed by an artificial neural network.

Their task was to use speech fingerprints both to identify isolated words, and for speaker recognition. So, when mentioning the collection of parameters for the fingerprint extraction, they refer them to singular words.

The peculiarity of this experiment is the collection of data not only referred to time or frequency characteristic, but also to statistical or electronic ones as well. We can synthesize these parameters in three groups:

- statistical parameters: the mean and standard value of every file is taken into account, considering audio samples as population to be analysed
- electronic parameters: dynamic range and crest factor, i.e., respectively, the ratio between highest and lowest amplitude of the signal, and between the highest amplitude and its Root Mean Square value (RMS) value, both measured in dB
- classical time and frequency features: Zero Crossing Rate (ZCR) to outline the time domain behavior of the signal, and its Power Spectral Density (PSD) for describing the the power of the signal along the frequencies.

The sequence of all these parameters represents the fingerprint.

In the matching phase, the neural network involved was structured as follows:

- the input layer, with six neurons, so that there was one for each parameter representing the fingerprint
- the hidden layer, with thirty-six neurons
- the output layer, consisting in a neuron for the person identification and a neuron

for the word identification. The chosen activation function is the hyperbolic tangent sigmoid one

The network is fed with an input matrix, representing the parameters of the spoken word by a specific speaker, and trained with a back propagating algorithm which updates weights and bias function according to Levenbert-Marquardt optimization. The evaluation metric of the experiments consists in a recognition ratio of word-person identification, and the results are the ones in Table 1.3:

Word	Recognition Ratio(%)
Yes	100.0
No	100.0
Up	98.8
Down	97.6
Left	100.0
Right	100.0

(a) Recognition ratios for words. Tests conducted on 500 samples.

Person	Recognition Ratio(%)
Esraa	100.0
Alaa	100.0
Ayyam	97.8
Heyam	98.6
Mohammad	97.5

(b) Recognition ratios for people. Tests conducted on 600 samples.

Table 1.3: Results of experiments in [12]

Another noticeable approach is the one proposed by Faraji et al.: in this case [13], speech fingerprints are used for speech enhancement, occurring by training Generative Adversarial Networks (GANs).

In this case, there are not significant words to spend on the fingerprints themselves, since they are a combination of MFCCs and Normalized Spectral Subband Centroids (NSSCs), and their first and second derivative.

Also in this situation, an audio preparation regarding application of Short Time Fourier Transform (STFT) and division of the spectrogram in sub-bands is performed, with the goal of emphasizing the most relevant characteristics of the audio.

The most interesting part is the incorporation of these features for training the GAN: the goal, in detail, is to feed the fingerprints of noisy audio, as long as examples of both clean audio and plain noise. Eventually, the generator learns how to elaborate the enhanced signal, which will be finalized with a Wiener Filter (starting from the information contained in the STFT of that same noised speech file).

This neural network is constructed as follows:

- the discriminator has three hidden layers with 512 nodes each, and Leaky Rectified

Linear Unit (Leaky-ReLU) is used as activation function after each layer; also a dropout is applied, by a factor of 0.2;

- the generator follows the same structure, with the only slight difference in the activation functions, which are simple Rectified Linear Units (ReLUs);
- the output activation function, in both cases, is a sigmoid, to perform better discrimination;
- the training is performed with an Adam optimizer provided with a learning rate of 10^{-4}, batch size set to 128, and 50 epochs.

The evaluations are conducted using the Source to Distortion Ratio (SDR), Short-Time Objective Intelligibility (STOI) and Perceptual Evaluation of Speech Quality (PESQ), which can change from scenario to scenario (i.e. signals which are noisy according to different SNRs). Also different feature combinations can concur to change performances in this experiment, since they can affect the feature vector size, the training time per epoch, and number of network parameters.

The results, metric by metric, are shown in Table 1.4, Table 1.5, Table 1.6. All of them point out better performances if they show a high value.

Feature Set	$-5dB$	$0dB$	$5dB$	$10dB$	$15dB$
Noisy	1.13	1.40	1.72	2.07	2.43
STFT	1.71	2.12	2.52	2.82	2.99
NSSC	1.56	2.07	2.48	2.80	3.07
MFCC	1.69	2.11	2.60	2.90	3.12
STFT + NSSC	1.77	2.20	2.60	2.90	3.04
STFT + MFCC	1.83	2.27	2.64	2.94	3.14
MFCC + NSSC	1.82	2.25	2.63	2.96	3.21

Table 1.4: Average PESQ results for all noise types at various SNRs: better performances are reached when combining MFCCs with the other features, and of course when there are favourable SNRs.

1.2.3. Deep Neural Network Techniques

We conclude the state-of-the-art attempts by introducing an approach which is not using manual features for the actual fingerprints; they now result in a sequence of high-level features extracted by the network itself.

This proposal worth mentioning is the one of Martín-Gutiérrez, Hernández-Peñaloza,

Feature Set	$-5dB$	$0dB$	$5dB$	$10dB$	$15dB$
Noisy	-5.21	-0.34	4.62	9.61	14.6
STFT	3.80	7.71	11.5	15.1	17.8
NSSC	3.05	7.10	10.8	14.0	16.5
MFCC	3.17	6.96	10.7	14.3	17.2
STFT + NSSC	4.16	7.95	11.7	15.2	17.9
STFT + MFCC	4.18	7.96	11.7	15.3	18.3
MFCC + NSSC	4.11	7.80	11.6	15.2	18.5

Table 1.5: Average SDR results for all noise types at various SNRs: for the more favourable ones the values are nearly overall satisfying for every feature or combination of features.

Feature Set	$-5dB$	$0dB$	$5dB$	$10dB$	$15dB$
Noisy	0.56	0.67	0.78	0.87	0.93
STFT	0.69	0.79	0.87	0.92	0.94
NSSC	0.64	0.76	0.85	0.90	0.93
MFCC	0.68	0.79	0.86	0.91	0.94
STFT + NSSC	0.70	0.80	0.88	0.92	0.94
STFT + MFCC	0.71	0.81	0.88	0.92	0.95
MFCC + NSSC	0.70	0.80	0.88	0.92	0.95

Table 1.6: Average STOI results for all noise types at various SNRs: also for 5 dB in this case the metric is overall satisfying.

Menéndez, Álvarez in [14].

Their goal is to perform speaker diarization, so to detect and to identify the speaker at a certain point of the speech.

In order to achieve this, for every audio file a voice activity detection model splits speech and non-speech parts, then both background music and noise are filtered out.

At this point, the speech recorded is further preprocessed, computing the Mel-Spectrogram. All the spectrograms, at this point, are fed to a DNN. This architecture is a Convolutional Neural Network (CNN), with three layers layers consisting in:

- convolutional layer itself
- max pooling layer
- Fully-Connected Layers (FCLs) at the end of the whole convolutional part

The network is trained using categorical cross-entropy, and an Adam optimizer is chosen.

After the training process is concluded, the output of the last hidden layer is chosen

as the representation of the mel-spectrogram, and that can be considered as the speech fingerprint of this project. This embedding layer allows to extract the fingerprints.

The matching phase is performed after a technique preventing false positives, and it involves a distance-based algorithm as the general pipeline suggests. A new recording will be associated to a speaker if and only if the distance is below a certain threshold, otherwise a new speaker has been encountered.

The experiments are conducted on a number of seven speakers at first, then twenty. The metrics used are the common ones of machine learning theory, in order to be sure that the predictions are correct among all IDs, and they are: accuracy, precision, recall, F1 score.

In Table 1.7 and Table 1.8 the results are shown, with the metrics gathered two by two: the high percentages indicate an overall nice performance.

Dataset Name	Accuracy (Train ‖ Val)	Precision (Train ‖ Val)
LibriSpeech ASR corpus[1]	96.38 ‖ 98.82	96.97 ‖ 99.06
LibriSpeech ASR corpus[2]	86.84 ‖ 84.05	89.72 ‖ 87.68
CSFA	64.32 ‖ 63.68	65.25 ‖ 64.71

Table 1.7: Comparison among the experiments with different datasets training the system. 1: corpus referred to 7 speakers. 2: corpus referred to 21 speakers.

Dataset Name	Recall (Train ‖ Val)	F1-Score (Train ‖ Val)
LibriSpeech ASR corpus[1]	95.35 ‖ 98.45	96.40 ‖ 98.76
LibriSpeech ASR corpus[2]	84.58 ‖ 80.74	83.96 ‖ 84.06
CSFA	3.97 ‖ 64.03	64.60 ‖ 64.36

Table 1.8: Comparison among the experiments with different datasets training the system. 1: corpus referred to 7 speakers. 2: corpus referred to 21 speakers. The training recall for CSFA is unexpectedly low, but the validation one is still acceptable.

1.2.4. Speech and Music Discrimination

One of the problems that can be faced when talking about speech fingerprinting is the discernment between music and speech. It is not so weird to think of a scenario in which a recording has background music, and the risk is to extract features that describe the speech, but also contain information about the music of which we are not interested.

A work that can briefly explain how it is not trivial to find features that work only and exclusively for the speech is the one in [10].

Carey et al. focused on this issue, and decided to analyse a set of classic features, and their delta, so to use them for the discrimination of music against speech, and evaluate the consequent results. The discrimination is performed by modelling a Gaussian Mixture Model (GMM) for speech and music respectively, through the use of those same features.

The specific organisation of the features was as follows.

At first, Carey et al. filtered every spectrogram with a mel-scale filterbank, which had nineteen filters. Then, the following features were chosen: cepstral coefficients, filterbank energy, pitch, zero-crossing

Also the delta were computed, over a specific amount of frames. Then, apart from cepstral coefficients, mean and standard deviation were computed as well.

The metric used to evaluate the discrimination was the EER, and the results are the ones in Table 1.9.

Feature	Statistic	Music	Speech	EER (%)
Amplitude	μ	0.00	0.00	3.5
	σ	1.14	2.50	
Δ Amplitude	μ	0.00	0.00	2
	σ	0.02	0.53	
Zero-Crossing	μ	0.18	0.17	20
	σ	0.10	0.13	
Δ Zero-Crossing	μ	0.00	0.00	13
	σ	0.19	0.33	
Pitch	μ	75	52	9
	σ	27.3	21.7	
Δ Pitch	μ	0.28	0.05	7.5
	σ	61.5	53.5	

Table 1.9: Statistics of the GMMs: the EER results relatively high for every feature, so it is likely this could not be the best approach for this type of discrimination

Regarding cepstral coefficients and their deltas, a Detection Error Tradeoff (DET) graph is shown in Figure 1.3, pointing out better results than the ones in Table 1.9.

The features, taken as themselves, are not enough for a satisfactory music and speech discrimination. Both Table 1.9 and Figure 1.3 are showing that there is plenty of room for improvement. Also Carey et al. point out the fact that other features, or a combination of feature, should be a better hypothesis for eventual future works.

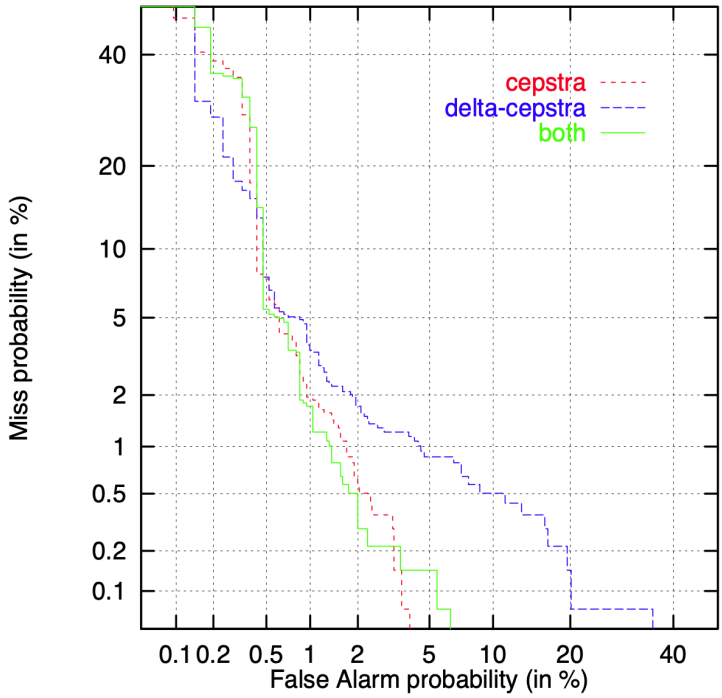


Figure 1.3: DET graph for cepstral coefficients: the point in which miss probability and false alarm probability are equal seems a bit lower than the EERs in the table, still it seems there could be some more improvement for a smoother discrimination

1.3. Summary

The state of the art shows that, even though the general pipeline for audio fingerprinting and matching is quite simple and intuitive, there are plenty of ways to specifically implement the steps, especially when it comes to the extraction phase.

While there was no need to spend many words on the approaches conducted on music, also due to the fact that they reached some satisfying performances, things become a bit more difficult when it comes to speech: with the exception of keyword spotting, no approach addressed the joint recognition of speech and speaker identity, which is the basic requirement of a successful speech fingerprinting and matching.

A lot depends on the task as well, since results are reported in the field of speaker diarization or speech enhancement, and for this task the DNN approaches show satisfactory results [33][34][35].

So again, we noticed that classic features have difficulty in accomplishing the goals of our thesis, but we also recognized the potential of DNNs in this task. At this point, the question is:

Is there any chance to develop effective fingerprints acting similarly to music, so recognizing both speaker and the spoken sentence?

and a possible answer could be based on DNNs.

2 | Theoretical Premises

In this chapter we deal with the main arguments that determined the design of the thesis.

More specifically, the work in itself consists of two ideas:

- the first, which draws on the state of the art in the field of music, to verify the need to study a more speech-compliant approach;
- the second, a proposal by the candidate that draws inspiration from the state of the art attempts in the field of speech.

2.1. Energy Difference Fingerprint

This approach, i.e. the one from Haitsma and Kalker [7], is well known in the field of audio fingerprinting.

It is successful from several points of view: accuracy, robustness, reliability, not to mention the fact that the fingerprint itself, in addition to meeting the definition of compactness, is also a very light data structure, as it is a binary representation of the characteristics of the audio file.

2.1.1. Concept

Haitsma and Kalker's idea, which makes their fingerprint especially successful in terms of robustness, stems not only from the need to have to recognize an audio file in the shortest possible time, but from the ability to recognize that audio file despite possible accidents.

More specifically, we are not only talking about noisy contexts, but also about the way the same audio is reproduced.

The most trivial example is to consider conversion formats: a .wav format file, from an engineering point of view, is very different from an .mp3 file, since the latter conversion format is based on lossy coding, while .wav is based on a much more precise Linear Pulse Code Modulation (LPCM).

However, to the human ear, these differences are not perceptible: whether you listen, for example, to an opera aria in the first format or in the second, it will not change much. Certainly, with a good audio system, you will notice that the mp3 format is qualitatively worse, but there will be no doubt that you are playing the same song anyway.

This requirement is also necessary in our case: it is not important that there is noise in a recording, or that for any investigation you manipulate a .wav, .m4a, .mp3 file: we want a technology that recognizes the speech and the interlocutor, in order to avoid, for example, possible diffusion of false contents.

Since the human auditory system can be reproduced with good fidelity, the idea behind this fingerprint (as well as many others) is to reproduce the perceptual representation that the human being has of sound.

Moreover, we must not forget that, by abandoning a purely mathematical approach to extract the fingerprint, we also abandon some properties of the fingerprint itself: to be clearer, we can no longer speak, for example, of the transitive property, which at first glance could be attractive in the matching phase: given three audio files X, Y, Z, if X is similar to Y and Y is similar to Z, it is not said that X and Z are the same thing.

So, Haitsma and Kalker's idea can be summarized as follows:

for a fingerprint function F, there must exist a threshold parameter τ such that:

$$\|F(X) - F(Y)\| \leq \tau \tag{2.1}$$

if X and Y are similar

$$\|F(X) - F(Y)\| > \tau \tag{2.2}$$

if X and Y are dissimilar

2.1.2. Workflow

As is customary and we have learned by now, Haitsma and Kalker's pipeline is also essentially split into two parts:

- extraction phase
- matching phase

Extraction for the first phase, a graph is proposed by Haitsma and Kalker, giving a general idea:

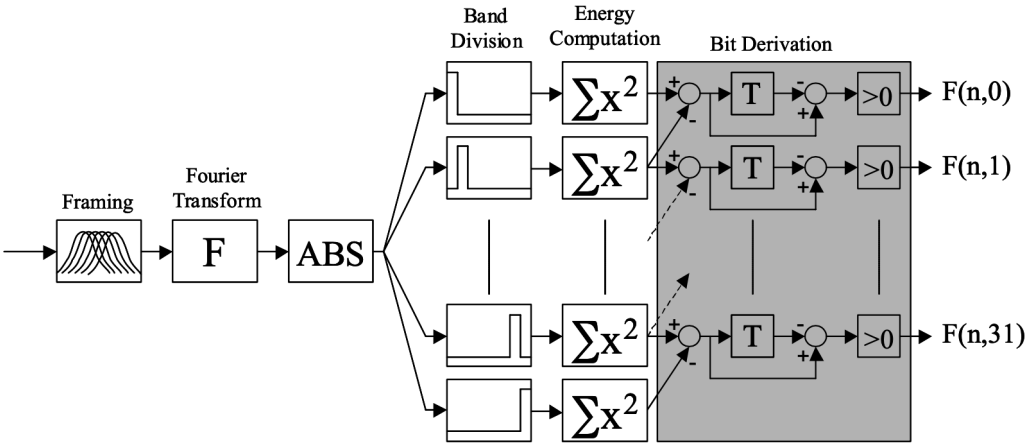


Figure 2.1: Scheme from [7], describing the extraction steps

The goal, more specifically, is to have several sub-fingerprints, each representing a different portion of audio in the same song, that together represent the file in its entirety. Each sub-fingerprint will be 32 bits long.

Keeping in mind that the goal is to build an object capable of resisting possible signal degradations, each audio file is treated in this way:

- the signal $x(n)$ is windowed with overlapped frames $x_l(n)$. Each frame is a Hamming window 0.37 seconds long, and the overlapping factor is $\frac{31}{32}$. In this way, there will be a sub representation of the file every 11.8 milliseconds, regardless of the sample rate, and this should also assure more reliability during the search for a match, since the large overlap will assure that little will be changed for a damaged audio;
- since we want to capture significant properties, we need to be able to observe the audio content in the frequencies domain as well, and that's why STFT is applied: at this point of the workflow, we already have window and only the actual Fourier Transform has to be applied. Only the the absolute value of the spectrum of $X(m, k)$, being m the time frames and k the frequency bins, is kept, since the human auditory system is basically insensitive to phase;
- in order to have, for each window, the sub-fingerprint represented by 32 bits, the signal is windowed with a logarithmic filterbank $F(k, b)$, with k being, again, the frequency bins and b being frequency bands. They are constructed choosing 33 non overlapping bands: the lower bound is 300 Hz, and the upper one is 2000 Hz,

in order to capture the frequencies which are more meaningful to human auditory system. Each filter is equally space in the logarithmic domain, and rectangular, so to help the precise non-overlapping characteristic;

- experimentally, Haitsma and Kalker agreed on the robustness of the sign of energy to many kinds of audio processing. So, the energy of the filtered window signal is computed

$$E(m,b) = [|X_l(m,k)| \cdot F(k,b)]^2 \tag{2.3}$$

and then this difference (i.e., the derivative) is calculated both along time and frequency axis. Given the energy of band b, frame m E(m, b) and the b-th bit of the sub-fingerprint of frame m F(m, b) the bits of the sub-fingerprint are computed as follows:

$$f(E(m,b)) = \begin{cases} 1 & \text{if } E(m,b) - E(m,b+1) - (E(m-1,b) - E(m-1,b+1)) > 0 \\ 0 & \text{if } E(m,b) - E(m,b+1) - (E(m-1,b) - E(m-1,b+1)) \leq 0 \end{cases} \tag{2.4}$$

which, in Figure 2.1, is effectively represented in the delay block section.

The computational cost behind this extraction is limited: since the upper bound of the logarithmic filter is 2000 Hz, there is no need to sample audios at an enormous rate.

The proposal involves a 5000 Hz sampling rate, with a frame length of 2048 samples for the STFT.

Matching Since the fingerprints thus constructed are binary, the most efficient distance describing their similarity is the Hamming distance.

Let there be two binary vectors u, v, of dimension n, then the normalised Hamming distance is defined as:

$$\frac{1}{n} \sum_{i=1}^{n} u_i \oplus v_i \tag{2.5}$$

with $\oplus$ being the XOR operator, which returns as output 0, if $u_i = v_i$, 1 otherwise. This means, the lower the Hamming distance the most similar the two vectors.

There has to be taken care of the fact that, since fingerprints of the same audio could be a bit different, the match occurs not only when the score of the Hamming distance is 0, as already mentioned in the idea of this paper, but every time it is below a certain threshold τ. We also recall that this τ allows to consider acceptable more or less samples:

the higher the threshold, the higher the possibility of encountering false positives.

The question is, how to decide a criterion for establishing this threshold?

The hypothesis that Haitsma and Kalker adopted in their work is as follows: they assume all fingerprints as independent samples and identically distributed, and on the basis of this hypothesis they model the probability of encountering false positives, in relation to an independent variable. This variable is none other than the Bit Error Rate (BER), which indicates precisely the percentage of non-coincident bits on the total. With it, they calculate the decisional threshold, which in fact they are investigating during the project.

Therefore, their work is based on the hypothesis that arbitrary musical pieces do not show any correlation between each other if represented as fingerprints.

Experimentally, in fact, they model the BER probability on a database of fingerprints. This modeling was conducted starting from 10^4 songs stored in a database, and then randomly selecting 10^5 pairs of fingerprint blocks.

The resulting standard deviation is three times larger than assumed, so the initial model of the probability of false positives with respect to BER is correct by a factor of 3.

In the experiments the authors conducted, one arbitrarily decides the value of BER, and based on this choice one evaluates the probability of false positives: more specifically, one reaches a probability of $3.6 * 10^{-20}$ with a BER $a = 0.35$.

Having decided on the criteria by which the similarity between two fingerprints is determined, how do they perform the database search?

After agreeing that the brute force algorithm is possible, but highly time-inefficient, they proceed in the following way: first, they perform an approach also adopted by Wang in [36] that involves searching for the match in a subset of the database, built in such a way that the possibility of finding fingerprints with a BER lower than the decisional threshold. From there, if necessary, more refined techniques are used.

The experiments that the authors conducted consider music tracks degraded in different ways: for example .mp3 coding with different bits per second, equalization, echo addition, time scaling, noise addition, resampling are considered. The results are reported in table 2.1, which takes into account different degradations for the following songs:

- "O fortuna," by Carl Orff
- "Success has made a failure of our home", by Sinead O' Connor

- "Say what you want," by Texas
- "A Whole Lot of Rosie," by AC/DC

Processing	Orff	Sinead	Texas	AC/DC (%)
MP3128Kbps	17, 170	20, 196	23, 182	19, 144
MP332Kbps	0, 34	10, 153	13, 148	5, 61
Real20Kbps	2, 7	7, 110	2, 67	1,41
GSM	1, 57	2, 95	1, 60	0, 31
GSM C/I = 4 dB	0, 3	0, 12	0, 1	0, 3
All-Pass Filtering	157, 240	158, 256	146, 256	106, 219
Amplitude Compression	55, 191	59, 183	16, 73	44, 146
Equalization	55, 203	71, 227	34, 172	42, 148
Echo Addition	2, 36	71, 227	34, 172	42, 148
Band Pass Filtering	123, 225	118, 253	117, 255	80, 214
Time Scale +4%	6, 55	7, 68	16, 70	6, 36
Time Scale −4%	17, 60	22, 77	23, 62	16, 44
Linear Speed +1%	3, 29	18, 170	3, 82	1, 16
Linear Speed −1%	0, 7	5, 88	0, 7	0, 8
Linear Speed +4%	0, 0	0, 0	0, 0	0, 1
Linear Speed −4%	0, 0	0, 0	0, 0	0, 0
Noise Addition	190, 256	178, 255	179, 256	114, 225
Resampling	255, 256	255, 256	254, 256	254, 256
D/A A/D	15, 149	38, 229	13, 114	31, 145

Table 2.1: Hits in database for different kinds of signal degradation. On the left of commas the hits using only the fingerprint, on right of commas when using also the most probable candidates strategy. We can point out that, at least for the most common treatments, in which the recording stays perceptually closer to the original, the system is effective.

Because this approach proves effective in the way it respects the robustness property, a number of concepts were used for our project. In particular:

- the extraction pipeline, in order to start gaining some confidence with the concept of fingerprint and to investigate if it is effective also for speech audio files;
- the hamming distance, because it is the most effective way to calculate the similarity between two binary fingerprints.

The matching phase will be a bit rearranged, since we wanted to focus mainly on the fingerprint extraction technique, thus leaving scalability considerations and requirements open for possible future works.

2.2. Convolutional Neural Networks

2.2.1. Purpose

The need to build CNNs arises from several needs: in the field of deep learning [37], in fact, even if it is fundamental to process data in such a way as to learn the complex patterns that they possess, one of the most classic problems that one encounters is the following: one builds a neural network, for any problem (classification, regression, clustering, reinforcement, etc...), and it learns to recognize the high-level characteristics of the data provided in a training phase; when, however, in the validation phase or directly testing the network on generic data, the predictions are not correct, resulting mainly in the presence of false negatives, i.e. data that would respect the rules of a given scenario, but are not recognized as such. This problem is known as overfitting, i.e., fitting a model that is too close to a specific dataset, and therefore cannot be generalized to unseen data.

Hence, we look for an alternative to architectures such as the Multi-Layer Perceptron (MLP), since they consist of layers whose neurons of the specific layer are all connected to the neurons of the next layer, so a simple architecture. What we need, therefore, is another architecture, which learns the features of the data, not taking all the connection with all the neurons into account, but focusing more on the "big picture" and on spatial relations. In other words, a tool which is able to study complex patterns and to perform task in a 2D (or, more generically, multi-dimensional) space.

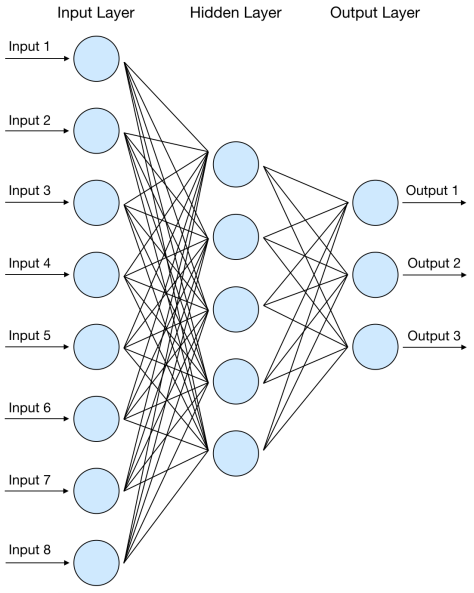


Figure 2.2: Simple overview on how a MLP works: every layer is FCL

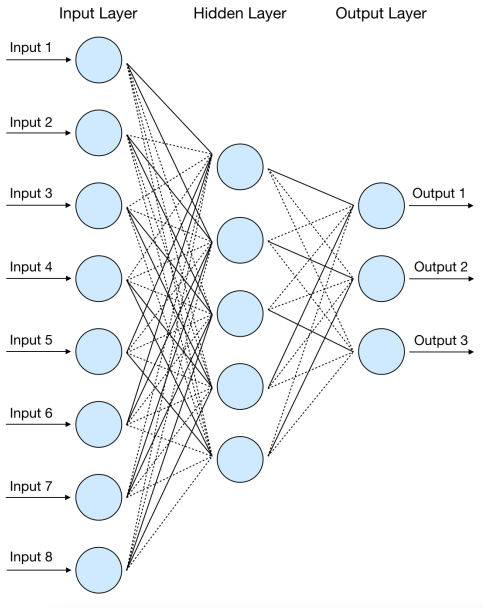


Figure 2.3: Naive Explanation on how a convolutional layer works: with respect to 2.2, not every connection is kept

The first attempts to define CNNs are based on biological processes, and more specifically on what happens in the animal visual cortex. Neurobiology in fact has shown that in this section of the brain neurons respond to stimuli only in a rather narrow section of the visual field, and that this is reconstructed by putting together each subsection, with a partial overlap. Assuming that these neurons can be represented as filters, each responsible for a specific area, it was defined a network that also has narrow filters, with which to evaluate then as a whole a visual field.

Entering a more mathematical dimension, it was agreed that the operation best representing this biological phenomenon is convolution. Let a two-dimensional input $f(x, y)$ and a filter $\omega(s, t)$, then convolution is defined as:

$$\omega * f(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s, t) f(x - s, y - t) \tag{2.6}$$

The winning feature of this operation is that $\omega(s, t)$ can be defined as any numerical pattern: each of these same patterns will be responsible for emphasizing specific features of the "big picture" (the input) under analysis.

2.2.2. General Architecture

Generally, if we look at the architecture of a CNN, the structure is as follows:

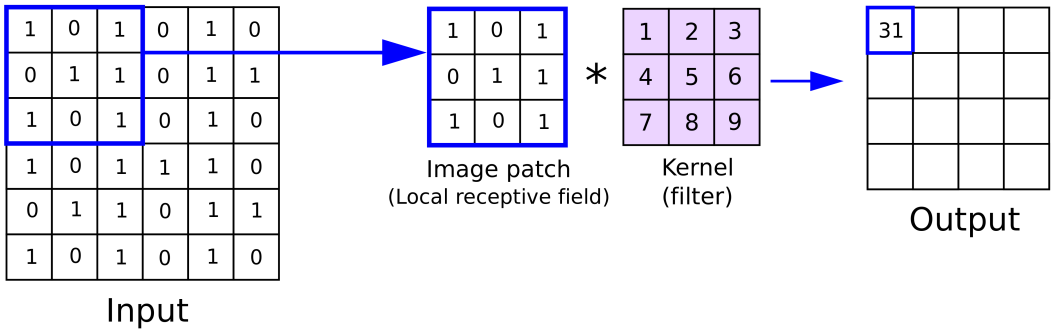


Figure 2.4: Actual functioning of a convolutional layer: the filtering process is the key (courtesy of [38])

- an input layer;
- one or more hidden layers;
- an output layer;

where, as input, we have a tensor of dimensions $(n \times h \times w \times c)$, with n being the amount of inputs, h being the input height, w being the input width and c being the input channels. The output varies according to the purpose of the neural network: in a classification problem, it will be the class, in a high-level feature extraction problem, it will be the feature vector, and so on.

We decided to focus on these partiular elements because, once we are going to implement the concepts and tools of Deep Learning, we will address the problem as if we had images as input.

This hypothesis can be valid especially if we use valid preprocessing techniques: in fact, we remember that applying the STFT to a signal, whatever it is, allows us to evaluate the behavior both in times and frequencies. Assimilating the value of this signal at time t and frequency f, with t = 0, ..., T-1 and f = 0, ..., F-1 (with T and F as hypothetical maximum duration and frequency), to a pixel, we can then use the concepts just discussed.

In the following, we are going to introduce three types of layers: convolutional, pooling, and fully connected.

Convolutional Layer

It is the one that properly performs the convolution on the input. In simple terms, considering only one dimension without loss of generality, be an input $x(n)$ and a filter

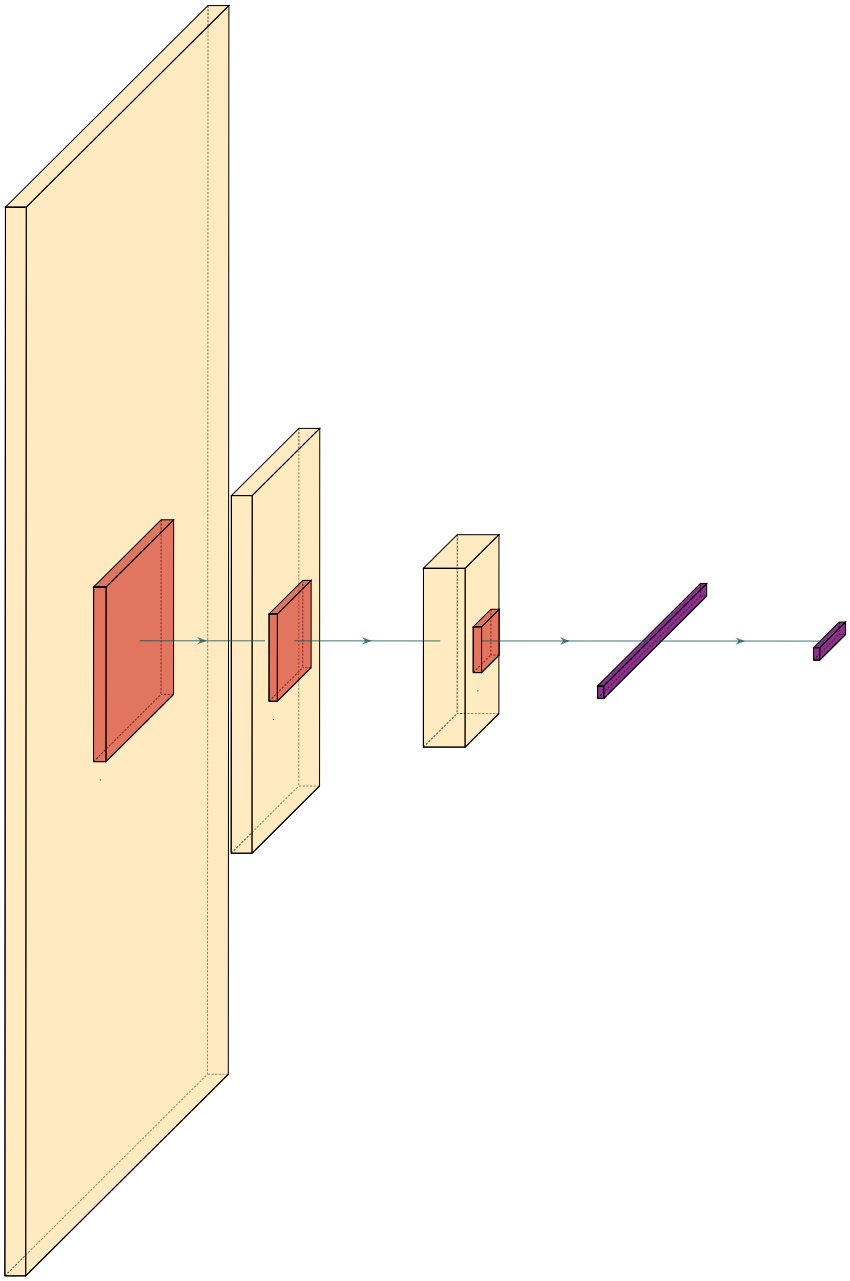


Figure 2.5: Architecture of a generic CNN with three convolutional layers, three pooling layers, two FCLs

$\omega(n)$, then the output s(n), which will be a feature map, is given by:

$$h(\sum_{n=-I}^{I} \omega(n) * x(n)) \tag{2.7}$$

With $h(\cdot)$, being the activation function, and [-I, I] the designated interval in which convolution is performed.

Activation is defined as a function that, in a specific node, defines the output, given as input a data or a sequence of data.

In the field of deep learning there are several activation functions, each with a specific purpose, whose output can activate or not activate the neural units of the next layer according to specific rules.

In this thesis project we use the concept of ReLU, and a variant of it, the Leaky-ReLU.

ReLU is defined as a function that outputs the positive part of the argument. Given an input x, therefore, we will have:

$$f(x) = max(0, x) \tag{2.8}$$

The effectiveness of this function lies in the fact that, since the input will have a certain positive value or zero, the neurons of the next layer will not necessarily be activated. Therefore, in a process of feature extraction, only a part of the most significant outputs of the previous layer will be observed each time.

In the training phase, especially if we use gradient-based learning methods, an activation function constructed in this way allows us to solve problems related to a gradient that becomes too small for the network to learn something, as could happen instead in functions that encourage the output to always have a specific value, or in a narrow range.

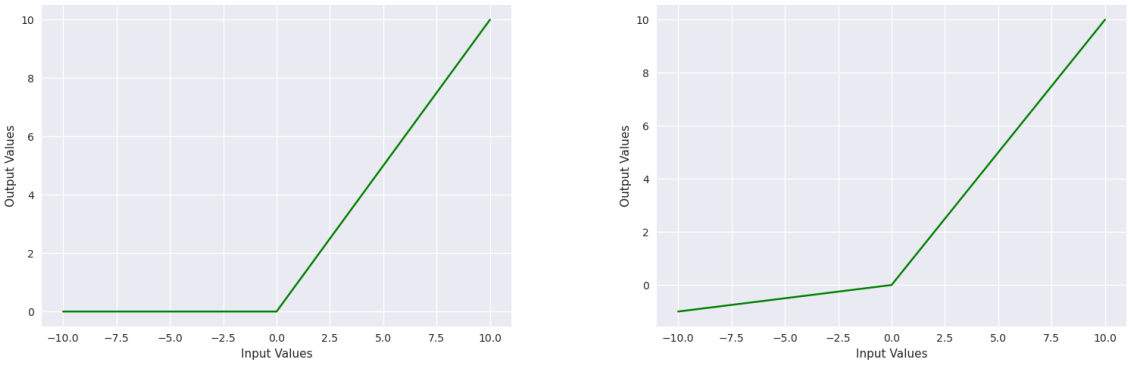
It is widely used in classification problems, and supervised learning in general, and is effective especially when there is the need to train the network on large/complex datasets.

The possible problems that may be encountered relate mainly to the fact that this function is not differentiable in zero, issue resolved by arbitrarily deciding the value of a possible derivative, usually zero or one, and especially the possibility that, for most of the inputs, the neurons are not activated: under these conditions, in a training that makes use of the backpropagation of errors, the gradient has no way of being studied along all the layers and, at a certain point, the state of inactivity becomes too long and overall the

neural network no longer learns the problem to be solved. Although in the training phase there are several hyperparameters that can be checked to solve this drawback, a possible solution is to adopt variants of the simple ReLU.

Between these, we remember the Leaky-ReLU. Let x be an input, then the Leaky-ReLU function $f(x)$ is defined as follows:

$$f(x) = \begin{cases} x & x > 0 \\ 0.01x & \text{otherwise} \end{cases}. \tag{2.9}$$



(a) ReLU function (b) Leaky-ReLU function

Figure 2.6: Graphic of ReLU and Leaky-ReLU

In this way, instead of not completely activating the subsequent units, a gradient is created, which, while remaining small to mean that the output should not activate the subsequent neurons, allows to partially solve the problem just mentioned a few lines above.

Pooling Layer

It may happen that, in a CNN, the outputs of the Convolutional Layers are features that are too sensitive to the position they have in relation to the input images. Not to mention that sometimes we simply have to deal with scenarios where the input data is quite large. In order to manipulate lighter outputs, preserving the characteristics of each data and avoiding the problem of local translation invariance, after a Convolutional Layer it is usual to put the so-called Pooling Layer, which performs a downsampling on the input data.

This operation happens on reduced portions of the input, so basically patches, and the dimension can be chosen during the construction of the neural network. On each patch

one can act, generally, in two ways:

- average pooling, where the average value is computed;
- max pooling, where the maximum value is chosen and kept.

Opting for this or that pooling approach depends on the task and the performance of the neural network.

Fully Connected Layer

At the end of a sequence of Convolutional Layer + Pooling Layer, the output is flattened and then fed to one or more fully connected layers. We are talking about structures whose neurons perform the following operation on the input x(n), giving an output:

$$y(n) = h(x(n)) \tag{2.10}$$

where $h(\cdot)$ is, most of the times, a non-linear function.

All the neurons therefore elaborate all the inputs, unlike the Convolutional Layers. The presence of these layers serves, normally, to complete tasks of classification or regression, choosing an appropriate function of activation $h(\cdot)$.

2.3. Triplet Loss

Specifically talking about supervised learning, and more in detail about classification, when a DNN is trained we are basically searching a good approximation of a rule that has to associate data and some specific targets.

The general way of finding this approximation is to define a loss function and choose a specific hypothesis space: the correct approximation of the rule is the one minimizing the loss function [39].

Depending on the specific task, machine learning and deep learning define many loss functions, each and everyone suitable for this or that problem.

In our work, we acknowledged the fact that we could try this powerful one, named triplet loss.

It is defined as follows: be it a reference input, named anchor sample (A), an anchor matching input, named positive sample (P), and a anchor non-matching input, called

negative sample (N), then we define triplet loss a loss function comparing A, P and N, so that:

- the distance from anchor to positive is minimized;
- the distance from anchor to negative is maximized.

The distance used for this comparison is generally L2 distance, and so triplet loss can be expressed through the following formula:

$$\mathcal{L} = max(\|f(A) - f(P)\|^2 - \|f(A) - f(N)\|^2 + \alpha, 0) \tag{2.11}$$

where $f(\cdot)$ represents the function leading to the output for each sample and α is a regularization parameter.

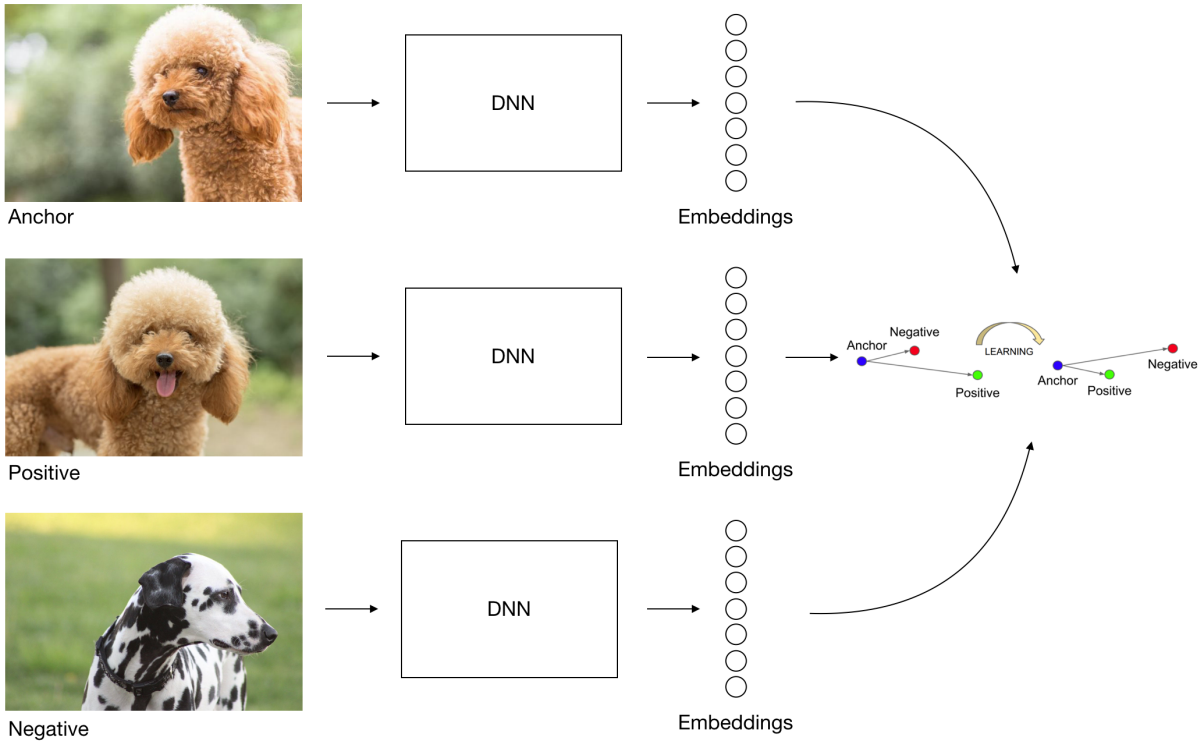


Figure 2.7: A clear example of the samples taken for training a DNN through Triplet Loss. Photos by Google Photo, learning process by [15]

This loss has been used for classifying huge amounts of data, especially in recognition problems: in human face recognition, this function helped in a really effective way in developing neural networks that are now state-of-the-art models [15][16]. In fact, if triplet loss is properly accompanying a DNN, which gives L2 embeddings as output, the network itself will learn a proper bound between different examples, exploiting the concept of

distance and keeping it for dissimilar samples against the similar ones, that will, so to say, “stay together”.

Our goal, then, is to analyze what happens if we use it for speech fingerprinting in a context of audio recognition.

3 | Methods

In this chapter we will explain more in detail the methodologies used to address the problem, as well as the reasons that led us to devise them.

3.1. Overview

We should first mention that, in this thesis, after analyzing the state of the art, we decided to question ourselves about:

1. the technologies for extracting fingerprints, to understand what was the most sensible way for them to be a meaningful representation of speech files;
2. possible technologies to pre-process noisy audio files.

Regarding point 1., we asked ourselves two further questions:

- can the state of the art in music provide an accurate and robust fingerprint for speech as well?
- if the previous question is not answered positively, what can be a valid alternative?

Regarding point 2., instead, we wondered if it was possible to use a technology to facilitate, in some way, the possibility to extract even more robust fingerprints, for example by subtracting the noise that overlaps with speech.

Based on these premises, and on the theoretical premises already discussed in Chapter 2, we formalized the following hypotheses:

- it is worthwhile to make a first attempt at extraction by drawing inspiration from the work of Haitsma and Kalker, in order to reconnect in some way with what has already been seen and consolidated by the state of the art;
- it is necessary to invest energies in the use of a convolutional network that, trained with the promising Triplet Loss, can extract a condensed sequence of high-level features that well describe the speech;

- for the sake of completeness, it is fair to analyze what happens when we try to subtract noise from an audio file: in case a fingerprint has satisfactory requirements in this way, this so-called pre-processing could be decisive for our purpose.

Recalling that it is our task to analyze extraction technologies even on more or less significantly disturbed audio, we will have four approaches:

- the extraction of a fingerprint analogous to the one of Haitsma and Kalker [7], which from here on we will call Energy Difference fingerprint;
- the extraction of an Energy Difference fingerprint, after having pre-treated the audio with a state-of-the-art DNN for audio source separation called Spleeter [17];
- the extraction of a fingerprint from a convolutional neural network, trained on triplet loss in order to extract an appropriate sequence of features, which we can call DNN fingerprint;
- the extraction of the aforementioned DNN fingerprint, after pre-processing the audio with the very same Spleeter network for audio source separation [17].

In Figure 3.1 the pipeline with the methodologies and their interaction is shown. The implementations of these technologies are conducted in Python [40] and C++ [41].

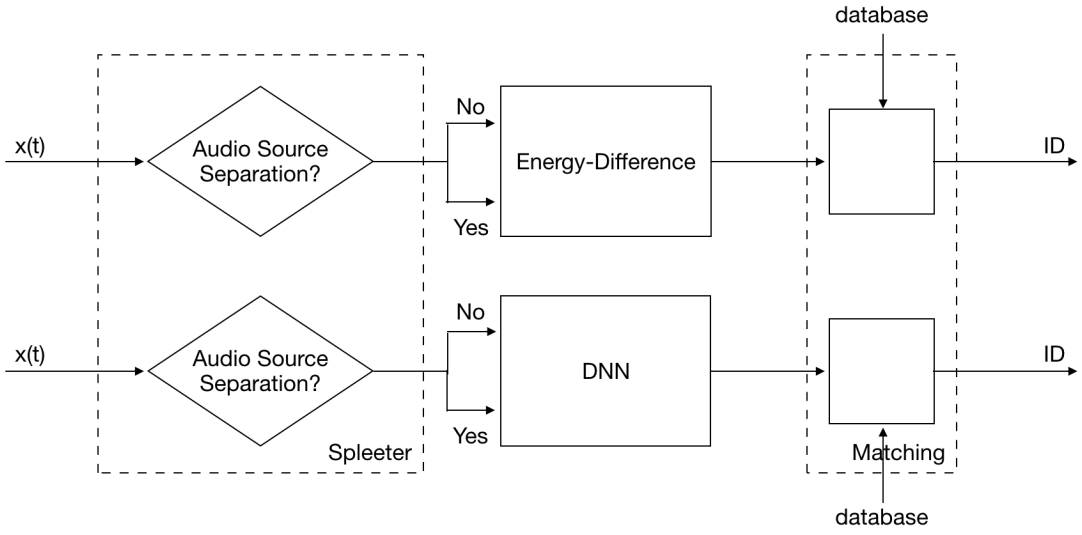


Figure 3.1: Interaction between the technologies: input x(t) is the speech signal, output ID is the estimated ID of x(t); database is considered as the already existing set of fingerprints.

Let us also point out that these methodologies consider the calculation of the similarity measure as well: we anticipate that while the Energy Difference fingerprint is binary, and

therefore the most appropriate measure is the Hamming distance, the DNN fingerprint has arbitrary values, and it was therefore more appropriate to use the L2 distance.

Concerning the matching methodology, it is unique in order to facilitate the comparison between all the previous methodologies: we must keep in mind that the purpose of this thesis is in fact to study more the extraction technologies than the matching techniques.

3.2. Spleeter

Spleeter is a tool developed by Deezer for the separation of music audio tracks [17].

Its implementation, whose simplified scheme is shown in Figure 3.2 is nothing more than an audio source separation using concepts from DNN.

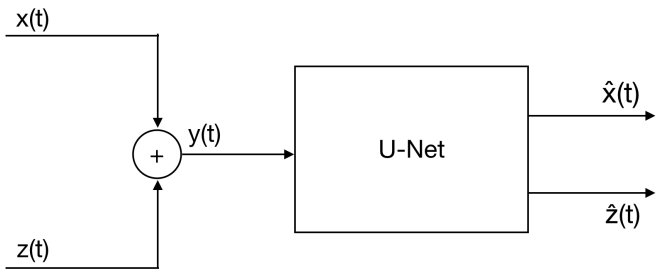


Figure 3.2: Simplified scheme of how Spleeter works. The actual U-Net architecture is not of our interest for this thesis.

Let there be a track x(t), and any signal z(t) which can be any other track in the same recording, then Spleeter:

1. receives the input signal y(t) = x(t) + z(t);
2. reconstructs a signal $\hat{x}$(t) such that

$$\hat{x}(t) \approx y(t) - z(t) \tag{3.1}$$

3. reconstructs a signal $\hat{z}$(t) hat such that

$$\hat{z}(t) \approx y(t) - x(t) \tag{3.2}$$

4. $\hat{x}$(t) and $\hat{z}$(t) are the outputs of the system.

The procedure to estimate $\hat{x}$ and $\hat{z}$ is entrusted to a U-Net, which then first decodes y(t) and then encodes the parts of which it is composed: this concept, in fact, has been extended not only for a simple signal y(t) as built in point 1., but for any signal

$$y(t) = \sum_i x_i(t) + z(t) \text{ with } i = 1; i = 1, \ldots, 3; i = 1, \ldots, 4 \tag{3.3}$$

For each way y(t) is constructed, Spleeter has a U-Net that fits its purpose.

In our specific case, let us assume that we are dealing with signals

$$y(t) = x(t) + z(t) \tag{3.4}$$

And let us further assume that this z(t) can be any disturbance, so a noise, and x(t) is speech.

Among the various U-Nets, Spleeter in fact has one that considers x(t) as the singing, z(t) as the musical accompaniment. We wondered, therefore, whether the methodology Deezer proposed might also be effective for our purposes.

3.3. Fingerprint Extraction and Similarity Measure

3.3.1. Energy-Difference Fingerprint

The scheme for the Energy-Difference fingerprint methodology is shown in Figure 3.3:

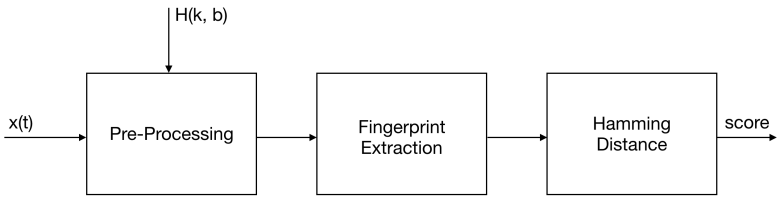


Figure 3.3: Energy-Difference fingerprint methodology: the blocks represent its steps.

Let us now analyse each block:

1. the signal x(t) is pre-processed as follows:
   (a) x(t) is sampled. Since we are dealing with speech, the sample rate does not

have to be high. 8 KHz is the designated Sample Rate (SR):

$$x(n) = x(t)|_{t=\frac{n}{SR}} \tag{3.5}$$

(b) let be a Hanning window $w(n), n \in [0, M-1]$, having length M and a hop size R. Let us denote with m the frame index, then we define the windowed signal:

$$x_m(n) = x(n + mR) \cdot w(n), n \in [0, M-1] \text{ with } m = 0, \ldots, N_w - 1 \tag{3.6}$$

in our case, we chose $M = 2960$, $R = 250$. N_w, i.e. the number of frames, can be arbitrary without losing any characteristic neither of the signal nor the fingerprint we intend to extract.

Then, for each $x_m(n)$, the STFT is computed:

$$X(m,k) = \sum_{n=0}^{N} x_m(n) \cdot e^{j\frac{2\pi k}{N}n} \text{ with } k = 0, \ldots, K-1 \tag{3.7}$$

where $k = 0, \ldots, K-1$ are the frequency bins and N is the number of samples for the STFT calculation. We chose $N = 3000$.

Only the magnitude of the spectrum is kept, so we end up with $|X(m,k)|$;

(c) after that, the signal is filtered with a filter $H(k,b)$, with $b = 0, ..., B-1$. It is divided into B equally spaced bands in the logarithmic domain, and preserves the frequencies from f_{min} to f_{max}. Adapting this concepts to bins, it is defined as follows:

$$H(k,b) = \begin{cases} 1 & \text{if } k-1 \leq b < k \\ 0 & \text{elsewhere} \end{cases} \tag{3.8}$$

A representation of $H(k,b)$ is found in Figure 3.4: we chose $f_{min} = 300\ Hz$, $f_{max} = 2\ KHz$.

The fully pre-processed signal is thus obtained:

$$Y(m,b) = |X(m,k)| \cdot H(k,b) \tag{3.9}$$

2. the fingerprint is extracted:

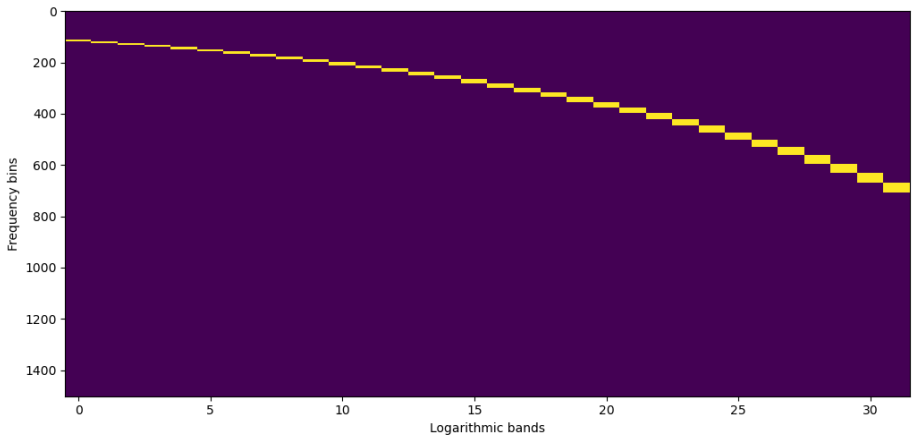


Figure 3.4: Representation of $H(k, b)$ for 32 bands

(a) the energy of the filtered signal is computed:

$$E(m, b) = |Y(m, b)|^2 \tag{3.10}$$

(b) the fingerprint in itself is then computed according to the original definition in [7]:

$$F(m, b) = \begin{cases} 1 & E(m, b) - E(m + 1, b) - (E(m, b - 1) - E(m + 1, b - 1)) > 0 \\ 0 & \text{else} \end{cases} \tag{3.11}$$

3. the Hamming distance is computed. Let a fingerprint $F_1(m, b)$, shifting on a second fingerprint $F_2(m, b)$, then the Hamming Distance d_i for the i-th shift is defined as:

$$d_i = \frac{1}{N} \sum_{j=i}^{M-1} F_{1,i,j} \oplus F_{2,i,j} \tag{3.12}$$

with M being the minimum number of time frames, N being the normalization factor, for understanding the error with respect to the total number of bands, and $\oplus$ being the XOR operator, which returns as output 0 if $F_{1,i,j} = F_{2,i,j}$, 1 otherwise.

We theorised this two-dimensional similarity measure, since the number of bands B is designed as fixed, while M is not. So, the fingerprint with minor number of frames M shifts on the other one, where the shift index is denoted by i in Equation (3.12).

We end up, then, with a vector of distances $d = d_i$ of length $M_{greater} - M_{smaller} + 1$.

3.3.2. DNN Fingerprint

The scheme for the DNN fingerprint methodology is the one in Figure 3.5:

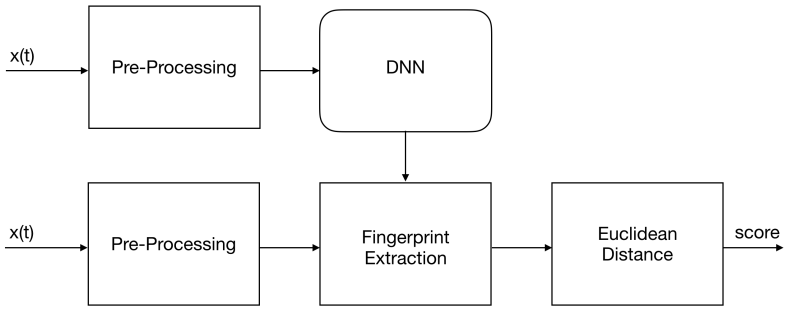


Figure 3.5: DNN fingerprint methodology: the blocks represent its steps.

Let us now analyse each block:

1. the signal $x(t)$ is pre-processed as follows:

   (a) $x(t)$ is sampled. Since we are dealing with speech, the sample rate does not have to be high. 8 KHz is the designated SR:

   $$x(n) = x(t)|_{t=\frac{n}{SR}} \tag{3.13}$$

   (b) $x(n)$ is segmented. In order to prevent eventual silences at the beginning, we end up with:

   $$x_{seg}(n) = x(T) \tag{3.14}$$

   being T the temporal interval T: $[1, 5]s$ sampled at SR;

   (c) let be a Hanning window $w(n), n \in [0, M-1]$, having length M and a hop size R. Let us denote with m the frame index, then we define the windowed signal:

   $$x_m(n) = x_{seg}(n + mR) \cdot w(n), n \in [0, M-1] \text{ with } m = 0, \ldots, N_w - 1 \tag{3.15}$$

   in our case, we chose $M = 512$, $R = 256$. Since every $x_{seg}(n)$ has the same time length, N_w is constant as well. According to the other parameters, $N_w = 125$.

Then, for each $x_m(n)$, the STFT is computed:

$$X(m,k) = \sum_{n=0}^{N} x_m(n) \cdot e^{j\frac{2\pi k}{N}n} \text{ with } k = 0, \ldots, K-1 \tag{3.16}$$

where $k = 0, \ldots, K-1$ are the frequency bins and N is the number of samples for the STFT calculation. We chose $N = 512$.

(d) we compute the power spectrogram $P(m,k)$ of the signal by applying the logarithm to the signal energy:

$$P(m,k) = log_{10}(|X(m,k)|^2 + \epsilon) \tag{3.17}$$

with $\epsilon > 0$ being a small constant to avoid logarithm of 0.

(e) the final pre-processed signal is normalised toward its mean, its standard deviation, and kept within the interval $[0, 1]$. Let:

$$P_{no}(m,k) = \frac{P(m,k) - \mu(P(m,k))}{\sigma(P(m,k))} \tag{3.18}$$

then:

$$P_r(m,k) = \frac{P_{no}(m,k) - min(P_{no}(m,k))}{max(P_{no}(m,k)) - min(P_{no}(m,k))} \tag{3.19}$$

2. the DNN is built as follows:

   (a) three convolutional layers, having Leaky ReLU as activation function, with pooling and dropout;

   (b) a final convolutional layer, just with its Leaky ReLU activation function;

   (c) two FCLs;

   (d) a final L2 normalisation layer.

   Figure 3.6 shows the architecture in a more intuitive way, Table 3.1 shows the parameters.

   It is then fed with a certain amount of pre-processed signals $P_r(m,k)$. The batch size is 16, and the network is trained on triplet loss for 50 epochs.

   The input size is $n_{data} \times (N/2+1) \times M \times 1$, with N and M being mentioned at point 1.d. and 1.c., respectively. 1 is the number of channels.

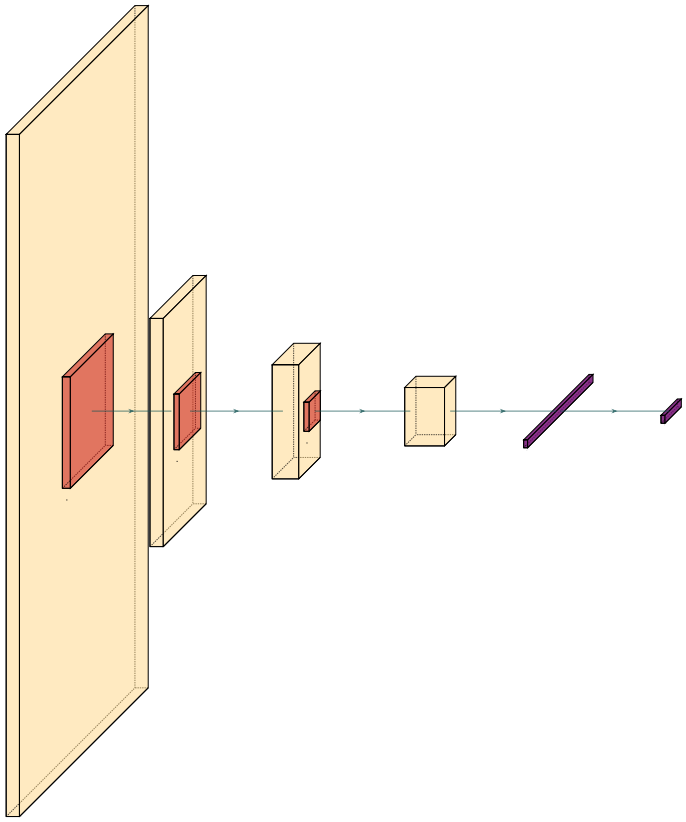


Figure 3.6: DNN structure

Layer	Output Channels	Output Shape	Stride	Neurons
Conv1	128	(257×126)	1×1	None
MaxPool1	128	(86×42)	3×3	None
Dropout1	128	(86×42)	None	None
Conv2	128	(86×42)	1×1	None
MaxPool2	128	(43×21)	2×2	None
Dropout2	128	(43×21)	None	None
Conv3	256	(43×21)	1×1	None
MaxPool3	256	(22×11)	2×2	None
Dropout3	256	(22×11)	None	None
Conv4	512	(22×11)	1×1	None
FCL1	None	128	None	128
FCL2	None	32	None	32
$L2_{norm}$	None	32	None	None

Table 3.1: Parameters of the DNN used for DNN fingerprint extraction

The output size, i.e. the output size of this fingerprint, depends on the number of neurons Q of the last FCL. We chose $Q = 32$;

3. the trained network is the tool we use for extracting high level features, which then will constitute our fingerprint. So we obtain:

$$F(q) = \mathcal{F}(P_r(m, k)) \tag{3.20}$$

where $\mathcal{F}(\cdot)$ is the whole transformation representing all the layers, taking $P_r(m, k)$ in input and leading to the final embedding, which constitutes our fingerprint itself;

4. let two fingerprints $F_1(q)$ and $F_2(q)$, then we compute L2 as:

$$L2 = \sqrt{\sum_{i=1}^{Q} \|F_{1,i}(q) - F_{2,i}(q)\|^2} \tag{3.21}$$

In order to have a clearer idea in mind on the how Triplet Loss works from a more practical point of view, Figure 3.7 shows a 2-D representation of the fingerprints.

3.4. Matching Phase

Considering all the fingerprints in a database DB and the fingerprint of a query file q, the matching algorithm we use is a brute force one.

Assuming the presence of a query fingerprint F_q, the similarity measure is computed between F_q and all the fingerprints in a database F_{DB}.

After that:

1. best matching item $DB_{\hat{i}}$ is the one with index:

$$\hat{i} = \arg\min_i d(F_q, F_{DB_i}) \tag{3.22}$$

and distance:

$$d_{\hat{i}} = d(F_q, F_{DB_{\hat{i}}}) \tag{3.23}$$

2. in order to determine if a correspondence is found, we further apply a threshold τ:
   - if $d_{\hat{i}} \leq \tau$, a match is found between q and $\hat{i}$;

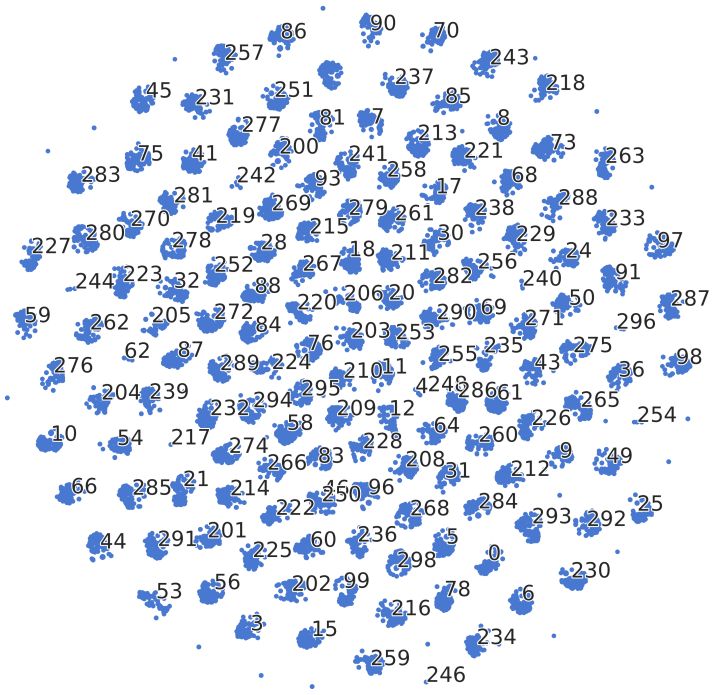


Figure 3.7: Fingerprints extracted by the proposed neural network, plotted in 2D space after a dimensionality reduction. The reduction itself is performed with t-distributed Stochastic Neighbor Embedding (t-SNE) algorithm.

- if $d_{\hat{i}} > \tau$, no match is found in the database for the file q.

The threshold τ can be determined by analysing the Receiver Operating Characteristic (ROC) curve.

4 | Experiments

Here we report the experiments which were carried out, from the data that were used, to the evaluation techniques, including an examination of the results as well as a comparison between all the versions of the project.

4.1. Audio Data Creation

Since it was necessary to evaluate the robustness of the proposed methods, the dataset on which we operated was organized as follows:

1. audio files considered original, without any particular disturbance and well audible;
2. files to which noise has been applied;

4.1.1. Original Audio Files

Original files have been downloaded from website VoxForge.org [42], which is an open speech dataset in which users can upload and download audio files for speech recognition tasks.

The most famous languages and the most important from the linguistic point of view are present, in our case we have limited ourselves to choosing the following subsets:

- English files with no background noise ($\mathcal{X}^{EN}_{SNR=\infty}$)
- Spanish files with no background noise ($\mathcal{X}^{ES}_{SNR=\infty}$)

The files have been chosen trying to vary between speaker genre, age, and sentence which has been recorded. Plus, only good-quality recordings were kept, in order to have a starting point as crystal-clear as possible.

With this premises, we tried to have two nearly-homogeneous originals audio folders, and this is what we obtained:

1. 155 English .wav mono audio files, sampled at $SR = 48KHz$, encoded with LPCM-

16;

2. 172 Spanish .wav mono audio files, sampled at $SR = 48KHz$, encoded with LPCM-16.

Files with Spanish speech are more numerous, due to the higher variety in the recordings than the English ones.

4.1.2. Noisy Audio Files

After having chosen the files, the next step is to corrupt them with noise, according to different SNRs, which are, in decibel (dB):

- 5 dB;
- 0 dB;
- -5 dB;
- -10 dB.

Noises have been kindly provided by Fraunhofer IDMT, Semantic Music Technologies department, which have already been used in the past for other fingerprint and matching tasks.

The ones that are applied on our original files are chosen so that they simulate real-life scenarios, in which there could be the need to investigate on the fingerprint of a specific audio speech. In particular two main groups were chosen:

- generic noises, implying different real-life ones, from windows and doors being opened or closed, to moving objects, bottles and cans being opened, hand mixers, alarm clocks and so on;
- background music, mainly for simulation of commercial or news: for this reason, instead of choosing a potpourri of different music genres, we sticked to only one, which has been decided to be classical.

The algorithm we implemented for the noising process is the following: given an input file $y(t)$ and a noise file $n(t)$ with equal length, first compute a Scale Factor (SF) to be applied to the noise signal.

Let:

$$y_{mean} = \overline{y(t)^2} \tag{4.1}$$

and

$$n_{mean} = \overline{n(t)^2} \tag{4.2}$$

where $\cdot\bar{\cdot}\cdot$ is the mean value of the signals.

SF is then defined as:

$$SF = \frac{y_{mean}}{n_{mean}} \cdot 10^{\frac{-SNR}{10}} \tag{4.3}$$

with SNR being the desired SNR. Then we scale the noise to the desired volume:

$$n(t)_{new} = \sqrt{SF} \cdot n(t) \tag{4.4}$$

And, finally, the signal:

$$\widehat{y}(t) = y(t) + n(t)_{new} \tag{4.5}$$

is obtained.

For the seek of uniformity among all the files, before being saved in the proper directory and with the proper name, the audio is normalised so that it has values in the interval $[-1, 1]$.

All the audios are saved in .m4a format, in order to have light files.

We end up obtaining:

- 17628 English speech files with generic noise;
- 18720 English speech files with background music;
- 20340 Spanish speech files with generic noise;
- 21600 Spanish speech files with background music.

4.1.3. IDs Assignation

During all the aforementioned processes, there are also some steps involving IDs declaration.

We decided to keep this as simple as possible, so:

- the original audio files have a sequential assignation: each recording has its specific number. Regarding the Energy-Difference fingerprint, if there are long recordings, their segments keep the label; in the DNN approach, each segment has its own ID;
- the noisy and de-noised recordings are saved by names containing three numbers:

1. the ID used by the Energy-Difference fingerprint;
2. the ID used by the DNN proposal;
3. a third number, indicating the j_{th} noise segment. It is not useful for the sake of the matching evaluation, but it is a flag for the fact that the same noise has been applied to different IDs, and might be useful to recognise some special noises which might systematically lead to an error.

4.2. Evaluation Criteria

For developing some meaningful considerations about our experiments, we are going to spend some words on the chosen evaluation metrics.

4.2.1. Equal Error Rate

EER is the rate at which the False Positive Rate (FPR) equals the FNR. It is a well known concept used in biometric tasks, and our speech fingerprints and matching problem, since it's taking biological characteristics on which there is the need to perform some sort of acceptance (deciding the proper audio ID), can be considered as one [43].

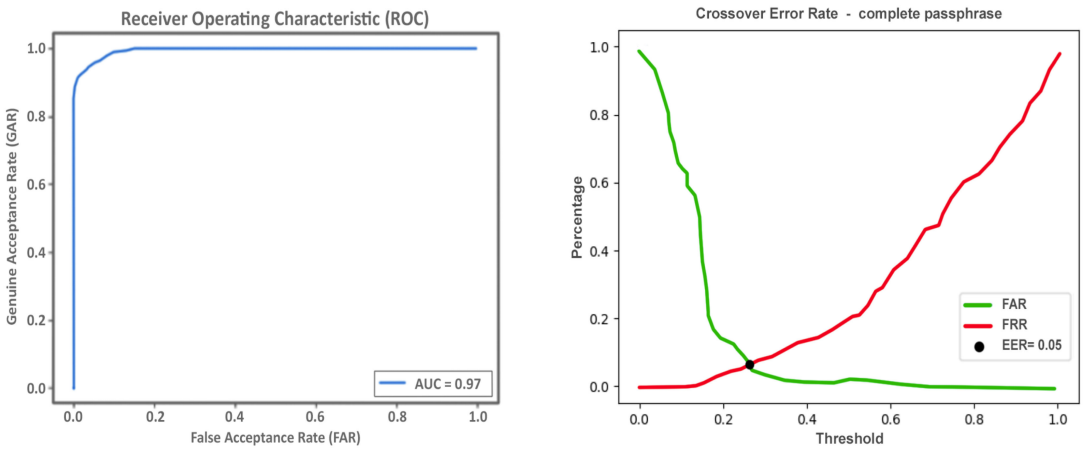
This metric can be easily obtained by the computation of graphs, such as:

1. the EER graph, plotting FPR and FNR behaviour according to different decisional thresholds;
2. the ROC curve, which illustrates the behaviour of the True Positive Rate (TPR) over the FPR, according to different decisional thresholds.

Generally, the smaller the EER the better performance the system has. The value in which the EER is met is usually chosen as the best decisional threshold [39].

There is not a so-called "best way" of decisional threshold attribution, but this is a widely accepted option for benchmarking, since it represents a sort of appropriate configuration in many applications.

Threshold will be reported in our experiments. We must remember, though, that it cannot be considered as a metric, so it is not a way of comparison between how the two fingerprints work: it has to be considered as a parameter of the matching algorithm, and we will try to explain this concept while commenting on the results.



(a) (b)

Figure 4.1: Typical way a ROC curve and a EER curve look, respectively. It is a common habit to examine these graphs, especially in biometric tasks. These are part of results of [44]

4.2.2. Accuracy

This metric is generally able to describe the performance of a system along all classes, and it is defined as the ratio between the total amount of correct predictions and the whole amount of predictions.

We have to distinguish between two possible ways of defining how to compute accuracy:

1. a multi-class problem is faced, so there is the need to investigate, given a sample i over N samples, that the predicted label $\hat{y}_i$ is equal to the ground truth label y_i. This results in:

$$ACC_{mult} = \frac{1}{N}\sum_{i=0}^{N-1} 1(\hat{y}_i = y_i) \tag{4.6}$$

2. a binary-class problem, also in the sense of considering the prediction $\hat{y}_i$ just as either correct (true) or incorrect (false) toward the ground-truth y_i. So, having True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN):

$$ACC = \frac{TP + TN}{TP + FP + TN + FN} \tag{4.7}$$

It is meaningful when the scenario involves well-balanced data among all classes, so this metric is giving a clear idea of the whole big picture: otherwise, for classes having way less samples, there is a higher chance accuracy could be deceptive [39].

Of course, in this case, the higher the metric the better.

In our situation, the accuracy is computed depending on the scenarios:

1. the one in which the whole amount of original audio files is in the database, which can be interpreted as a multi-class problem with accuracy ACC_{mult}. Since the audio data involved is well-balanced among every class, i.e. all excerpts have been corrupted with the same exact amount of noise files, the method offered by scikitlearn Application Programming Interfaces (APIs) [45] is used;
2. the one in which only a partial amount of original audio files is in the database. Here, in order to have an accuracy which actually describes the methods performance, we applied Equation (4.7) to obtain ACC and selected TP, TN, FP, FN according to the following criteria:
   (a) the assertion that the fingerprint of query belongs or does not belong to the database;
   (b) the predicted ID of the query, belonging or not belonging to the database.

We end up with these definitions. A match is considered as:

- TP, if the query belongs to the database, and the predicted ID is correct;
- FP, if the query does not belong to the database, but there is a predicted ID chosen from the ones in the database; if the query belongs to the database, but it is assigned to a wrong ID;
- TN, if the query does not belong to the database, and there is no predicted ID;
- FN, if the query belongs to the database, but there is no predicted ID chosen from the ones in the database.

4.3. Results

Here we show how the experiments went, according to the aforementioned metrics.

For every approach, we decided to evaluate performances in the aforementioned scenarios. More specifically:

1. the whole amount of original audio files is the database, the noiseless ones are the query;
2. the whole amount of original audio files is the database, the noisy ones are the query;

3. only a partial amount of the original audio files is the database, the noisy ones are the query: this is meant to investigate how the decisional threshold is actually effective.

We refer to the decisional threshold as τ, to the accuracy as Acc or Acc_{mult}, as appropriate.

4.3.1. Energy-Difference Fingerprint

Base Test

In order to test the Energy-Difference technology without the use of Spleeter, we did some first experiments confronting the noiseless audios with themselves. All seems to work when all the queries have a match, but it starts having trouble when only 45% has. Besides, by the time we introduce the chance to have true negatives, the system is not able to set a proper threshold, and this is visible for the Spanish speech.

English Tests			
Files in DB	EER (%)	τ	Acc (%)
100%	1.25	1.65	100.00
45%	2.33	1.66	60.90

(a)

Spanish Tests			
Files in DB	EER (%)	τ	Acc (%)
100%	4.08	2.75	100.00
45%	6.32	4.91	47.22

(b)

Table 4.1: Results of base case

Tests conducted where every file in query had a match in the database

In this experiment, we built a database on clean files, and tried to perform speech matching using query files distorted with either noise or music. The corresponding results are in Table 4.2 and in Table 4.3.

It is noticeable from this experiment that the results are not quite satisfactory: the EERs in Table 4.2 and in Table 4.3 show that even for favourable SNRs the system has struggles in ID identification.

The tests show that audio source separation, at least using Spleeter, is worsening the performances: the EER decreases a bit, so we could think there is a slight improvement, but the accuracy drops quite fast. We can also assume that this approach is, somehow, more sensible to noises, on the hypothesis that exerpts of audio with same ID could have been made noisy with the same file.

English Tests with Noise				English Tests with Noise			
SNR (dB)	EER (%)	τ	Acc_{mult} (%)	SNR (dB)	EER (%)	τ	Acc_{mult} (%)
5	14.04	4.97	57.58	5	13.03	4.91	26.02
0	17.68	5.13	57.34	0	16.23	5.06	27.39
-5	20.91	5.28	57.28	-5	19.80	5.25	25.54
-10	24.03	5.41	57.17	-10	23.87	5.38	25.11

English Tests with Music				English Tests with Music			
SNR (dB)	EER (%)	τ	Acc_{mult} (%)	SNR (dB)	EER (%)	τ	Acc_{mult} (%)
5	31.03	5.63	5.13	5	23.41	5.38	2.92
0	37.01	5.84	4.97	0	29.59	5.59	2.97
-5	42.89	6.00	4.63	-5	37.79	5.84	2.84
-10	47.53	6.13	4.36	-10	45.94	6.09	2.94

(a) Not requiring the use of Spleeter (b) Requiring the use of Spleeter

Table 4.2: Results of methodologies requiring the Energy-Difference fingerprint (English): all queries are present in the database

Spanish Tests with Noise				Spanish Tests with Noise			
SNR (dB)	EER (%)	τ	Acc_{mult} (%)	SNR (dB)	EER (%)	τ	Acc_{mult} (%)
5	13.07	6.41	58.74	5	11.58	6.19	16.75
0	15.18	6.69	58.32	0	14.41	6.53	17.81
-5	18.56	7.00	58.03	-5	17.48	6.88	18.08
-10	21.66	7.25	57.86	-10	21.71	7.22	17.47

Spanish Tests with Music				Spanish Tests with Music			
SNR (dB)	EER (%)	τ	Acc_{mult} (%)	SNR (dB)	EER (%)	τ	Acc_{mult} (%)
5	23.13	7.34	9.82	5	15.84	6.69	2.50
0	28.87	7.84	9.05	0	20.89	7.19	3.10
-5	35.69	8.19	8.52	-5	28.46	7.81	3.06
-10	41.39	8.47	8.19	-10	39.16	8.38	3.08

(a) Not requiring the use of Spleeter (b) Requiring the use of Spleeter

Table 4.3: Results of methodologies requiring the Energy-Difference fingerprint (Spanish): all queries are present in the database

Tests conducted where approximately only 45% of the query has a match on the database

In this experiment, we built a database on clean files, and tried to perform speech matching using query files distorted with either noise or music: in this situation, not every query has a match on the database, and this will help us understand how the methodologies work during a more realistic scenario. The corresponding results are in Table 4.4 and in Table 4.5.

If the results were already less than promising in the case where each query had its own match, in a more realistic scenario the performance is more than halved, especially if the speech has been overlaid with background music.

The methodology that require the use of Spleeter, also in this case, does not contribute to any improvement: in the best case, the performances (already unsatisfactory) remain stable.

It is therefore agreed that, with this extraction methodology, τ is ineffective for this task.

Even though the fingerprint is robust toward energy-preserving transformation, as lossy encoding, the presence of background noise leads to the impossibility of performing any kind of *speech* matching.

English Tests with Noise			
SNR (dB)	EER (%)	τ	Acc (%)
5	13.53	4.81	30.50
0	17.09	4.97	30.36
-5	20.60	5.13	30.32
-10	23.54	5.22	30.23

English Tests with Music			
SNR (dB)	EER (%)	τ	Acc (%)
5	30.54	5.47	3.21
0	37.10	5.69	3.06
-5	42.09	5.88	2.75
-10	46.72	6.00	2.46

(a) Not requiring the use of Spleeter

English Tests with Noise			
SNR (dB)	EER (%)	τ	Acc (%)
5	12.62	4.75	14,19
0	15.95	4.91	14.91
-5	19.70	5.06	14.53
-10	23.15	5.19	13.58

English Tests with Music			
SNR (dB)	EER (%)	τ	Acc (%)
5	22.68	5.19	2.31
0	29.02	5.44	2.34
-5	37.06	5.69	2.26
-10	44.86	5.97	2.33

(b) Requiring the use of Spleeter

Table 4.4: Results of methodologies requiring the Energy-Difference fingerprint (English): only 45% of queries is present in the database

Spanish Tests with Noise			
SNR (dB)	EER (%)	τ	Acc (%)
5	14.40	6.47	28.20
0	16.53	6.69	28.02
-5	19.85	6.91	27.85
-10	22.52	7.13	27.73

Spanish Tests with Music			
SNR (dB)	EER (%)	τ	Acc (%)
5	25.82	7.38	5.04
0	30.99	7.84	4.85
-5	36.51	8.22	4.47
-10	42.57	8.47	4.20

(a) Not requiring the use of Spleeter

Spanish Tests with Noise			
SNR (dB)	EER (%)	τ	Acc (%)
5	13.16	6.38	9.81
0	15.54	6.56	10.35
-5	18.52	6.81	9.44
-10	22.38	7.09	9.74

Spanish Tests with Music			
SNR (dB)	EER (%)	τ	Acc (%)
5	17.50	6.75	2.42
0	22.72	7.16	3.04
-5	29.88	7.75	2.40
-10	39.45	8.34	2.40

(b) Requiring the use of Spleeter

Table 4.5: Results of methodologies requiring the Energy-Difference fingerprint (Spanish): only 45% of queries is present in the database

Comments

This approach does not seem as effective as when applied to music, for the following reasons:

- the EER is a bit high, which means that there is a good portion of data that cannot be correctly discriminated;
- the threshold is in reference to a hamming distance normalized only with respect to the variable dimension of the overall fingerprint: thus the distance itself can range from 0 to 32. A threshold that accepts predictions in which 18% of the bits may be wrong indicates that the proposed system may not be the best one;
- the accuracy is low, therefore it is not performing the correct classification; let us consider also the significant difference between the accuracy of the scenarios in which the files have been contaminated with noise and the accuracy in which they have been contaminated with background music: in the most realistic scenario, i.e. the one in which audio has been de-noised and not every file from the query has a match in the database, it oscillates between 9.44% and 14.91% when the disturbance is noise, between 2.26% and 3.04% when the disturbance is background music. This means that the system is hardly able to recognize speech at all, especially if there is also a musical signal.

4.3.2. DNN's Fingerprint

Base Test

In order to test the implementation of our methodologies (in particular, the DNN technology without the use of Spleeter), we did some first experiments confronting the noiseless audios with themselves.

Everything seems to work, also in the case where the fingerprints do not have a match in the database.

English Tests			
Files in DB	EER (%)	τ	Acc (%)
100%	0.00	0.77	100.00
45%	0.00	0.77	99.00

(a)

Spanish Tests			
Files in DB	EER (%)	τ	Acc (%)
100%	0.00	0.49	100.00
45%	0.00	0.77	99.40

(b)

Table 4.6: Results of base case

Tests conducted where every file in query had a match in the database

One can clearly notice that this network was trained with some of these audio files: the accuracy indeed is pretty high despite the noise. When the SNRs are not favourable, the accuracy lowers down a bit, and this is a first hint on the fact that these fingerprints are accurate, but we have to investigate about robustness.

The method requiring Spleeter shows a performance decrease, even if it is still tolerable: the accuracy is overall very good; it lowers down only for the least favourable SNR and just in some scenarios. These new tests confirm the first finding that was obtained with the Energy-Difference fingerprint: source separation, at least when performed by means of the Spleeter network, is reducing the accuracy, instead of being helpful.

English Tests with Noise				English Tests with Noise			
SNR (dB)	EER (%)	τ	Acc_{mult} (%)	SNR (dB)	EER (%)	τ	Acc_{mult} (%)
5	0.21	0.90	99.76	5	1.33	1.05	99.22
0	0.57	0.97	99.35	0	1.65	1.06	98.65
-5	1.40	1.05	98.01	-5	2.74	1.10	96.24
-10	3.81	1.14	92.57	-10	4.65	1.15	91.41

English Tests with Music				English Tests with Music			
SNR (dB)	EER (%)	τ	Acc_{mult} (%)	SNR (dB)	EER (%)	τ	Acc_{mult} (%)
5	0.08	0.86	99.68	5	1.78	1.07	97.95
0	0.21	0.90	99.37	0	2.62	1.10	96.41
-5	0.63	0.98	98.90	-5	4.33	1.14	92.29
-10	1.88	1.07	96.08	-10	9.76	1.21	79.10

(a) Not requiring the use of Spleeter (b) Requiring the use of Spleeter

Table 4.7: Results of methodologies requiring the DNN fingerprint (English): all queries are present in the database

Spanish Tests with Noise				Spanish Tests with Noise			
SNR (dB)	EER (%)	τ	Acc_{mult} (%)	SNR (dB)	EER (%)	τ	Acc_{mult} (%)
5	0.13	0.91	99.93	5	0.29	0.96	99.83
0	0.53	1.00	99.17	0	1.06	1.05	98.14
-5	2.14	1.09	95.32	-5	3.26	1.12	94.37
-10	5.87	1.17	86.28	-10	7.12	1.19	87.67

Spanish Tests with Music				Spanish Tests with Music			
SNR (dB)	EER (%)	τ	Acc_{mult} (%)	SNR (dB)	EER (%)	τ	Acc_{mult} (%)
5	0.03	0.81	100.00	5	0.48	0.99	99.44
0	0.11	0.89	99.90	0	0.11	0.89	99.90
-5	0.57	1.00	98.67	-5	5.42	1.17	87.37
-10	2.28	1.10	92.05	-10	13.24	1.24	71.99

(a) Not requiring the use of Spleeter (b) Requiring the use of Spleeter

Table 4.8: Results of methodologies requiring the DNN fingerprint (Spanish): all queries are present in the database

Tests conducted where approximately only 45% of the query has a match on the database

Since the network was again trained on a part of these audio files, this methodology is very effective also when discriminating between a fingerprint which is already stored in the database and a fingerprint which is not: the EER is really low, and also the threshold determination seems very effective, considering the really high accuracy percentages.

The methodology requiring the use of Spleeter has a bit lower performances. Still, especially for favourable SNRs, performances are good.

English Tests with Noise			
SNR (dB)	EER (%)	τ	Acc (%)
5	0.00	0.63	99.99
0	0.00	0.63	99.99
-5	0.08	0.88	95.60
-10	1.00	1.07	72.51

English Tests with Music			
SNR (dB)	EER (%)	τ	Acc (%)
5	0.00	0.66	99.99
0	0.00	0.61	99.99
-5	0.00	0.57	99.99
-10	0.00	0.63	99.97

(a) Not requiring the use of Spleeter

English Tests with Noise			
SNR (dB)	EER (%)	τ	Acc (%)
5	0.02	0.84	98.84
0	0.08	0.88	96.58
-5	0.60	1.03	76.28
-10	2.07	1.12	59.12

English Tests with Music			
SNR (dB)	EER (%)	τ	Acc (%)
5	0.00	0.68	99.99
0	0.11	0.91	93.69
-5	1.05	1.07	68.99
-10	5.73	1.19	46.89

(b) Requiring the use of Spleeter

Table 4.9: Results of methodologies requiring the DNN fingerprint (English): only 45% of queries is present in the database

Comments

The DNN approach has more satisfactory results:

- the EER is low, showing that one can work on such an approach to obtain an increasingly accurate system;
- the threshold refers to a distribution of L2 distances as in Figure 4.2 and, in this sense, poses a fairly sensible division;
- the accuracy, though it decreases when Spleeter is used, is still enough to say that this proposal is able to recognize new fingerprints, and to store them in the database if there is the need to.

Spanish Tests with Noise			
SNR (dB)	EER (%)	τ	Acc (%)
5	0.06	0.91	95.78
0	0.08	0.93	93.86
-5	0.22	0.97	86.62
-10	2.16	1.13	52.29

Spanish Tests with Music			
SNR (dB)	EER (%)	τ	Acc (%)
5	0.01	0.68	99.99
0	0.01	0.67	99.99
-5	0.01	0.62	99.99
-10	0.05	0.85	96.28

(a) Not requiring the use of Spleeter

Spanish Tests with Noise			
SNR (dB)	EER (%)	τ	Acc (%)
5	0.37	1.02	78.82
0	0.62	1.05	71.72
-5	1.32	1.10	60.82
-10	4.87	1.19	43.76

Spanish Tests with Music			
SNR (dB)	EER (%)	τ	Acc (%)
5	0.61	1.05	71.91
0	1.29	1.09	60.53
-5	2.45	1.14	51.62
-10	7.91	1.22	38.52

(b) Requiring the use of Spleeter

Table 4.10: Results of methodologies requiring the DNN fingerprint: only 45% of queries is present in the database

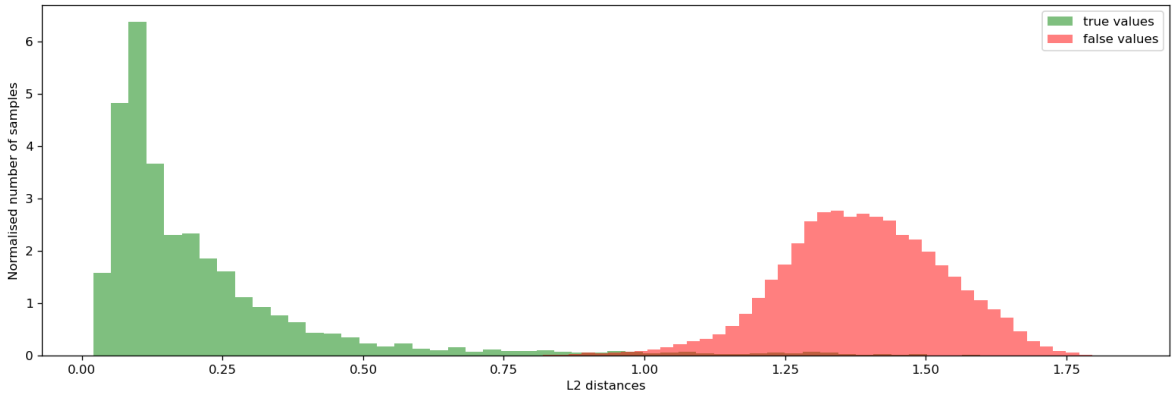


Figure 4.2: Distribution of Scores in the DNN approach for de-noised English speech with background music at $SNR = 5$

Regarding the difference between noise and background music, there is a slight decrease in performances, with EER rising up and accuracy lowering down. This means that there is room for improvement toward the robustness property, even though we are quite satisfied with the fact that, at least with the most favourable SNRs, there are some satisfactory metrics. It appears that, despite the slight decrease in performances for the least favourable SNRs, the network has been trained successfully in recognizing speech and speech only.

4.4. Conclusions

In Figure 4.3 and Figure 4.4 we compare EER and accuracy of the proposed systems. We leave out the decisional threshold, since as said in Section 4.2.1 this metric depends on the conditions of the system itself, so it is useful more for the system in itself rather that for cross-checking.

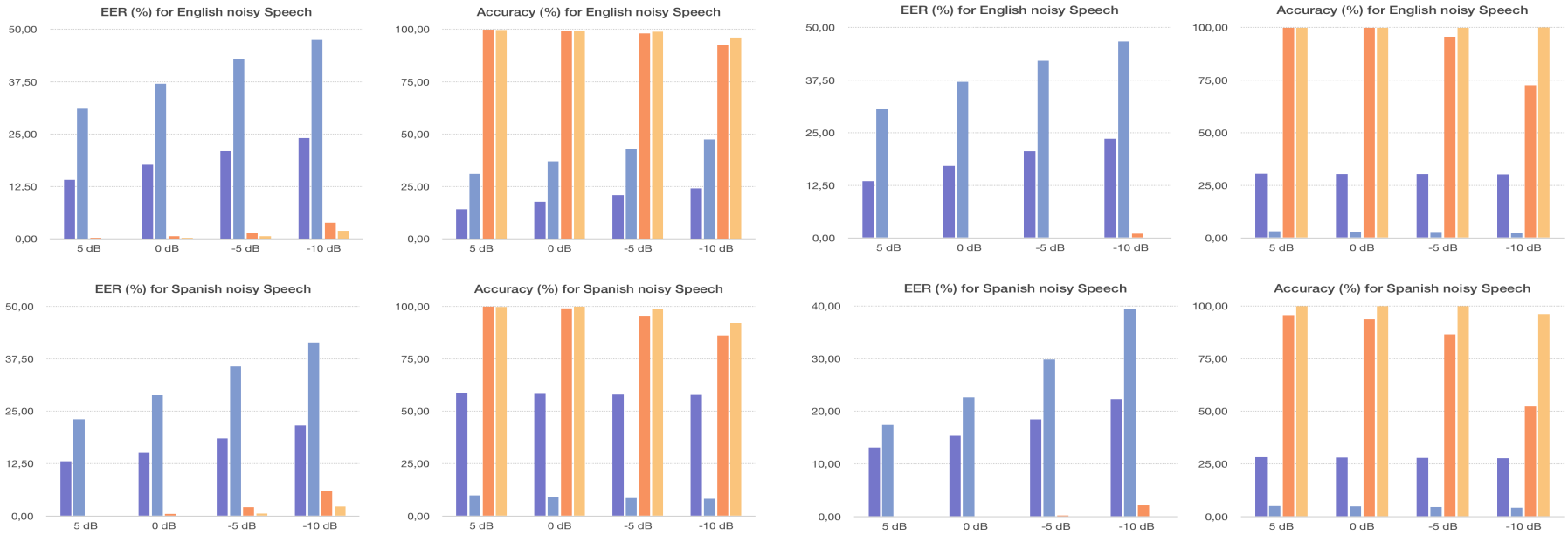
It is clear that EER is globally lower with the DNN approach, meaning that the new proposal is more effective from a discrimination point of view. Also the accuracy shows great improvements, so this means the DNN approach could be a good way for extracting more significant speech fingerprints, since the accuracy property itself is satisfied.

Observing the different kind of noises, we have to admit that DNN approach has some fluctuating values for the least favourable SNRs, so the robustness property can still be improved.

Despite this slightly weak point, the accuracy values stay globally better, especially for the SNRs that are favourable for speech (5 dB, 0 dB).

In a few words, by introducing a carefully designed DNN in the picture, and picking Triplet Loss for training the embedding extraction stage so that we could compare them by means of a L2 normalisation, we managed to outperform the baseline, i.e. the energy-difference fingerprint.

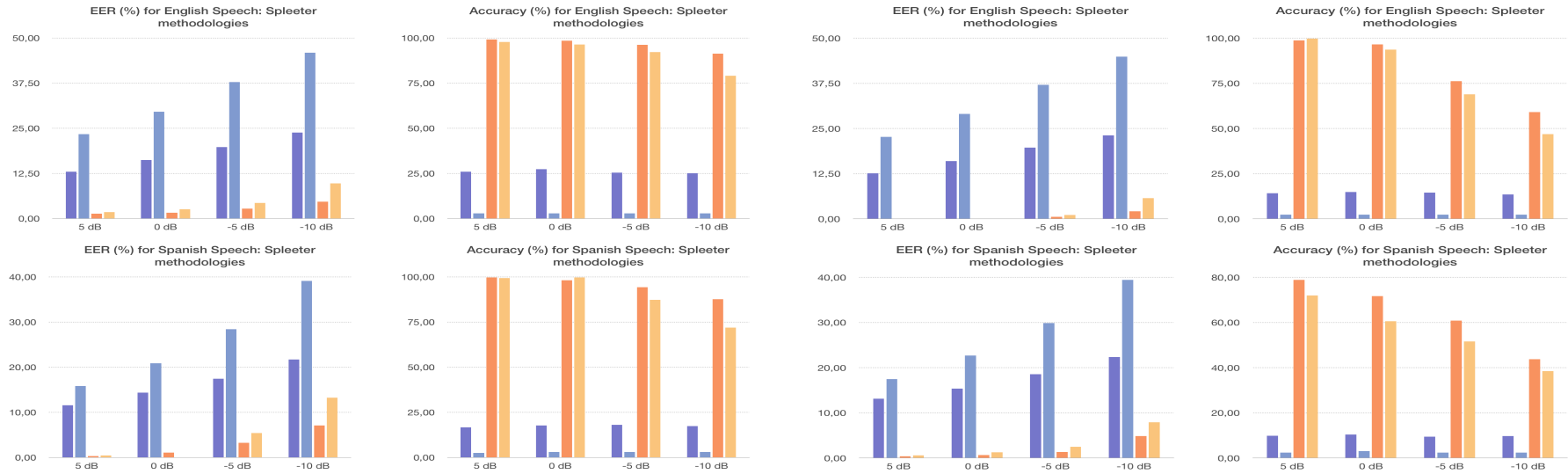
As aforementioned, some room for improvement remains in terms of robustness, especially to very loud background music, but nevertheless the results are promising, in comparison with the baseline.



(a) All queries with a match in the database

(b) Approximately 45% of queries with a match in the database

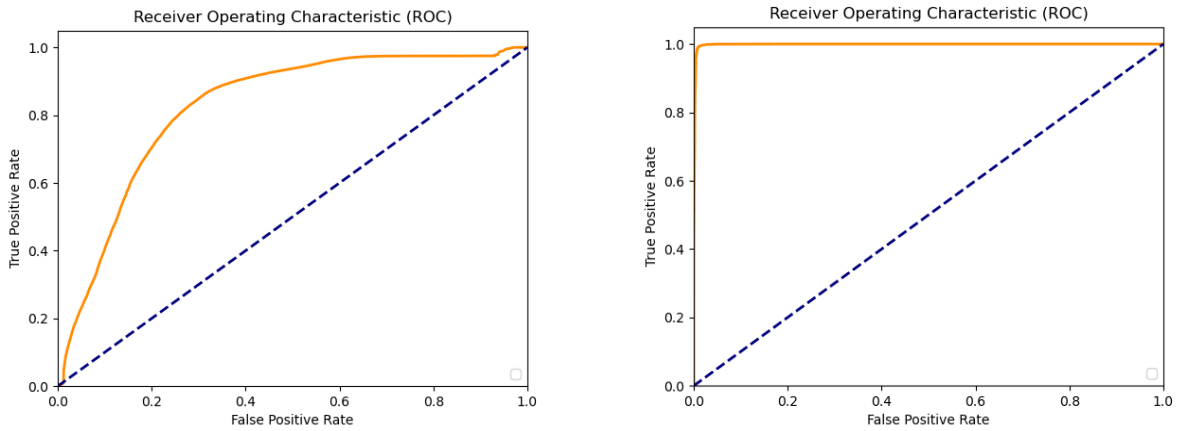
Figure 4.3: Comparison of metrics for the methods not involving the use of Spleeter: purplish colours referred to the the experiments with the energy-difference fingerprint, warm colours referred to the DNN fingerprint; darker colours referred to the scenarios with noise, lighter ones referred to scenarios with background music. The lower EER the better, the higher accuracy the better.



(a) All queries with a match in the database

(b) Approximately 45% of queries with a match in the database

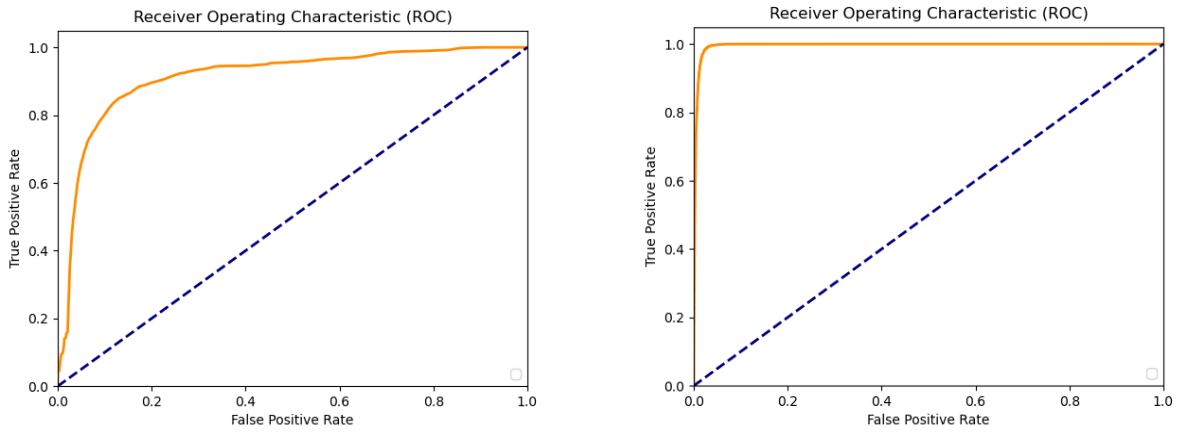
Figure 4.4: Comparison of metrics for the methods involving the use of Spleeter: purplish colours referred to the the experiments with the energy-difference fingerprint, warm colours referred to the DNN fingerprint; darker colours referred to the scenarios with noise, lighter ones referred to scenarios with background music. As already said, the lower EER the better, the higher accuracy the better.



(a) ROC curve for the Energy-Difference fingerprint

(b) ROC curve for the DNN's proposal

Figure 4.5: Comparison between different ROCs: tested on English speech files, made noisy with background music at $SNR = 5dB$ and then de-noised with Spleeter. It is noticeable that the first approach does not perform a discrimination as sharp as the one the second approach does.



(a) ROC curve for the Energy-Difference fingerprint

(b) ROC curve for the DNN's proposal

Figure 4.6: Comparison between different ROCs: tested on Spanish speech files, made noisy with every-day-life noises at $SNR = 0$ dB and then de-noised with Spleeter. In this case, the energy-difference fingerprint has a ROC curve which is slightly better than the one in Figure 4.5a; however, the second one stays way more precise.

5 | Conclusions and Outlook

In this thesis, the problem of speech fingerprinting and matching was explored.

Investigating the state of the art of more established music fingerprinting technology, we analyzed attempts at speech fingerprinting and the applications in which they were used.

Taking into account the progresses in speaker recognition, word recognition and speech enhancement, the idea was to design a fingerprint that would work more similarly to music fingerprints: we decided at first to design a proposal referring to Haitsma and Kalker's work, and then to implement some concepts of deep learning, starting from the intuition not only to consider an audio file as a two-dimensional representation, like an image, but to consider the process of ID assignment similar to some "face recognition" tasks. From here comes the idea to implement, for the extraction phase, a CNN trained to optimize a triplet loss.

While the first proposal definitely does not work for speech fingerprinting, the second proposal shows significant improvements, reaching satisfactory results.

We believe that this thesis can lead the way to future developments, with more means to develop an increasingly accurate, precise and robust architecture.

In particular, we believe that some improvements can be done:

- on the variety of ways in which a single audio file can be fed to the DNN for the training phase, for example with a more varied number of noises;
- on background music, looking at how other musical genres can influence performances, especially contemporary and popular music;
- on audio-preprocessing, not so much according to classical techniques but using speech enhancement technologies, which have achieved satisfactory results in recent years;
- on a wider variety of languages, hopefully the ones better representing all the language strains, so that the network can be effective on nearly the whole amount of

languages and can cover potentially the majority of people.

Acronyms

API Application Programming Interface.

BER Bit Error Rate.

CNN Convolutional Neural Network.

DCT Discrete Cosine Transform.

DET Detection Error Tradeoff.

DFT Discrete Fourer Transform.

DNN Deep Neural Network.

EER Equal Error Rate.

EPN Exponential Pseudo Norm.

FCL Fully-Connected Layer.

FFT Fast Fourier Transform.

FN False Negative.

FNR False Negative Rate.

FP False Positive.

FPR False Positive Rate.

GAN Generative Adversarial Network.

GMM Gaussian Mixture Model.

HT Haar Transform.

KLT Karhunen-Loève Transform.

L2 Euclidean Distance.

Leaky-ReLU Leaky Rectified Linear Unit.

LPCM Linear Pulse Code Modulation.

MFCC Mel Frequency Cepstral Coefficients.

MLP Multi-Layer Perceptron.

NDCR Normalised Detection Cost Rate.

NNBF Nearest-Neighbor Based Fingerprint.

NSSC Normalized Spectral Subband Centroid.

PCA Principal Component Analysis.

PESQ Perceptual Evaluation of Speech Quality.

PSD Power Spectral Density.

ReLU Rectified Linear Unit.

RMS Root Mean Square value.

ROC Receiver Operating Characteristic.

SDR Source to Distortion Ratio.

SF Scale Factor.

SFM Spectral Flatness Measure.

SNR Signal-to-Noise Ratio.

SR Sample Rate.

STFT Short Time Fourier Transform.

STOI Short-Time Objective Intelligibility.

SVD Singular Value Decomposition.

t-SNE t-distributed Stochastic Neighbor Embedding.

TN True Negative.

TP True Positive.

TPR True Positive Rate.

TRM TRM Recognises Music.

WHT Walsh-Hadamard Transform.

ZCR Zero Crossing Rate.

List of Figures

List of Tables

Bibliography

[1] *Shazam WebSite*. https://www.shazam.com/company.

[2] *An Introduction to Forensics Audio*. https://www.soundonsound.com/techniques/introduction-forensic-audio.

[3] Robert C. Maher. "Audio forensic examination". In: *IEEE Signal Processing Magazine* 26.2 (2009), pp. 84–94. DOI: 10.1109/MSP.2008.931080.

[4] Pedro Cano, Eloi Batlle, Ton Kalker, and Jaap Haitsma. "A Review of Audio Fingerprinting". In: *Journal of VLSI signal processing systems for signal, image and video technology* 41.3 (2005), pp. 271–284.

[5] Mehmet Kivanç Mihçak and Ramarathnam Venkatesan. "A Perceptual Audio Hashing Algorithm: A Tool for Robust Audio Identification and Information Hiding". In: *Information Hiding*. 2001.

[6] Eric Allamanche. "Content-based Identification of Audio Material Using MPEG-7 Low Level Description". In: *ISMIR*. 2001. URL: http://dblp.uni-trier.de/db/conf/ismir/ismir2001.html#Allamanche01.

[7] Jaap Haitsma and Ton Kalker. "A highly robust audio fingerprinting system". In: *ISMIR International Conference on Music Information Retrieval*. Paris, France, 2002, pp. 107–115.

[8] Pedro Cano, Eloi Batlle, Harald Mayer, and Helmut Neuschmied. "Robust Sound Modeling for Song Detection in Broadcast Audio". In: 2002.

[9] J.G. Lourens. "Detection and logging advertisements using its sound". In: *IEEE Transactions on Broadcasting* 36.3 (1990), pp. 231–233. DOI: 10.1109/11.59850.

[10] M.J. Carey, E.S. Parris, and H. Lloyd-Thomas. "A comparison of features for speech, music discrimination". In: *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258)*. Vol. 1. 1999, 149–152 vol.1. DOI: 10.1109/ICASSP.1999.758084.

[11] Li Chai, Jun Du, Qing-Feng Liu, and Chin-Hui Lee. "A Cross-Entropy-Guided Measure (CEGM) for Assessing Speech Recognition Performance and Optimizing DNN-Based Speech Enhancement". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), pp. 106–117. DOI: 10.1109/TASLP.2020.3036783.

[12] Khaled Matrouk, Abdullah Alhasanat, Haitham ary, Ziad Alqadi, and Hasan AL-Shalabi. "Speech Fingerprint to Identify Isolated Word-Person". In: *World Applied Sciences Journal* 31 (Jan. 2014), pp. 1767–1771. DOI: 10.5829/idosi.wasj.2014.31.10.468.

[13] Farnood Faraji, Yazid Attabi, Benoît Champagne, and Weiping Zhu. "On the Use of Audio Fingerprinting Features for Speech Enhancement with Generative Adversarial Network". In: *2020 IEEE Workshop on Signal Processing Systems (SiPS)* (2020), pp. 1–6.

[14] David Martín-Gutiérrez, Gustavo Hernández-Peñaloza, Jose Manuel Menéndez, and Federico Álvarez. "An End-to-End Speaker Diarization Service for improving Multimedia Content Access". In: *NEM Summit*. 2020, pp. 1–4.

[15] Florian Schroff, Dmitry Kalenichenko, and James Philbin. "FaceNet: A Unified Embedding for Face Recognition and Clustering". In: *CoRR* abs/1503.03832 (2015). arXiv: 1503.03832. URL: http://arxiv.org/abs/1503.03832.

[16] Alexander Hermans, Lucas Beyer, and Bastian Leibe. "In Defense of the Triplet Loss for Person Re-Identification". In: *CoRR* abs/1703.07737 (2017). arXiv: 1703.07737. URL: http://arxiv.org/abs/1703.07737.

[17] Romain Hennequin, Anis Khlif, Felix Voituret, and Manuel Moussallam. "Spleeter: a fast and efficient music source separation tool with pre-trained models". In: *Journal of Open Source Software* 5.50 (2020). Deezer Research, p. 2154. DOI: 10.21105/joss.02154. URL: https://doi.org/10.21105/joss.02154.

[18] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition, Fourth Edition*. Academic Press, 2009. ISBN: 9781597492720.

[19] F. Kurth, Michael Clausen, and A. Ribbrock. "Identification of Highly Distorted Audio Material for Querying Large Scale Data Bases". In: (Jan. 2002).

[20] S.R. Subramanya, Rahul Simha, B. Narahari, and Abdou Youssef. "Transform–Based Indexing of Audio Data for Multimedia Databases S.R. Subramanya Rahul Simha B. Narahari Abdou Youssef". In: (Nov. 1998).

[21] G. Richly, L. Varga, F. Kovacs, and G. Hosszu. "Short-term sound stream characterization for reliable, real-time occurrence monitoring of given sound-prints". In: *2000 10th Mediterranean Electrotechnical Conference. Information Technology and Electrotechnology for the Mediterranean Countries. Proceedings. MeleCon 2000 (Cat. No.00CH37099)*. Vol. 2. 2000, 526–528 vol.2. DOI: 10.1109/MELCON.2000.879986.

[22] Constantin Papaodysseus, George Roussopoulos, Dimitrios K. Fragoulis, Athanasios D. Panagopoulos, and Constantin Alexiou. "A New Approach to the Automatic Recognition of Musical Recordings". In: *Journal of The Audio Engineering Society* 49 (2001), pp. 23–35.

[23] Somsak Sukittanon and Les E. Atlas. "Modulation frequency features for audio fingerprinting". In: *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*. Vol. 2. 2002, pp. II-1773-II–1776. DOI: 10.1109/ICASSP.2002.5744966.

[24] Christopher J. C. Burges, John C. Platt, and Soumya Jana. "Extracting noise-robust features from audio data". In: *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*. Vol. 1. 2002, pp. I-1021-I–1024. DOI: 10.1109/ICASSP.2002.5743968.

[25] Eloi Batlle, Jaume Masip, and Enric Guaus. *Automatic Song Identification in Noisy Broadcast Audio*. 2002.

[26] *Etantrum*. https://sourceforge.net/projects/freetantrum/.

[27] *Musicbrainz*. https://musicbrainz.org.

[28] A. Kimura, K. Kashino, T. Kurozumi, and H. Murase. "Very quick audio searching: introducing global pruning to the Time-Series Active Search". In: *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*. Vol. 3. 2001, 1429–1432 vol.3. DOI: 10.1109/ICASSP.2001.941198.

[29] Christos Faloutsos, M. Ranganathan, and Yannis Manolopoulos. "Fast Subsequence Matching in Time-Series Databases". In: *ACM SIGMOD Record* 23 (June 2000). DOI: 10.1145/191839.191925.

[30] M.L. Miller, M.A. Rodriguez, and I.J. Cox. "Audio fingerprinting: nearest neighbor search in high dimensional binary spaces". In: *2002 IEEE Workshop on Multimedia Signal Processing*. 2002, pp. 182–185. DOI: 10.1109/MMSP.2002.1203277.

[31] Avery Wang and Julius Smith. "System And Methods For Recognizing Sound And Music Signals In High Noise And Distortion". In: *Journal of The Acoustical Society of America - J ACOUST SOC AMER* 121 (Jan. 2007). DOI: 10.1121/1.2723991.

[32] Maguelonne Héritier, Vishwa Gupta, Langis Gagnon, Gilles Boulianne, and Patrick Cardinal Samuel Foucher. "CRIM's Content-based Copy Detection System for TRECVID". In: *NIST TREC Video Retrieval Evaluation (TRECVID) Conference*. Gaithersburg, MD, USA, 2009.

[33] Aonan Zhang, Quan Wang, Zhenyao Zhu, John William Paisley, and Chong Wang. "Fully Supervised Speaker Diarization". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2019), pp. 6301–6305.

[34] Eva Sharma, Guoli Ye, Wenning Wei, Rui Zhao, Yao Tian, Jian Wu, Lei He, Ed Lin, and Yifan Gong. "Adaptation of RNN Transducer with Text-To-Speech Technology for Keyword Spotting". In: *ICASSP 2020 - 2020 IEEE International Conference*

on Acoustics, Speech and Signal Processing (ICASSP). 2020, pp. 7484–7488. DOI: 10.1109/ICASSP40776.2020.9053191.

[35] Craig Macartney and Tillman Weyde. "Improved Speech Enhancement with the Wave-U-Net". In: *ArXiv* abs/1811.11307 (2018).

[36] Avery Wang. "An Industrial Strength Audio Search Algorithm". In: *ISMIR International Conference on Music Information Retrieval*. Washington, D.C., USA, 2003.

[37] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[38] *Anh H. Reynolds Quantum Chemist & Data Scientist Blog*. https://anhreynolds.com/blogs/cnn.html.

[39] Tom M. Mitchell. *Machine Learning*. New York: McGraw-Hill, 1997. ISBN: 978-0-07-042807-2.

[40] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009. ISBN: 1441412697.

[41] ISO. *ISO/IEC 14882:2011 Information technology — Programming languages — C++*. Geneva, Switzerland: International Organization for Standardization, Feb. 2012, 1338 (est.) URL: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=50372.

[42] Voxforge.org. *Free Speech... Recognition (Linux, Windows and Mac) - voxforge.org*. http://www.voxforge.org/.

[43] Jyh-Min Cheng and Hsiao-Chuan Wang. "A method of estimating the equal error rate for automatic speaker verification". In: *2004 International Symposium on Chinese Spoken Language Processing*. 2004, pp. 285–288. DOI: 10.1109/CHINSL.2004.1409642.

[44] Aniello Castiglione, Michele Nappi, and Stefano Ricciardi. "Trustworthy Method for Person Identification in IIoT Environments by Means of Facial Dynamics". In: *IEEE Transactions on Industrial Informatics* PP (Apr. 2020), pp. 1–1. DOI: 10.1109/TII.2020.2977774.

[45] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

Acknowledgements

First of all, I would like to thank all those who have materially contributed to the realisation of this thesis. First and foremost, Professor Paolo Bestagini for having made the idea of conducting the thesis abroad possible.

I would like to thank Patrick, for having agreed to welcome me into the Fraunhofer IDMT family during these months.

I would like to thank Milica, for setting the tone in the early stages of this project, and Artem for being present and giving wise advice over the last few months.

A special thanks to Luca, for his patience, his availability, and because certainly this experience would not have been the same without his contribution.

I would like to thank my joking and supportive parents, who thought it was a good idea to have an heir in the Colzani family. I still don't know if that was the right choice.

I would like to thank all the people with whom I shared my experiences in Germany and who made these months light and fun: from the guys at we4you, to my roommate Anna; from Mahsa to the guys from Delhi and the Italian guys, who showed me that I can stand up to people ten years older than me.

Special thanks to Aviral, the real soul of all the parties in Ilmenau; to Francesco M. and Alessandro, with their pizzas and their veracious attitude; to Francesco B. for being the trash queen confidante I needed. Thanks to Aakash, one of the most rational and witty people I know.

Thanks, of all people, to Aditi, the one who managed to make the experience in a foreign land lighter and with whom I was able to let go of a deeper, meaningful and, I presume, lasting friendship.

I thank Anna, Axel and Matteo S., for the young friendship that has lasted despite the fact that I had to, after only a few months of acquaintance, spend my life elsewhere.

I would like to thank Donzo, for his patience and tenderness, because crying over each

other and telling each other about our discomforts has been vital to keep going in these months.

I thank BEST Milan, responsible in these years for some memorable memories and a certain growth from a personal point of view. And above all, some of the smartest women I've ever met in my life, with whom I had the honour of working on the board and who had to put up with me as secretary: to Vale, Ilaria, and Yara, without you there would have been a certain lack of drive to mature.

I would like to thank the guys with whom I spent my first months at the Politecnico, and who have become my travelling companions for the last five years, even though I took a different path from them: from Marta to Mattia, from Eleonora with her advice in the last month to Davide, from Alessio with his wisdom to Luca with his memes.

A heartfelt thank you to Falcon, who has been a constant companion of fits of rage, advice and swearing in the last moments of our work.

Special thanks to Dario, meme man and sex symbol that Monza and Brianza do not deserve, but certainly need.

I thank Lea, for always being there from a distance and for proving that our friendship, although telematic, is solid and valid over time.

I thank Giorgio, for sharing my hatred for others and for his comic vision that has brought a smile to my face on the most sleepless nights.

I thank Edoardo, with his irony and his aesthetic and musical tastes, as a reminder that I am not the only one who does a bit of what she likes.

I thank Matteo M., with his culture, his wit, his love for music and his beautiful speech abilities: if I have not lost the Italian language definitively, it is certainly thanks to him.

I thank Greta, because with her intelligence and patience, learning German has become a possible mission again, regardless of how much I need to get certain words into my head.

And, most of all, I thank the woman who has put up with me and loved me the most in these years, sharing good times and never abandoning me, nor letting herself be abandoned, in the darkest periods. The person who has always been there even though she has, and allows me to have, full independence, the one I grew up with. The heir, in a way, of my dear grandmother.

Benny, if you hadn't been there, none of this would have meant the same thing.