Università degli Studi di Salerno

Dipartimento di Ingegneria dell'Informazione ed Elettrica e Matematica Applicata

Dottorato di Ricerca in Ingegneria dell'Informazione
Ciclo 35

TESI DI DOTTORATO / PH.D. THESIS

Social Robots, from intelligent perception to emphatic behaviors

ANTONIO ROBERTO

SUPERVISOR: **PROF. MARIO VENTO**
PROF. LUC BRUN

PHD PROGRAM DIRECTOR: **PROF. PASQUALE CHIACCHIO**

Anno 2023

Abstract

Social robots are a particular category of robots able to perceive information about the environment to reason about the acquired information with the particular aim to interact with humans. Remarkable applications of social robots include museum guides, nurses, autism treatment, hotel assistants, and elderly care. Studies prove that the capability of social robots to personalize the conversation and perceive the interlocutor's emotions are the key behaviours that allow them to be considered emphatic.

To these purposes, social robots rely on multiple sensory modalities to acquire information about their interlocutor robustly and accurately. Their sensorial equipment is crucial considering that this kind of robot commonly works in challenging environments e.g., dynamic lighting conditions and loud environmental noise. In addition to these challenges, social robots must converse in real-time with humans to give them the feeling of natural interaction. Considering the application context, this result is only possible if the computation is performed on board of the social robot platform, which makes the task harder due to the implicit computational and power constraints.

This thesis tackles these requirements in the context of Deep Learning. In particular, a novel software architecture optimized for multimodal real-time interactions has been proposed as a general-purpose solution for social robots. The realization of a robotic prototype al-

lowed to identify the main issues perceived by humans about state-of-the-art algorithms related to human-robot interaction when deployed together in a real application. In light of this result, this thesis advances the state-of-the-art by proposing and validating novel auditory and natural language understanding algorithms optimized to be executed on robotic embedded systems while keeping high accuracy.

The proposed social robot architecture includes all the software modules that allow to meet the main requirements of a social robot: first, a dialogue manager able to personalize the human-robot interaction by exploiting the biometrics perceived by the sensors of the social robot; second, a multimodal sensor aggregation module able to exploits the information acquired by different types of sensors to increase the robustness to environmental noise; finally, parallel processing pipelines that, properly designed and implemented, ensure real-time performance. A social robot prototype based on the proposed architecture has been realized and deployed in the SICUREZZA exhibition for three days. 161 people who interacted with the robot evaluated their experience by answering 5 questions with a score between 1 and 5. The maximum score was achieved for more than 40% of the answers and the average rate was between 4 and 5. This result acquires more relevance considering that the people who attended the conference were technically skilled and, therefore, their feedback is reliable. The survey also allowed to investigate the feeling of humans about the performance of the state-of-art algorithms available on the proposed prototype. This analysis results in the need for audio algorithms more robust to environmental noise and more efficient human utterances processing pipelines.

Starting from this evidence, this thesis initially proposes two learnable audio representations to achieve the following goals: robustness to environmental noise and computational efficiency.

Firstly, a novel convolutional layer, namely Denoising-Enhancement Layer (DELayer), has been proposed. It is able to denoise and enhance

the input signal by combining an attention module (DELayer) with the SincNet layer; the final representation is thus able to attenuate the frequency components affected by environmental noise and amplify the ones relevant for the classification. The experimental results demonstrate that the end-to-end model based on the proposed DELayer, namely DENet, is definitely more effective than the existing methods in detecting and recognizing audio events of interest on the MIVIA Audio Events and on the MIVIA Road Events benchmarks. The novel CNN achieves state-of-the-art performance on both datasets and further analyses show its robustness to noise, its stability among different environmental conditions and its generalization capabilities.

The latency evaluation of the DENet model pointed out the need for a more optimized audio model to be run on CPU devices without batching the predictions and, therefore, increasing the speed of the model itself. For this reason, in this thesis the DELayer fundamentals have been extended to DEGram, a learnable spectrogram-based representation. Being a spectrogram-like representation, DEGram allows the use of shallower CNN with comparable accuracy. Consequently, it reduces the computational requirements of the final system while keeping the robustness of DENet. DEGram is the result of the combination of two novel layers: SincGram and a Time-Frequency version of the DELayer, namely TF-DELayer. The former, like SincNet, is able to learn the frequencies of interest for the audio analysis problem we are dealing with, extracting task-specific features through trainable band-pass filters. The latter can denoise the input signal from environmental background noise using a time-frequency attention mechanism, focusing on the part of the input signal in which the sound of interest is temporally located and on the frequency components not affected by noise. Differently from the previous DELayer, the TF-DELayer uses a squeeze and excitation attention mechanism to drastically reduce the attention overhead. The experimental analysis demonstrated the effectiveness of DEGramNet, an audio CNN based on the proposed

DEGram. DEGramNet was able to achieve state-of-the-art results on the VGGSound dataset (Sound Event Classification) and comparable results with a complex Network Architecture Search approach on the VoxCeleb1 dataset (Speaker Identification), proving to be a general-purpose architecture for audio analysis tasks. Moreover, the ablation study proved that the proposed DEGram representation was more effective than Spectrogram, being able to improve the performance of the same CNN architecture trained with both the representations and to achieve better accuracy with respect to deeper models. Finally, the reduction of the number of features computed by DEGram allows the same network to reduce the inference time on CPU architectures, making the model work in real-time on embedded systems.

To address the user-perceived latency issue, in this thesis three multitask neural network models have been proposed to perform the following tasks simultaneously: Gender Recognition, Emotion Recognition, Age Estimation, and Speaker Re-Identification. I proved that it is possible not only to reduce the computational requirements w.r.t. the single-task counterparts but also to obtain a better generalization of the model itself. The experiment allowed to identify the best architecture to use in very limited setups and when higher accuracy is required. In particular, each multitask architecture is characterized by a different trade-off between feature sharing and model parameters. All the architecture have in common the CNN backbone, i.e. ResNet18, and the audio representation, i.e. DEGram. Together with the multitask models, this thesis also proposes a training loss that allows to take into account different datasets even if they don't have the labels for all the tasks. Moreover, the GradNorm algorithm has been used to avoid unbalancing in the performance of the different tasks. The proposed architectures have been compared with state-of-the-art algorithms and single-task counterparts on standard benchmarks for the considered tasks, i.e. VoxCeleb1 and VoxCeleb2 (Speaker Re-Identification and Age Estimation), Mozilla Common Voice (Gender Recognition) and

IEMOCAP (Emotion Recognition). The trained models outperformed state-of-art models on three tasks over four proving their effectiveness. Moreover, the multitask approach proved to improve the performance of Gender Recognition, Emotion Recognition, and Age Estimation with respect to the single-task models, validating the hypothesis that the multitask paradigm not only reduces the computational requirements but also improves the generalization capabilities of the model.

Finally, this thesis further reduces the response latency of the social robot by identifying the best transformer architecture for Natural Language Understanding in the context of Social Robotics. In particular, the trade-off between the accuracy and processing time of different multitask transformer architectures has been evaluated. The evaluation takes into account the problems related to the unavailability of data for training the model with a good degree of generalization and the limited computational and memory resources of GPU-enabled embedded devices. The considered transformer architectures are characterized by different optimization approaches like knowledge distillation and grouped convolutions. On one hand, the results proved that using transformer models it is possible to achieve good performance even using a fine-tuning approach (performance indices over 90%). It has been also demonstrated that the results are highly influenced by the task (i.e., imbalance in the sentences, vocabulary size, and the number of entities). To address this issue, in the thesis are identified some design strategies allowing to better choice the transformer to use, so that it is possible to avoid wasting weeks training them all. The processing times of these transformers have been evaluated over an embedded system commonly used for robotic applications, namely the NVIDIA Jetson Xavier NX. The analysis allowed to reduce the impact of the NLU model on the average response time by around 20-25%.

Overall, this thesis explores Social Robot architectures able to be emphatic with humans through the expression of personalization capabilities. The proposed prototype also allowed to identify the main

gaps in state-of-art sensor processing algorithms as they are perceived by humans. This thesis also addresses the identified issues by proposing novel audio representations very robust to environmental noise and capable to gain the performance of shallow neural networks commonly running on Social Robots with a negligible computational cost. In addition, the optimization of the computational requirements for the audio analysis and NLU components of the social robots have been also addressed by proposing two multitask models able to reduce the computation latency by 75% and 33% with respect to earlier models, respectively.

Contents

1 Introduction **3**
1.1 Application context 4
1.2 Motivation and thesis overview 10
1.3 State of the art 11
1.3.1 Conversational artificial intelligence 11
1.3.2 Audio perception in social robotics 23
1.3.3 Contributions of this thesis 35

2 Social robots: from design to human feedback evaluation **37**
2.1 Background 38
2.2 A general-purpose social robot architecture 39
2.2.1 Sensors and actuators 41
2.2.2 Sensor data analysis 42
2.2.3 Multimodal sensors aggregation 45
2.2.4 Multimodal re-identification 49
2.2.5 Behaviour manager 50
2.2.6 Dialogue manager 51
2.2.7 I/O services manager 52
2.3 Architecture implementation 53
2.4 Human validation 63
2.4.1 Survey description 64

2.4.2 Results 65

3 Robust and lightweight audio representation learning 69
3.1 Background 70
3.2 Noise robustness in end-to-end convolutional networks . 72
3.2.1 Methodology 72
3.2.2 Experimental framework 83
3.2.3 Experimental results 87
3.3 Lightweight audio convolutional networks 95
3.3.1 Methodology 95
3.3.2 Experimental framework 105
3.3.3 Experimental results 110

4 Multitask audio neural networks for social robots 119
4.1 Background 120
4.2 Methodology 121
4.2.1 Multitask network architectures 121
4.2.2 Convolutional neural network backbone 125
4.2.3 Backbone adaptation to multitask learning . . . 126
4.2.4 Multitask network loss 126
4.3 Experimental framework 128
4.3.1 Datasets 129
4.3.2 Evaluation metrics 131
4.3.3 Implementation details 131
4.4 Experimental results 134
4.4.1 Processing time evaluation 136

5 Efficient natural language understanding for social robots 141
5.1 Background 142
5.2 Methodology 143
5.2.1 BERT 143
5.2.2 Distil-BERT 144

5.2.3 ALBERT 144
5.2.4 SqueezeBERT 145
5.3 Experimental framework 146
5.3.1 Datasets 146
5.3.2 Implementation details 146
5.4 Experimental results 148
5.5 NLU impact evaluation 150

6 Conclusions **153**
6.1 Outlook 159

Bibliography **161**

We always overestimate the change that will occur in the next two years and underestimate the change that will occur in the next ten. Don't let yourself be lulled into inaction.

- Bill Gates -

Chapter 1

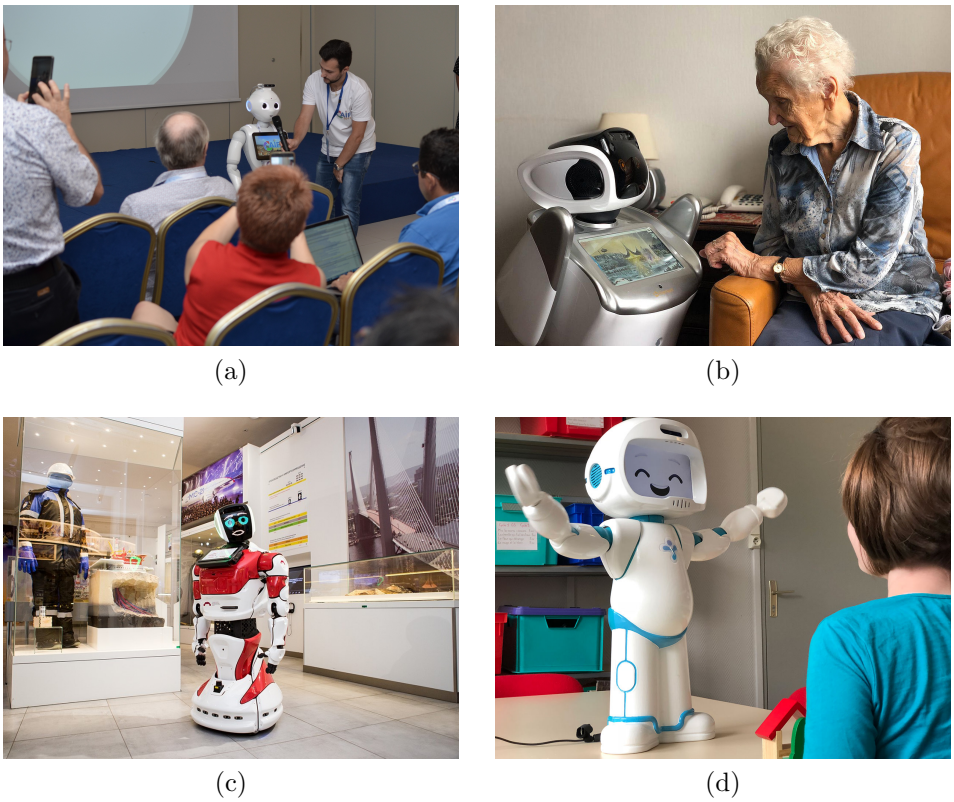
Introduction

1.1 Application context

In recent years, the disruptive growth of Artificial-Intelligence (AI) based technologies pervasively affected our daily life [1]. We are assisting to a revolution of tasks automation which, regardless of the application domain, is shifting from algorithmic control to *intelligent* autonomous planning[2]. As an example, our houses are evolving smarter every day thanks to intelligent devices and virtual assistants [3, 4]. A fundamental role in the AI revolution is played by Cognitive Robots [5]. A robot is an electromechanical system composed of sensors, actuators, a control structure, and a mechanical structure [6]. Cognitive Robots are robots able to perform tasks in dynamic contexts without human supervision, by exploiting their capability of perception, data-processing, and decision-making [7].

A cognitive robot should exhibit, apart from standard functions (navigation, motion planning, *etc.*), more semantic capabilities such as interpretation, human-robot interaction, and communication [8]. These requirements result in a particular category of cognitive robots, namely social robots, which are characterized by their ability in interacting with humans in several ways (*e.g.* spoken language and movements) while giving the interlocutor the feeling of empathy [9, 10, 11]. Popular applications of social robots include (Figure 1.1) museum guides [12], nurses [13], autism treatment [14], hotel assistants [15], and elderly care [16].

The general architecture of a social robot, as shown in Figure 1.2, is composed of two levels of functionalities [17]: first, a low-level hardware and control layer which is responsible for interacting with the environment by acquiring raw information from sensors and affecting the "world's state" through actuators; second, an *intelligent social controller* layer that is responsible for the transduction of raw sensor data to high-level information which, together with long-range knowledge, is exploited from a social controller in charge of reasoning about



(a) (b) (c) (d)

Figure 1.1: Applications of social robots for performing different tasks: greetings participants at a conference (a), taking care of elder people (b), guiding people into a museum (c), and making therapy with children (d).

goals and planning actions and/or dialogues.

To perceive the surrounding environment, social robots are usually equipped with multiple sensors (*e.g.* stereo cameras, microphones, and tactile sensors) and must analyze the working scene in both an accu-

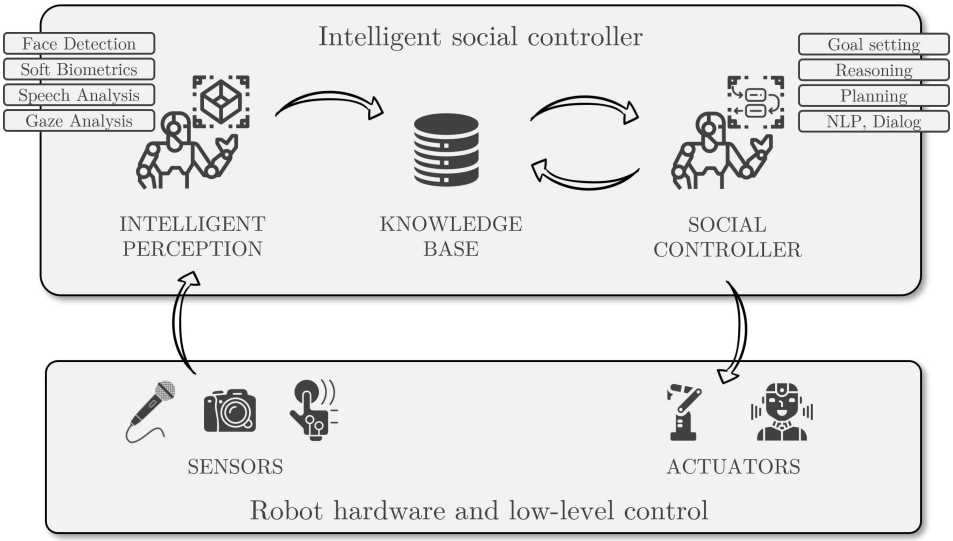


Figure 1.2: General architecture of a social robot. A limited subset of tasks that the Intelligent perception and Social controller has to deal with is also reported.

rate and robust way. Robustness is particularly important when dealing with real applications, where the perception accuracy of sensor-specific algorithms can be negatively affected due to dynamic environments. Therefore, it becomes crucial to exploit and fuse multiple modalities (*i.e.* data acquired from multiple sources) for increasing the robustness of the robot against sensor "faults" in the perception phase [18, 19]. For instance, the current position of a social robot can be acquired by either the GPS or the LIDAR and, moreover, they can be jointly used to increase the measurement precision. Furthermore, the social controller does not care about "how" that particular information has been acquired but it needs to know just its actual value.

Once a high-level representation of the scene has been captured by

(a) (b)

Figure 1.3: Comparison between conventional TOD systems (a) relying on robust text inputs and user authentication and a TOD social robots (b) able to exploit sensory data to improve the conversation with the interlocutor.

the intelligent perception module, it is used by the social controller to perform a task. In the particular case of a social robot, the main goal is to interact with humans. They commonly interact with people for chit-chatting or provide a service [20], like restaurant searches or taxi booking. In the first case, the robot has conversations aimed at maximizing the user engagement and, therefore, protracting the dialogue as much as possible; on the other hand, the latter kind of robot, also known as Task-Oriented Dialogue (TOD) systems, aims to complete a specific task in the fastest possible way (*i.e.* in the minimum number of conversational turns). Of course, due to their value for real-world business TOD systems rapidly became a hot topic in the research community [21].

Differently from conventional text-based dialogue systems, social robots allow the user to interact more naturally through the use of spoken language. Moreover, a social robot, as shown in Figure 1.3, can exploit all its sensory equipment to dialogue in an empathetic way [22], *i.e.* perceiving and expressing emotions (Affective Dialogue Systems), caring for each individual (Personalized Dialogue Systems), and casting into knowledge (Knowledgeable Dialogue Systems).

Within this context, a Social Robot is able to exploit not only the

spoken words of the interlocutor in order to recognize their emotion, but also look at their face [23] and the tone of voice [24] containing most of the sentiment-related information. It commonly happens that the emotion expressed through words is in contrast with that expressed by facial and speech expressions, which are more reliable in general. Obviously, understanding the emotional state of the interlocutor is important for a social robot not only to increase the quality of the current conversation through emphatic answers but also to get feedback about the state of the conversation itself. Indeed, emotions like boredom, frustration, enthusiasm, etc., can be used by the robot as a reward function to improve future conversations through long-life learning algorithms [25].

Regarding the ability of a Dialogue system to personalize the conversation, it commonly happens in Graphic User Interface (GUI) based systems that the user has a profile which identifies him/her without errors. The profile is used by the system to perform, for instance, a recommendation, as in the case of Amazon, Netflix, *etc.* [26, 27]. Unfortunately, in most application contexts this kind of approach is not practical, for example, in shops, cinemas, and hotels, because of two main reasons: first, the users cannot be identified without errors without additional instruments (even in that case, it is frustrating to use a different id instrument for each place); moreover, the people that frequent those places are each day different and come with very low frequency (once per life in the worst case of hotels). In these cases, social robots get into the game thanks to their ability in recognizing hard and soft biometrics [28, 29]. In fact, using the face and the voice of the interlocutor it is possible for the robot to understand if the person has already interacted with it (Re-Identification) and to group people based on categories like gender and age. Even if in the second case a profile cannot be stored, through soft-biometrics it is possible to personalize the dialogue, for example through the linguistic style [30], based on the category to which the interlocutor belongs.

Although Social Robots appear as a sort of panacea in the field of Human Machine Interaction thanks to their capabilities, they bring with them several challenges. First of all, social robots have to work "in the wild", unconstrained environments characterized by high variability and several noise sources like, for instance, changing in the lighting conditions, occlusions and, environmental noise[31]. These working conditions may cause errors in the intelligent perception module and, consequently, unwanted robot actions.

Another important requirement for a robot is surely the real-time constraint. In the particular context of social robots, the interlocutor does not want to wait before receiving an answer from the robot itself [23]. Modern robotics systems often rely on Deep Learning algorithms to perform their tasks, but they require heavy computation. A common choice is to use cloud-based services or high-performance servers to distribute the computation; however, the availability of a stable and fast internet connection is not guaranteed for this kind of application [32]. Even if it is, the streaming of audio and visual data to external systems over the network, besides being bandwidth and energy intensive, may be forbidden for privacy purposes [33] [34]. It is also important to take into account that communication over the network can be slowed down by several external factors [35]. For these reasons, the computation must be performed on board the robot, by designing a solution that is a good trade-off between accuracy and computational costs [36]. Despite the several scientific works published in recent years, a standard low-level architecture for social robots able to work in real-time and accomplish all the requirements presented above has not yet been defined.

The research community has been focusing on improving perception and reasoning algorithms for addressing these requirements. In addition to the achieved progress in the field, the companies adoption in real-life products increased, making this topic one of the hottest artificial intelligence trends. The global social robotics market was

valued at USD 1.98 billion in 2020 and is expected to reach USD 11.24 billion by 2026, registering a Compound Annual Growth Rate of 34.34% from 2021 to 2026 [1].

1.2 Motivation and thesis overview

In this thesis, I design and validate a social robot architecture that accomplishes the above-mentioned requirements thanks to a multi-modal sensor aggregation component and a novel emphatic dialogue manager module. The proposed architecture has been realized by relying on state-of-art audio-visual perception algorithms. I carried out an experiment involving real users communicating with the social robot prototype and giving their feedback about the interaction. The experiment highlighted the following gaps in state-of-the-art algorithms when deployed in social robotics applications: first, the need for increasing the robustness to environmental noise of information retrieval algorithms based on the audio modality; second, reducing the computational and memory requirements of audio and natural language understanding algorithms to allow a real-time interaction when these algorithms are executed on low-power embedded devices together with other robot's subsystems. Then, this thesis addresses the identified gaps by proposing novel deep learning algorithms able to drastically reduce the processing latency while proving high robustness to environmental noise.

The thesis is organized as follows: Section 1.3 presents an overview of the state of the art for the two main tasks dealt with in this thesis, *i.e.* conversational systems (Section 1.3.1) and audio perception (Section 1.3.2). In the first part the focus is on data representation and efficient modeling of audio signals for biometrics recognition; the

[1] `https://www.mordorintelligence.com/industry-reports/social-robots-market`

latter, instead, covers the recently proposed approaches for developing task-oriented conversational systems.

Chapter 2 is devoted to the presentation of a general-purpose social robot architecture, which is designed to carry out personalized conversations based on perceived biometrics and has been evaluated in real environments through user feedback. Chapter 3 addresses the robustness of social robots' audio perception algorithms by proposing novel neural network layers able to learn an audio representation directly from data and to efficiently filter out environmental noise thanks to a time and frequency attention mechanism. Chapter 4 is dedicated to the development of memory and computationally efficient audio neural networks for soft-biometrics recognition relying on the previously proposed neural network layers able to run in real-time on board of social robotics platforms. Chapter 5 discusses the limitations of state-of-the-art transformer models in social robotics applications and presents an improved multitask model for intent and entity recognition. Finally, in Chapter 6 I draw the conclusions of this work.

1.3 State of the art

1.3.1 Conversational artificial intelligence

Conversational agents based on Artificial Intelligence algorithms commonly accomplish to three functions [37]: question answering, task completion, and social chat. In the first case, the agent has to provide concise answers to the interlocutor questions based on a knowledge base; in the second case, the agent is required to accomplish a user task like, for example, meeting scheduling; finally, a social chat agent needs to converse with the users like a human as in the Turing test. Both question-answering and task completion systems belong to the category of *task-oriented* dialogue systems, while the remaining type

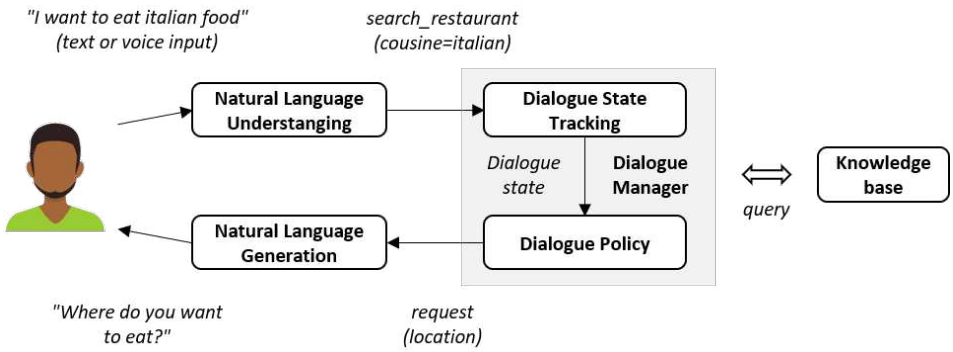


Figure 1.4: General architecture of a Task-Oriented Dialogue system.

of agents is commonly called *chit-chat* dialogue system.

Based on the category it belongs to, a conversational system may be designed in different ways. Task-oriented dialogue system architectures are commonly modular [38] in order to increase the explainability of the system and better control the information flow. Moreover, each module can be substituted without effort with a new one implementing the same interface. The main modules are four, as shown in Figure 1.4.

The input sentence is first given in input to the Natural Language Understanding (NLU) module which is responsible for recognizing the user intentions and extracting associated information. The result of the NLU model is then codified in a semantic structure, also called *semantic frame*. The semantic frame is then passed to the Dialogue State Tracking module which keeps track of the conversation state, *i.e.* representing the system's belief of the user's goal at any point during the dialogue. At this point, the Dialogue Policy module decides the following conversational action to perform, *i.e.* the semantics of the answer to retrieve to the interlocutor, based on the current conversation state. Finally, the Natural Language Generation (NLG) module

translates the conversational action retrieved by the dialogue policy into a natural language response. In last years, some works proposed fully data-driven systems which unifies the architecture using deep neural networks that, taking the interlocutor inputs, retrieve the systems response [39].

The architecture of a task-oriented dialogue system is often extended with two additional modules when dealing with Spoken Natural Language [40], *i.e.* the Automatic Speech Recognition (ASR) module and the Text-To-Speech module (TTS). The first module is in charge of translating speech utterances into text to be given in input to the NLU module. On the other hand, the latter module is responsible for generating a speech signal saying the text generated by the NLG module. Although the extension of the base architectures increases the naturalness of the conversation, it comes with several challenges. First of all, the ASR module is a source of errors which are propagated along the processing chain [41]. The errors have to be taken into account by the NLU and dialogue management modules. For what concerns the TTS module, a spoken dialogue system needs to be expressive and, therefore, also the expression to be retrieved together with a conversational action has to be decided by the Dialog Policy module [42, 43].

Social bots, on the other hand, are structured in a monolithic way. These systems are commonly modeled through *sequence-to-sequence* algorithms trained to mimic large-scale human conversations [44].

This thesis focuses on Task-Oriented Social Robots and, for this reason, in the following chapters, the state-of-the-art algorithms for Natural Language Understanding and Dialogue Management are presented. Particular attention is devoted to performance assessment which is one of the most critical aspects of these systems.

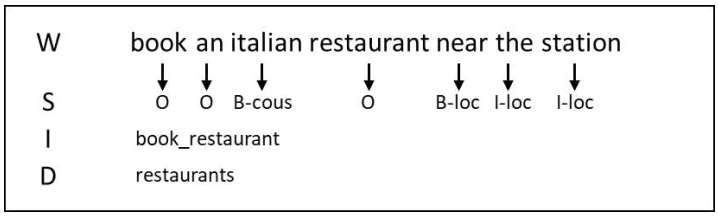


Figure 1.5: Example output of the Natural Language Understanding task. The sentence (W) is represented through its domain (D), the intent (I) and the list of entities (S) in BIO format.

1.3.1.1 Natural Language Understanding

Natural language understanding is a *turn-level* task mapping a sentence into a semantic frame [45]. The semantic frame is a triple composed of three information *domain*, *intent*, and *entities* (a.k.a. slots). Each information within the tuple defines a sub-task, commonly performed in order.

In Figure 1.5 is shown an example of the output of the NLU task. The domain and the intent represent the context to which the sentence belongs to and the aim of the user in that domain respectively. These tasks are commonly faced either as single or, more recently, multi-classification problem usually in a closed-set form. Domain and intent recognition are often performed together as a single task. The entity recognition task can be instead seen as a sequence labeling problem which aims to extract specific entities from the text, like locations and dates.

Nowadays, as in other fields of Artificial Intellegence, the main approaches to NLU are based on Deep Learning algorithms. When dealing with intent recognition and entity recognition, those methods are commonly designed according to a general architecture composed of three layers [46], namely input layer, hidden-layer, and output layer. The input layer is responsible for projecting each word in an embed-

ding space representing semantic relations between words. The embedding function can be either trained from scratch or pre-trained on large-scale corpora in an unsupervised manner [47, 48]. Then, one or more hidden layers are applied to update the embeddings of the input words according to the context information of the sentence. Based on the context each word can assolve different roles and, therefore, this step is crucial for extracting a semantic representation of the sentence itself and the role of each word within it. Hidden layers are commonly implemented through Bidirectional Recurrent Neural Networks (Bi-RNN) [49] and more recent architectures as Attention-based Recurrent Neural Networks [50] and Transformers [51]. Finally, the output layer performs either intent or entity recognition. Commonly used layers are softmax [50] and Conditional Random Fields (CRF) [52]. The latter, exploits the correlation in the predictions of neighbour words for the task of entity recognition, improving the recognition of entities composed of several words such as, for instance, street addresses or full names.

According to [46], two neural-based paradigms can be identified in the literature: *independent models* and *joint models*.

Independent models adopt the approach of training a different model for each task [49, 50]. These algorithms are designed accordingly to the previously presented architecture. Based on the task we have to deal with, the output layer is applied either on all the hidden states or on the last hidden state of the encoder layer for the entity and intent recognition tasks respectively. These approaches are particularly expensive due to the need of using two neural networks for tasks that share several properties. Moreover, the interactions between intent and entities are not exploited and the pipeline using two methods may introduce additional error propagation.

Contrary to independent models, joint models try to exploit the relationships existing between the entity and intent recognition tasks. Two categories of joint models can be identified: *parameters and state*

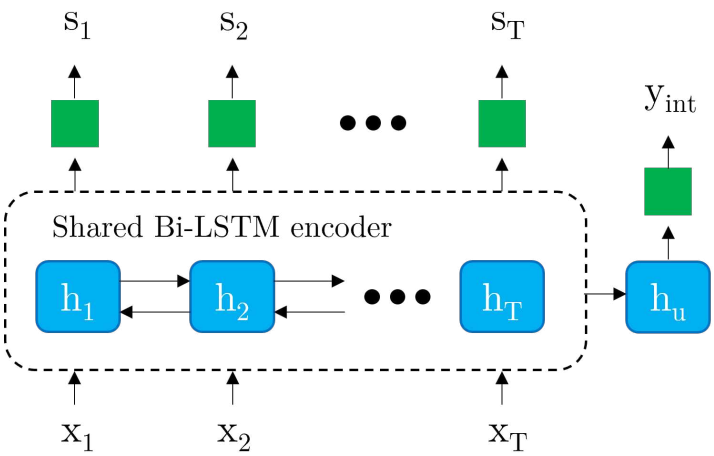


Figure 1.6: RNN-based parameter sharing model. The shared representation of each word in output from the bidirectional LSTM is given in input to a dense layer to predict the slot label s_i. Moreover, the final representation of the sentence is used to predict the intent y_{int}

sharing and *gating mechanism*. Parameters and state sharing methods, as suggested by their name, share information about the two tasks through either weights or states. In the first case (represented in Figure 1.6), a shared encoder, usually a Bidirectional RNN (Bi-RNN), is trained by using a multi-objective loss function to obtain a shared representation [53]. Then, to produce the prediction, several approaches have been proposed, for instance, encoder-decoder methods with a different decoder for each task and attention-based seq-to-seq models. More recently, with the extensive and successful adoption of transformers in Natural Language Processing tasks, the scientific community started to fine-tune pre-trained large-scale language models (*e.g.* BERT) for the two tasks [54]. State sharing methods differ from weights sharing, in that they use two recurrent models

that share states [55]. It is worth mentioning that this approach still requires two models to be (jointly) trained. Finally, gating methods explicitly model the dependency between entities and the detected intent through the use of a gating layer, function of a *slot context vector* (different for each word step) and a *global intent context vector* [56].

The performance of Natural Language Understanding algorithms is commonly measured through the Accuracy and the F1 Score [46]. For the intent recognition task, the evaluation is performed at the sentence level by dividing the number of correct predictions of the classifier by the total number of sentences to predict. On the other hand, the entity recognition performance is evaluated through the F1 Score, *i.e.* the harmonic mean between the precision and the recall. The precision is computed as the ratio between correctly identified entities and the total number of predicted entities. The recall, instead, is computed as the percentage of recognized entities. A correctly recognized entity is considered as such if the boundaries in the BIO format (Figure 1.5) are exactly identified. Errors in the boundaries are commonly not distinguished by not recognized entities in the evaluation.

The main datasets used to benchmark NLU modules are ATIS, MEDIA, SNIPS-NLU, SLURP and Facebook Multilingual. ATIS (Airline Travel Information System) dataset [57] is surely the most adopted one. It contains more the 5K utterances related to the airline domain like, for instance, search a flight. The MEDIA dataset [58] contains french simulated conversations between humans and hotel representatives. The SNIPS-NLU [59] and SLURP [60] datasets are crowd-sourced datasets related to digital assistants. They are available in both written and spoken forms, allowing the evaluation of both the speech recognition and natural language understanding modules. In the last years, several multi-lingual datasets have been proposed, *e.g.* the Facebook Multilingual Dataset [61], allowing the evaluation of language-agnostic algorithms. The details of the mentioned datasets are reported in Table 1.1.

Table 1.1: Standard datasets for Natural Language Understanding and their statistics.

Dataset	Language	# intent	# entities	# sentences
ATIS	English	18	83	5.871
MEDIA	French	-	68	17.172
SNIPS-NLU	English	7	39	14.484
SLURP	English	91	55	16.521
Facebook Multilingual	English	12	11	43.323
	Thai	12	11	8.643
	Spanish	12	11	5083

At the state of the art joint models are competitive with independent models with half of the parameters (weights-sharing approach) on reference datasets ATIS and SNIPS. Moreover, in presence of enough computational power, fine-tuning a pre-trained transformer model such as BERT allows to maximize the performance of both intent and entity recognition tasks [46].

1.3.1.2 Dialogue Management

Once the Dialogue System has understood the intent of the current utterance of the user, it has to answer in the most proper way by taking into account the history of the conversation. As anticipated, this behavior, namely Dialog Management is composed of two parts: Dialogue State Tracking and Dialogue Policy.

The Dialogue State Tracking (DST) module defines and is responsible for updating a representation of the belief state of the conversation based on the dialogue history up to a specific time t [62]. In fact, the NLU module extracts intents and entities at turn level and it is a task of the DST module to store and to track this information. Fur-

thermore, the DST module has to take into account the uncertainty within the prediction of the NLU module; for instance, in the sentence "Book a pizza restaurant for five", the word "five" may refer to both the hour or the number of people. With an additional question "how many people" followed, for instance, by the answer "three people", the DST has to increase the confidence of the association between "five" and the booking hour while setting the number of people to three.

The simplest approach to DST is based on a binary representation of the available entities with respect to the ones needed to perform the task [63]. In this way, the dialog policy can decide if it is needed to ask the interlocutor for additional information. In a more elaborated version, the state can be represented by the list of N-best list values for each entity type and the related probabilities [64].

Once a useful representation for the dialog policy module has been identified, it is important to define the way for updating it. In the past, the updating algorithm was either based on rule systems or on statistical learning algorithms like, for instance, conditional random fields [37]. More recent methods are based on deep learning algorithms like Neural Belief Tracker [65] and TRAnsferable Dialogue statE generator (TRADE) [66]. These methods try to solve the problem "does this pair (entity-value) need to be updated in this conversation step?" and, therefore, require an ontology with predefined values. Deep learning based DSTs allow to generalize along multiple entities and tasks [66] but they do not scale with the number of entities to recognize and, therefore, they are not a feasible solution in the context of social robotics.

The Dialogue State Tracking Challenge (DSTC) [62] is a series of challenges specifically organized as a benchmark for DST algorithms. The evaluation metrics include both Accuracy and Ranking-based scores. In the first case, the metric is computed considering the number of conversational turns in which the most probable value for the entity of interest is correct. On the other hand, ranking-based scores

take into account also the rank of the correct label in case of a wrong prediction.

Based on the current conversational state, the Dialogue Policy (DP) module has to identify the best conversational action to retrieve to the interlocutor. Initially, dialogue policies were implemented through finite state machines (FSMs). However, the design and the update of FSMs containing millions of states is very difficult and time-costly. With the availability of a sufficient amount of conversational corpora, data-driven methods have been proposed to learn the best dialogue strategy. There are mainly two data-driven ways of dealing with the DP task (Figure 1.7), namely with Reinforcement Learning and Supervised Learning algorithms.

In the first case, due to the sequential nature of the task itself, the problem is generally formulated as a Markov Decision Process. At each conversational step, the dialogue system takes an action and, based on it, it receives a reward. Based on the action, the state is updated. Due to this "trivial" formalization, the early attempts to dialogue policy learning have been dominated by Reinforcement Learning algorithms [21]. In particular, several model-free RL algorithms have been proposed based on Deep Q Networks and Policy Gradients methods [67, 68]. Successively, due to the need of dealing with more complex multi-domain dialogues, hierarchical RL methods have been introduced in order to select the better policy based on the current domain [69].

The main issue related to RL dialogue policy algorithms concerns the fact that a very large number of interactions are needed to train a good policy [21]. Furthermore, in order to obtain a better policy, it is preferable to get feedback from the user after each conversational action. Deploying the system in its early stages in order to learn the policy on the field is a solution generally avoided due to the possible consequences on the product quality. On the other hand, paying users to interact with the dialogue system is very expensive because of the

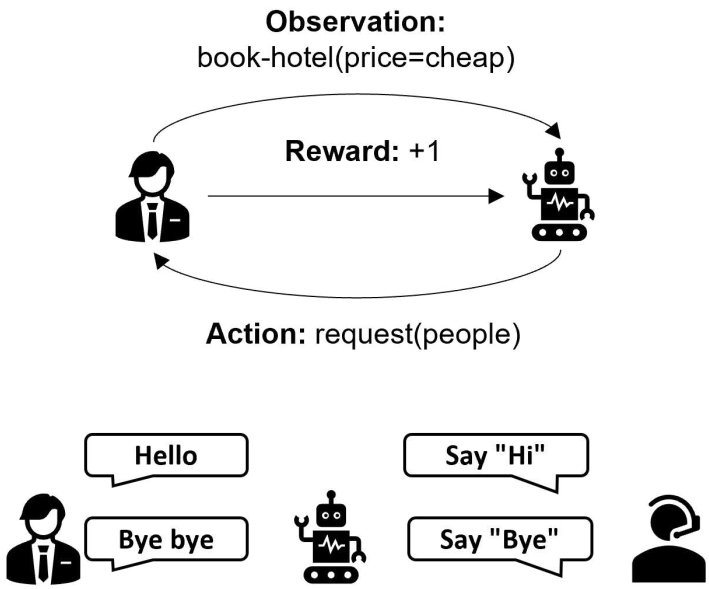


Figure 1.7: Dialogue policy learning approaches. The policy can be learned on-field through Reinforcement Learning algorithms (on top) or over large-scale human-human conversations (bottom).

high number of users needed to generalize over different conversations.

For these reasons, a branch of the scientific community focused on simulating users in order to pre-train RL algorithms [70]. User simulators can operate both at the granularity or methodology level. In the first case, the simulator retrieves user actions in form of intents and entities, simplifying the model [71]. On the other hand, the latter strategy allows to simulate the utterances at the language level in order to train end-to-end systems [72]. State-of-the-art methods for user simulation are mainly based on two approaches, namely agenda-based [72] and model-based algorithms [73]. In the first case, the dialogue is based on a structured representation of the user goal that is exploited

by a set of heuristic generation rules to interact with the dialogue system to train. However, for the more complex task is not possible to define an agenda of the dialogue and these systems commonly lack linguistic variability that may lead to the generalization capability of the dialog management system [21]. To deal with these issues, several scientific works proposed model-based user simulators trained on human-human conversation corpora. Nonetheless, modeling a user conversational system presents the same difficulties as modeling the operator one.

Based on these considerations, supervised dialogue policy algorithms are often preferred to RL ones. Supervised approaches strongly rely on the quantity and variability of conversational data in order to avoid over-fitting.

As happened for computer vision and other fields, dialog management algorithms move from hand-crafted conversation representations to learned ones [74]. These methods are commonly based on Recurrent Neural Networks and, therefore, the conversation representation is deeply learned in a data-driven fashion. More recent approaches substituted RNNs with modern Transformers [75] directly working on the raw input text.

While the evaluation of the single components of a conversational system can be performed through metrics like Accuracy, F1 Score, etc., it is a very challenging task to evaluate the performance of the overall architecture [76]. The evaluation takes into account two main points: the completion of the task and the time spent in the conversation. The main metric related to the first point is the *task completion rate* and is computed as the ratio between the number of conversations that successfully solve the user's problem over the total number of conversations [37]. Also, soft variations of the metrics have been proposed taking into account the partial completion of the task [77]. In the Reinforcement Learning setup, a positive reward is given at the end of the conversation if the task has been successfully completed or

a negative reward instead. With regard to the second type of evaluation metric, it is a common choice to use as a loss function the number of conversational turns required for the dialogue system to complete the task. A smaller average conversation time translates into a better evaluation score. In this case, RL systems are penalized at each conversational turn with a negative reward [78].

Once defined the evaluation metrics, the main issue in the evaluation of these systems lies in the impossibility of an automatic evaluation. Therefore, all the considerations made in the previous section about the training of RL dialogue policies can be replicated for evaluation. In this case, simulated users can translate into a biased evaluation and, therefore, an overestimation of the performance.

1.3.2 Audio perception in social robotics

In the last years there we assisted to a growing interest of the scientific community devoted to audio-only and multi-modal perception algorithms for social robots. This branch of research aims to increase the robustness of image-only algorithms, typically not reliable enough in presence of brightness variations, scene occlusions, and other disturbances [79]. Moreover, some tasks can be accurately approached only through the audio modality, like Automatic Speech Recognition (ASR) and Sound Event Detection (SED) [80, 81].

Although audio signals are invariant to the above-mentioned video challenges, audio analysis systems hide several challenges: first, social robots are commonly equipped with different types of microphones to respect the requirements in terms of budget, space, and power consumption [82]. Each microphone has different properties, for instance, bandwidth, cardioid diagrams, and sensitivity. This variability results in the fact that the same audio signal may be given in input to an audio analysis algorithm with very different representations based on the microphone characteristics [83]. Second, social robots have to work in

highly variable environments very often affected by background noise. These working conditions, together with the variability in the signal energy due to sound source distance and amplitude, cause a drastic reduction of the Signal-to-Noise Ratio (SNR) [84]. A low SNR makes it difficult for audio analysis algorithms to correctly distinguish the sound of interest from noise. In addition to input devices and working environment variability, social robots' audio analysis algorithms has also to deal with the duration of the input signal. Even considering the same type of sound, like voices, the input length may significantly change. For this reason, audio analysis algorithms must be designed to take into account the temporal evolution of the sounds to analyse [85]. Finally, with few exceptions, the amount of audio training data is commonly limited, unlike the huge databases available for various computer vision tasks [86].

In the following chapters, the state of the art regarding machine learning systems for audio analysis is presented. Moreover, a detailed analysis of memory and computationally efficient models in the context of social robots' perception is reported in Section 1.3.2.2.

1.3.2.1 Audio representation learning

An audio analysis system, like most pattern recognition systems, is commonly composed of two main sub-processes: feature extraction and acoustic modeling [87]. Roughly speaking, the first step is in charge of extracting an information-full (depending on the task) and a small representation of the model input; the latter step, instead, models the relationships between the input representation and the related label.

Before the revolution introduced by the deep learning paradigm [88], features were designed according to temporal characteristics of the audio signal, like for instance, the zero crossing rate, *i.e.* the number of times the signal amplitude crosses the zero value, and the time

domain envelope, *i.e.* the boundaries within the signal can be contained [89]. Considering that the perception of sounds is strongly affected by the frequency components [90] of the sound itself, temporal features have been commonly combined with spectral ones. An example of spectral features is the spectral roll-off, *i.e.* the low-pass frequency allowing to keep a specified amount of the signal energy [91], wavelets [92] and Mel-Cepstrum coefficients [93].

In order to compute these features over the audio stream acquired from the robot microphone, it is required for audio analysis systems to frame the input signal [94]. The audio stream is commonly framed using a sliding window which is characterized by the window and the step sizes. These two hyper-parameters must be properly tuned to obtain a good representation of the audio signal. In particular, the spectral features computed from the Fast Fourier Transform (FFT) [95] assume a stationary signal and, therefore, require the right signal length to properly model the frequency components and simultaneously avoid significant variations in the obtained frame. Also, the step size is a very important parameter, it allows to analyse the signal components that have been split by two consecutive non-overlapping windows. In addition to the above-mentioned considerations, the tuning of the sliding window parameters strongly affects the computational requirements for the resulting audio analysis system; for instance, a small window size together with a small step size drastically increases the number of audio frames to process. When framing a signal and computing the FFT, it is important to take into account the high frequencies implicitly introduced in the transformation. These frequencies may cause the aliasing phenomena [96]. To avoid spectral features alterations, it is common to multiply the obtained audio frame by a low-pass filter window. Common choices for the windowing functions are Hamming, Hanning and Blackman [97].

Hand-designed audio features have historically been combined with acoustic models based on traditional machine learning algorithms.

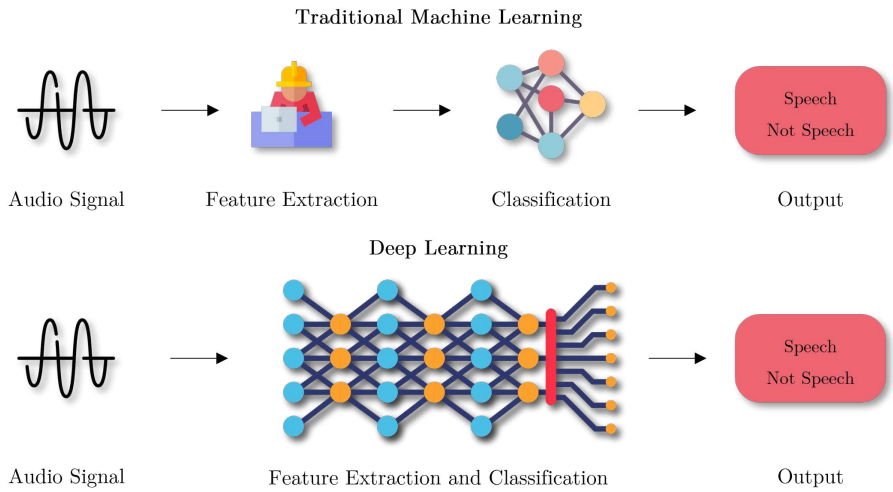


Figure 1.8: Paradigm shift from traditional machine learning (top), based on hand-crafted features and statistical classifiers, to deep learning, based on deep neural networks (bottom).

In the particular case of classification tasks, it was very common to adopt Support Vector Machines (SVM) and Artificial Neural Networks (ANN) [98, 99, 100]. On the other hand, for those tasks dealing with sequences, like ASR, algorithms modelling also temporal evolution like Hidden Markov Models were preferred [101]. Traditional machine learning algorithms and hand-crafted methods proved to generalise well on new data even when trained over small training sets. Nevertheless, these kinds of approaches are upper-bounded when the number of training samples increases (*i.e.* underfitting) [102] and don't allow them to reach the performance level required to deploy the final system in sensitive contexts. Moreover, the designing of effective features may be a very expensive work and requires an expert in the domain.

The approach to audio analysis disruptively changed with the in-

troduction of Deep Neural Networks (DNNs), and trainable representations in general [103, 104]. As shown in Figure 1.8, The Deep Learning paradigm substitutes the feature engineering process and shallow classifiers with very deep neural networks able to perform both tasks directly from data. In particular, DNNs take as input directly raw data and learn the most proper representation in an unsupervised manner during the optimization process. Deep learning methods proved to be very effective in learning those processes that humans perform being unable to describe, in an algorithmic way, their expertise (like, for instance, computer vision, speaker diarization, and object manipulation) [105]. The killer characteristics of this kind of learning algorithm reside in the fact that no prior knowledge of the domain is required to train an accurate model and the learned representation can be reused for different applications through a process called transfer learning [106]. The reusing of learned representation allows deep learning models to deal with tasks where the quantity of data is limited, *e.g.* the medical imaging field.

Based on the same considerations done above about the importance of spectral components in modeling acoustic signals, a very popular way to apply deep learning to audio signals is to give in input to the neural network a time-frequency representation of the input signal [107]. Among these kinds of representations, the Short-Time Fourier Transform (STFT) is surely the most used one. The STFT formula is reported in Equation 1.1. The f-th component of the t-th frame of a discrete signal windowed with size N is defined as follows:

$$\mathcal{X}(t, f) := \sum_{k=0}^{N-1} w(k)x(tN + k)\ e^{\frac{-2\pi ikf}{N}} \tag{1.1}$$

Where w is the windowing function (*e.g.* Hamming). The resulting representation is commonly named Spectrogram. Also in the case of the STFT, it is common to use a step size smaller than N in order to

obtain a smoother representation [89].

The spectrogram is a particularly well-suited representation for deep learning models, and in particular Convolutional Neural Networks (CNNs), due to the locality of the spatial relationships in the two-dimensional representation (*i.e.* adjacent frequency components and time frames) [108]. Nevertheless, it comes with some drawbacks. In particular, the spectrogram suffers from a trade-off between the resolution in the time and the frequency domain, as stated in Gabor's uncertainty principle [109]. A small window size allows to obtain a spectrogram very detailed in the temporal domain; on the contrary, a big window size allows to increase the frequency resolution. An example related to a trumpet sound is reported in Figure 1.9. It can be seen as the middle spectrogram (window size equal to 5ms) is characterized by a very high time resolution but the frequency axis seems to be blurred. Viceversa, the latter spectrogram (window size equal to 20ms) is characterized by a low temporal resolution and a high frequency one, allowing to identify the sound harmonies. In practical applications, it is common to identify a good trade-off between the frequency and the temporal resolution based on the task one is dealing with [110, 111].

With the progress of the biological sciences field, humans modeled how their auditory system responds to the different frequencies [112]. Between these representations, the Mel Spectrogram is surely the most widely adopted one[113]. The Mel spectrogram is the result of a filterbank modeling the auditory system represented in the time-frequency domain. Each filter of the filterbank is modeled through a $sinc^2$ function, which corresponds to a triangular filter in the spectral domain, as shown in Figure 1.10. The mel filterbank applies a non-linear transformation of the frequency scale, called Mel Scale [114], which is reported in Equation 1.2.

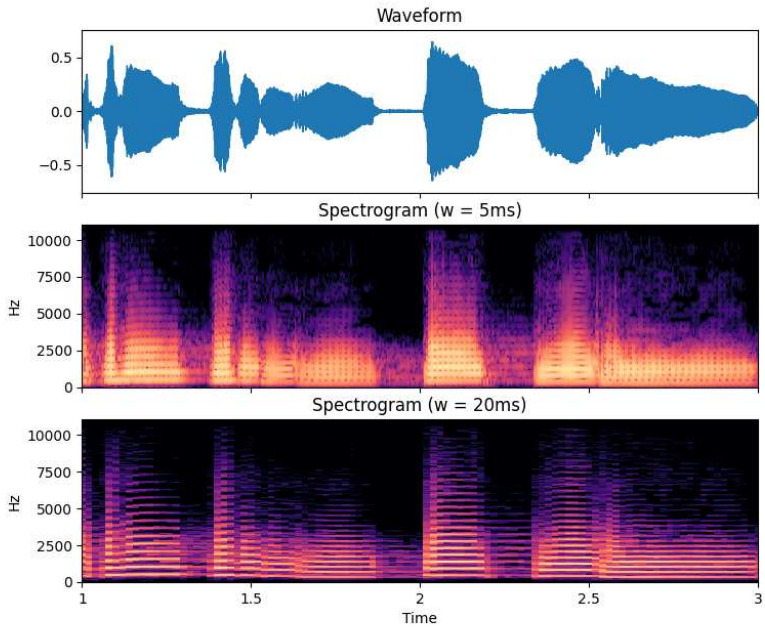


Figure 1.9: Comparison between the raw waveform of a trumpet sound and its spectrogram representation computed with a window size of 5 and 20 milliseconds.

$$m(f) = 2595\ log_{10}\left(1 + \frac{f}{700}\right) \tag{1.2}$$

where m(f) represents the Mel scale mapping function between the frequency domain and the mel one. This mapping results in a linear behaviour below 1KHz while being logarithmic above this threshold. To speed up the computation, the Mel spectrogram is commonly computed as a linear matrix multiplication between the spectrogram and

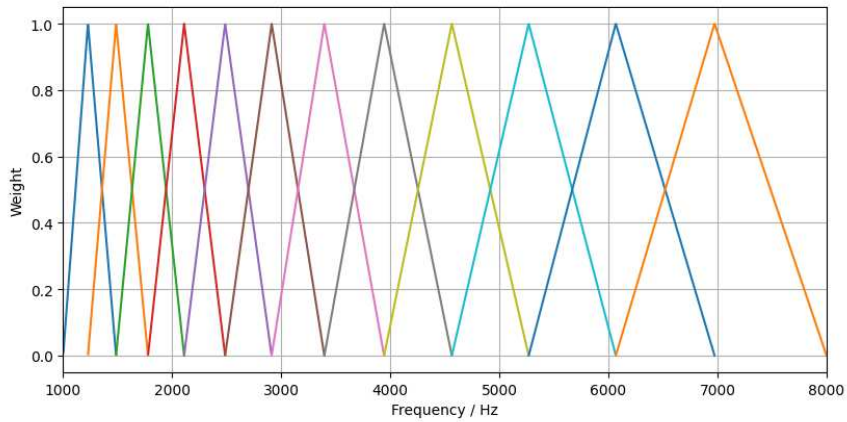


Figure 1.10: Mel filterbank. The filterbank is composed of 12 filters in the range 1KHz - 8KHz at a sampling rate equal to 16KHz.

a projection matrix corresponding to the mel filterbank. The multiplication in the frequency domain substitutes the convolution in the temporal one.

An alternative representation to the Mel Spectrogram is the Gammatogram [115]. This representation better models the human auditory systems relying on a different filterbank. The Gammatone filterbank has been originally introduced in [116] and it took its name from the gammatone impulse response:

$$g(t) = at^{n-1}e^{-2\pi bt}cos(2\pi f_c t + \phi), t > 0 \tag{1.3}$$

where n is the filter order and b represents its bandwidth. As for the Mel filterbank, the Gammatone one is distributed according to a logarithmic scale, namely the Equivalent Rectangular Bandwidth (ERB) scale, reported in Equation 1.4.

$$e(f) = 24.7(4.37f + 1) \tag{1.4}$$

Also in this case, $e(f)$ represents the ERB mapping function between the two reference domains.

The spectrogram and spectrogram-like representations are commonly combined with CNNs and Convolutional Recurrent Neural Networks (CRNNs) to extract a deep representation of the audio input signal. The time-frequency representation is given in input to the model according to three main approaches: through an image conversion, as a multi-channel one-dimensional signal, and as a raw two-dimensional signal. The image conversion process allows scientists to transfer learning from image-related tasks, which are characterized by huge datasets [117]. Unfortunately, this is not true for all the tasks and, in some cases, all the previous knowledge is discarded. Moreover, the RGB conversion process requires a projection in a fixed range which, based on the projection function, may drastically affect the performance of the final system [118].

For these reasons, common alternative approaches are to directly feed in input to convolutional neural networks the raw time-frequency representation. Spectrogram-like representation can be seen either as a multi-channel one-dimensional signal or a single-channel two-dimensional signal [119]. The former case treats the frequency components as feature maps and exploits only time locality through 1D convolutional filters. On the contrary, the latter approach exploits both time and frequency locality to avoid collapsing the frequency information. The first approach results in a lighter model able to better deal with small training datasets to achieve good generalization but may suffer from frequency information collapsing. On the other hand, two-dimensional CNNs are commonly preferred due to the possibility to better analyze time-frequency relationships and the possibility to rely on well-studied architecture adopted in computer vision.

Even if spectrogram-like representations have been the main approach to deep learning based audio analysis, they still rely on several hyper-parameters which significantly affect the performance of the final system. Considering all the considerations made above, *i.e.* the time-frequency resolution trade-off and the need for more unsupervised representations, the signal processing scientific community started working on models taking as input directly the raw waveform [120, 121, 122]. These models take as input the raw waveform and stack several layers of strided convolutions and max-pooling to create a compact representation given as input to the classification layers.

The main advantage of end-to-end raw-waveform neural networks for audio signal processing is the decoupled time and frequency resolution. The frequency resolution of raw-waveform models can is a function of the number of 1D convolutional filters; on the other hand, the temporal resolution can be regulated through the stride and the convolutional filters [123]. As a drawback, these models need to be very deep to perform better than spectrogram ones and, consequently, they need huge amounts of data to generalize well [124]. Moreover, once fixed the CNN architecture and the input length, the common increase of the sampling rate, required to obtain a better resolution of the frequency spectrum, further amplifies the above-mentioned problem.

1.3.2.2 Efficient speech analysis neural networks

Social robots, as previously discussed, have to recognize several biometrics of the interlocutor to provide an emphatic human-robot interaction. While these problems have been well studied by the computer vision research community, the audio modality is recently gaining attention [125, 126, 127, 128]. Within this context, the main objective is to exploit the multiple modalities available on the robot to robustly recognize biometrics even in challenging environments (*e.g.* low light-

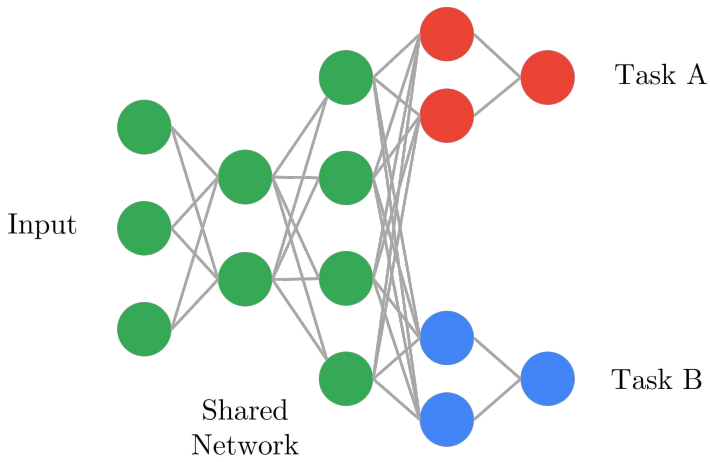


Figure 1.11: Multitask neural network concept. A shared network (green neurons) is used to compute a general-purpose representation from the input. Then, task-specific branches (red and blue neurons) are used to perform task-specific operations.

ing conditions, noisy environments).

Even if deep learning algorithms obtained notable results in recognizing the interlocutor biometrics from his/her voice, processing the audio stream using simultaneously multiple neural networks to deal with these problems is generally too demanding for this kind of application where the computations are done on board of low-power embedded devices characterized by limited GPU and memory resources [23]. It becomes then crucial to identify an efficient way to analyse the audio stream and perform other tasks (robot vision, planning, *etc.*), on board of the robot to avoid the introduction of network latency and to interact in real-time.

Multi-task neural networks (MTNs) [129] are recently attracting interest as a solution to these problems. MTNs use a common feature

extraction backbone to obtain a high-level representation of the input. Then, multiple branches of the network are used to perform each specific task. The possibility of using a shared network for extracting features allows the reduction of the memory required for the model loading proportional to the number of tasks to solve and, therefore, to speed-up the inference time. MTNs have been exploited for several applications ranging from face analysis [130] to speech [131] and language analysis [132].

Despite the significant engineering contribution of MTNs, the field of speech analysis commonly focuses on a single task (*e.g.* Automatic Speech Recognition) and exploits additional related problems (*e.g.* gender recognition) to increase the performance of the model over the main task [133, 134, 135]. In fact, it has been theoretically proved that the Multi-Task Learning paradigm guarantees a more generalized representation [136] thanks to the regularization action that additional tasks perform on the main one.

A common limitation of these methods consists of the strict requirement for the dataset adopted for training the MTN to provide, for each sample, the labels for all the considered tasks [131, 134, 137, 138]. This requirement makes it very difficult to use multiple datasets to increase the number of audio samples and, therefore, increase the generalization capabilities of the MTN. Those who treated multiple datasets either worked on the intersection of the available tasks [139] or followed a transfer-learning approach [140]. This approach is particularly limiting in social robotics applications, where there are no datasets that contain the labels for all the tasks of interest and, therefore, it is not possible to train a single network for all the tasks we deal with.

In addition to the availability of data, a crucial difficulty is represented by the losses imbalance [141] due to the need for different tasks to minimize different risk functions (*e.g.* mean square error and cross-entropy). In particular, different types of loss functions are commonly characterized by different magnitudes which result in gradient

unbalance and, therefore, in the predominance of a task w.r.t. others. Until now, this problem has been faced through the hard weighting of the losses contributions [142, 143]. However, this approach is time-wasting due to the need of optimizing additional hyper-parameters equal to the number of tasks. The number of weights to tune further increases when dealing with multiple datasets, as in the case of social robotics applications. In this case, due to the different sets of problems to deal with, and thus different losses, the weights have to be different in order to balance the gradients related to that particular sample (or batch of samples).

Regarding the MTN design, it is common to design an MTN by sharing all the layers except for the output ones [144, 145, 146]. Even if this can be a useful choice, in particular for limiting the computational resources of the algorithm itself, the limited size of the final feature vector used for the classification step may not be sufficient for bringing the information needed to perform at the best all the tasks. For this reason, some MTNs share only part of the network and then create a branch for computing high-level problem-specific features for each task [133, 147]. Nevertheless, the effect that these design choices have on the performance of the different tasks has not been discussed yet.

1.3.3 Contributions of this thesis

This thesis tackles the requirements of social robotics applications in the context of Deep Learning methods.

As a first contribution, I propose a general-purpose social robot software architecture allowing parallel multimodal sensors data processing. The proposed architecture is equipped with a novel Dialogue Manager model able to personalize the conversation based on the interlocutor's biometrics. The proposed architecture has been deployed in a real prototype and evaluated through real user feedback. The experiment pointed out the need for robust audio analysis algorithms

and more efficient sensor processing pipelines. More details about the proposed model and the prototype performance assessment are reported in Chapter 2.

In light of the considerations made in Section 1.3.2, I propose in this thesis, as a second contribution, a raw-waveform and spectrogram-based audio representations. The two representations are specifically designed to fill the gaps identified in the context of social robotics: robustness to environmental noise, task adaptability, and real-time computation on embedded systems. The details about the proposed representations, together with the experimental validation of the performance are reported in Chapter 3.

As a third contribution, I focus on further optimizing audio analysis deep learning algorithms in the context of social robotics applications. In particular, I propose three audio multitask architectures for simultaneously dealing with the following tasks: Speaker Re-Identification, Emotion and Gender recognition, and Age estimation. I also propose a label masking approach to deal with datasets with missing labels and to automatically tune the task weights with the GradNorm algorithm [141]. The multitask architecture relies on the spectrogram-based representation proposed in Chapter 3 to provide more robust predictions. More details about the proposed multitask networks and the obtained results are reported in Chapter 4.

Finally, even if Transformer based models like BERT represent a breakthrough for the conversational AI field, due to their computational requirements, they do not fit well in the social robotics context. As the last contribution, in this thesis, I propose a new multitask NLU transformer architecture, specifically designed to run on board of embedded devices. The details of the proposed method and the related performance validation are reported in Chapter 5.

The last three contributions allow to advance the state of the art of deep learning in the context of social robotics by focusing on its gaps as they are perceived by the final user.

Chapter 2

Social robots: from design to human feedback evaluation

Based on

A social robot architecture for personalized real-time human-robot interaction.
Foggia, P., Greco, A., Roberto, A., Saggese, A., & Vento, M. - SUBMITTED TO Journal of Internet of Things.

2.1 Background

In this chapter, I propose a general software architecture for social robots. To this aim, I firstly identified the main requirements that a social robot must accomplish. As presented in Chapter 1, a social robot must: i) analyze images and audio acquired from its sensors; ii) perform multi-modal aggregation to increase the measurements precision and the robustness to environmental noise; iii) manage its behavior with the interlocutor in terms of movements and actions; iv) orchestrate and personalize the dialogue. In addition to application-dependent constraints, the proposed software architecture has been designed to guarantee independence from the hardware of the specific robot used and adaptability to different social contexts with minimal changes.

To achieve these goals, I propose in this chapter a novel dialogue manager algorithm able to personalize the human-robot interaction by exploiting the biometrics perceived by the sensors of the social robot; moreover, the architecture includes a multimodal sensor aggregation module able to exploit the information acquired by different types of sensors to increase the robustness to environmental noise; finally, parallel processing pipelines that, properly designed and implemented, ensure real-time performance. The realization with ROS modules also guarantees hardware independence.

To validate the proposed architecture, state-of-the-art algorithms have been selected and integrated into the processing pipeline together

with the proposed components to realize a social robot to be deployed in the SICUREZZA exhibition for three days. 161 people who interacted with the robot evaluated their experience by answering to 5 questions with a score between 1 and 5. The maximum score was achieved for more than 40% of the answers and the average rate was between 4 and 5. This result acquires more relevance considering that the people who attended the conference were technically skilled and, therefore, their feedback is reliable. The survey also allowed to identify the need for improving the robustness to environmental noise of audio analysis algorithms and for reducing the latency of processing nodes.

2.2 A general-purpose social robot architecture

The proposed social robot architecture is depicted in Figure 2.1. It has been realized with the two-fold purpose to meet all the above-mentioned requirements and to develop a general-purpose architecture that can be adapted to perform different tasks. To this aim, a modular software architecture has been designed based on the Robotic Operating System (ROS), a de-facto standard framework for robot programming. It provides hardware independence and the possibility to easily implement nodes communicating according to the publisher/subscriber protocol. In this way, it is possible for the developer to quickly realize parallel code on board of different types of robotics platforms and to speedup the inference to obtain real-time performance.

The proposed architecture is composed of the following modules: sensors and actuators, sensor data analysis, multimodal sensors aggregation, multimodal re-identification, behaviour manager, dialogue manager, and input/output (I/O) service manager. All the compo-

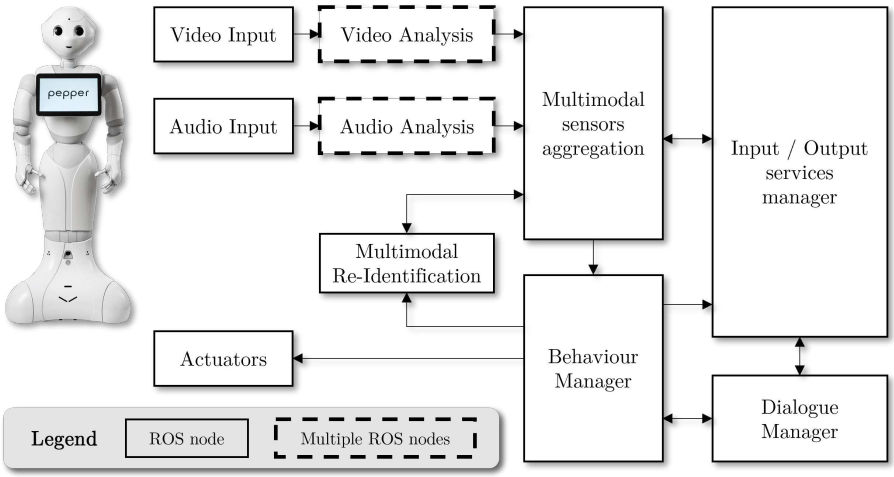


Figure 2.1: The proposed social robot architecture. Video input, audio input and actuators are the ROS nodes that allow to abstract the hardware. The pipelines (multiple ROS nodes) for video and audio analysis process the information acquired by the camera and microphone in parallel and their results are then aggregated and synchronized by the multimodal sensors aggregation node, which provides useful information for the multimodal re-identification of the interlocutor. The behavior manager receives the outputs of the previous nodes and of the dialogue manager, that orchestrates the conversation, and controls the actuators to manage the human-robot interaction. If necessary, the Input/Output services manager provides the access to privacy-safe services on the network.

nents are presented in the following sections.

2.2.1 Sensors and actuators

To meet the requirements related to personalization and multi-modal analysis, the minimum hardware that the social robot must be equipped with is a RGB camera, an array of microphones, and a speaker. These hardware components are generally available on robotics platforms commonly used to develop social applications. With this equipment, the robot can be aware of people (and their faces), objects and voices, and to converse with humans. Other applications may require additional sensors and actuators, such as depth and proximity sensors (depth camera, lidar, radar, sonar), wheels, arms, and tablets. For this reason, it is important to define a software architecture independent on the specific hardware adopted for the use case and easily adaptable to different purposes.

Therefore, the proposed architecture includes two ROS nodes, video input and audio input, responsible for acquiring video and audio data from the sensors of the social robot. Each node publishes the acquired data on a dedicated ROS topic with a standard interface. In this way, it is possible to design a *hardware-agnostic* software architecture. In particular, changing the social robot and/or sensory equipment only implies (if necessary) a re-implementation of these two nodes. Also the addition of new sensors is quite easy, because a new similar node may be defined.

Symmetrically, a ROS node (actuators) acts as an interface to control the robot actuators. This component follows the design approach adopted for the input nodes, *i.e.* decoupling the application logic from the specific hardware available on the social robot.

2.2.2 Sensor data analysis

The raw data acquired from the sensors of the robot are independently analysed by two processing pipelines, one for video modality (Section 2.2.2.1) and one for audio modality (Section 2.2.2.2). In both pipelines, the nodes communicate information with a publisher/subscriber protocol in order to parallelize the computations as much as possible. Moreover, the scheme of this processing pipeline allows a ROS node that has already processed its data to start processing new data without waiting for the other nodes. In this way, it is possible to drastically speed-up the inference. Moreover, the architecture has been designed in order to be easily extended with additional functionalities.

The following subsections describe in detail the two processing pipelines and the related components.

2.2.2.1 Video analysis

The video analysis pipeline, depicted in Figure 2.2, initially processes the camera frames through two ROS nodes: object detection and face detection. The former is responsible for the localization and the recognition of the objects in the environment in which the social robot has to work. The latter has to detect the position of the people in the scene that are probably going to interact with the robot. This node is extremely important for social robots because all the biometrics of the interlocutor are computed from his/her face image. The face detection result is passed to other two nodes: face embedding and soft-biometrics recognition. The first node extracts a vector representation of the detected faces in order to allow similarity-based tasks like people tracking and face re-identification. In parallel, the second node extracts from each face image the most commonly considered soft-biometrics, *i.e.* age, gender, emotion, and other facial attributes.

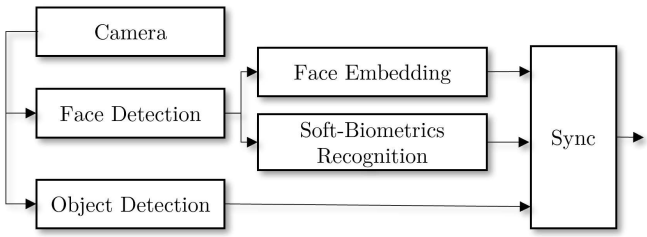


Figure 2.2: Video analysis pipeline. The images acquired from the camera are passed to the face detection and to the object detection nodes to recognize people and objects in the scene. For each person, the face embedding node extracts a feature vector encoding the identity and the soft-biometrics recognition node estimates gender, age, emotion and other facial attributes. Finally, all the computed information are synchronized.

These information are crucial to develop emphatic behaviour in social robots, which can personalize the behaviour accordingly even if the interlocutor has never been seen before. Finally, when all the information have been extracted, a node synchronizes them in order to manage the different inference times of the involved ROS nodes.

2.2.2.2 Audio analysis

The audio analysis pipeline, represented in Figure 2.3, processes the audio stream with two ROS nodes: voice activity detection and sound event detection. The former is responsible for the identification of an utterance in the audio stream and the retrieval of the related audio window. The latter detects and recognizes events that happened in the surrounding environment by analyzing the sound. Their output is passed to the sound source localization node, which localizes the position (w.r.t. the social robot) of the source of the detected sound. In this way, it is possible for the robot to understand the position of

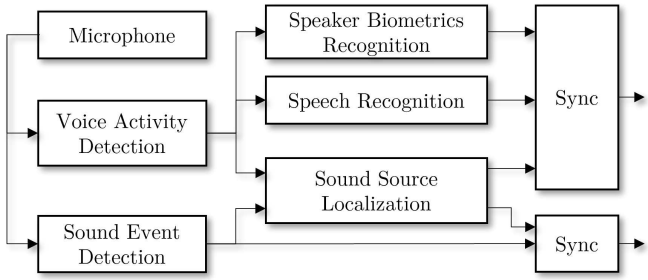


Figure 2.3: Audio analysis pipeline. The audio stream is processed by the voice activity detection and the sound event detection nodes to detect speech and sounds of interest. The sound source position is computed by the sound source localization node. The audio windows containing speech are used in parallel by the speech recognition node, which transcribes the text, and by the speaker biometrics recognition node, which estimates identity, gender, age, emotion, and other attributes from the voice. Finally, all the information related to the same audio event are aggregated.

the interlocutor and events of interest, even if they are not in the field of view of the camera.

The output of the voice activity detection node is processed by other two additional ROS nodes: speaker biometrics recognition and speech recognition. As suggested by the name, the first node extracts from the voice of the speaker all the soft biometrics, such as age, gender, emotion and a vector representation of his/her identity-related features. This node is crucial when the person is not visible from the camera, but also adds redundancy to the information obtained by the video analysis processing pipeline; therefore, it can improve the accuracy and the robustness of the predicted biometrics. The speech recognition node, on the other hand, transcribes the speech into text, in order to allow a spoken natural language interaction with the social

robot. Finally, as done for the video analysis pipeline, both the sound event detection and speech information are synchronized.

2.2.3 Multimodal sensors aggregation

The high-level information independently extracted from video and audio are combined in the multimodal sensors aggregation node. The fusion of the two modalities allows to get a representation of the state of the scene that aggregates all the available data in the best possible way. For instance, with the information produced by this node the social robot can localize the position of the interlocutor with respect to itself, by exploiting either the outputs of the sound source localization and the face detection nodes. Furthermore, the redundant information acquired from multiple sensors can be aggregated to obtain a more robust awareness of the environment. The obtained representation is then made available to the behaviour manager node with a fixed publication rate.

To this aim, the state of the environment around the social robot is represented with the records reported in Figure 2.4. It is mainly composed of three types of entities: person, object and event. Based on the type of entity, it may contain different information. For instance, only people are characterized by their biometrics but, on the other hand, both people and objects have a position in the environment.

The state of the environment is, by its nature, dependent on the previous state and on the data input collected by the sensors. To exploit this temporal correlation between consecutive acquisitions, it is crucial to associate each visible entity with an identifier, in order to track the related information coming from multiple sensors over time. To this purpose, the system assigns to each entity a high-level identifier, namely the alias, which is implemented as an incremental integer number. In addition to the alias, a person in the scene may be identified through their identity (ID). It is worth noting that the alias

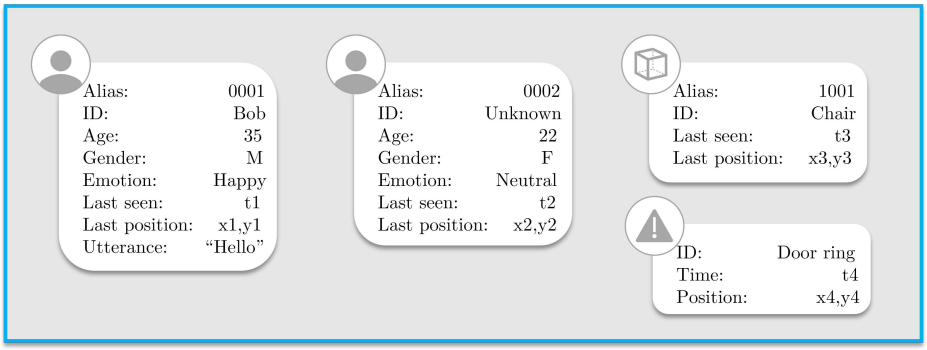


Figure 2.4: Environment state produced by the social robot. It consists of a collection of entities and their information acquired by multiple sensors. The entities can be people, objects or sound events. Based on the category of the entity, different attributes can be linked. For instance, one can recognize the emotion of a person but this is not available for objects.

information is also required for person entities, since an individual may be not recognized; in this case, the ID info is set to "Unknown" and only the alias is taken into account. Of course, the ID value can change over time whether the initially unknown speaker introduces itself during the conversation. In the same way, the alias information is used for objects in the environment, in order to distinguish between multiple instances belonging to the same class.

In order to associate the entities found in a new video frame and the ones present in the scene, it is possible to rely on a hierarchical tracking system composed of three modules: Identity Tracker, IoU Tracker and Similarity Tracker. The first tracker maps the people in front of the camera with the ones in the previous scene representation based on the recognized identity; obviously, it only works with people entities. For objects and not recognized people, the tracking algorithm

tries to perform the match based on the distance between the current position of the new entity and the last position of the entities belonging to the same category (*i.e.* person and object class). To this aim, the Intersection over Union similarity metric (IoU), which is defined as the ratio between the area of the intersection over the area of the union of the considered bounding boxes, can be considered. A bigger overlap between the considered bounding boxes corresponds to an higher similarity. Finally, all the non-assigned bounding boxes can be given in input to the Similarity Tracking module. It computes the similarity between the entities representations (*i.e.* embeddings) and performs a multi-object tracking approach through the Hungarian algorithm. If no matches are found for a specific entity, a new progressive alias is assigned to it.

This approach allows to keep memory of the entities surrounding the social robot, avoiding ghost situations (*i.e.* alternate detection and miss of an entity within consecutive frames). To understand if an entity leaved the scene, an expiring mechanism can be implemented: if the entity is not perceived for a pre-defined period of time it is considered left until the next perception.

At this point, the data acquired from the video analysis pipeline must be aggregated with the information coming from the audio analysis pipeline. Even in this case, the first step is to aggregate information by identity. If the social robot recognizes the voice of a person, it aggregates the perceived information with the related entity in the scene representation. On the other hand, if this entity is not in the scene, it will be added with the perceived information (*i.e.* biometrics, utterance, location, voice representation). If it is not possible to associate the entity through the identity information, the social robot can compute the similarity of the obtained voice and soft-biometrics (*i.e.* age and gender) with the ones of the people in the scene. It follows

the similarity S equation:

$$S = w_1 * v_{ent} * voice_{sim} + w_2 * biom_{sim} + w_3 * loc_{sim} \tag{2.1}$$

The similarity S between the entities in the scene is computed as a weighted sum (being w_1, w_2, w_3 the weights) of three contributions. The first contribution is $voice_{sim}$, that is the cosine similarity between the embeddings of the entity voice and the unknown one. To manage the possibility that an entity never talked before, this contribution is controlled by the flag variable v_{ent} which is equal to 1 if a voice representation for that entity is available, 0 otherwise. The value of $voice_{sim}$ is then scaled in the range $[0, 1]$. The second contribution is the normalized similarity $biom_{sim}$ between the age and gender biometrics information. It is computed as the normalized absolute difference between the two ages (age_{voice} and age_{ent}) multiplied by 1 if the gender of the voice gen_{voice} is the same of the entity gen_{ent}, 0 otherwise.

$$biom_{sim} = \begin{cases} \frac{|age_{voice} - age_{ent}|}{100}, if & gen_{voice} = gen_{ent} \\ 0, & otherwise \end{cases} \tag{2.2}$$

In this case, it is sure that the information is associated to the person because it is computed at least from the video analysis pipeline. The third contribution loc_{sim} is a function of the angular distance between the last position of the considered entity ang_pos_{ent} and the speech source of the listened voice doa_{voice}, scaled in the range $[0, 1]$. It is computed as:

$$loc_{sim} = 1 - \frac{|doa_{voice} - ang_pos_{ent}|}{180} \tag{2.3}$$

The distance is computed on the angular plan because it is the most popular and reliable way to identify the sound source position through microphone arrays at the state-of-the-art.

The fusion of hard and soft biometrics can be fine-tuned through a grid search approach or the Bayesian optimization framework to determine the appropriate weights. Although the similarity measure may not be a perfect indicator of the interlocutor's identity, it serves as a reliable indicator for person identification. It is also important to note that while hard and soft biometrics cannot be used for full identification, the chosen average formula is a practical solution for fusing these two types of biometrics.

Once the entity matching has been identified, each information extracted from the video and audio analysis processing pipelines at the current time ($i(t)$) is used to update in a confidence-based recursive way the aggregated information ($I(t)$) related to the matched entity.

The general formula is reported in the following:

$$I(t+1) = \alpha(i(t)) * i(t) + (1 - \alpha(i(t)))I(t) \tag{2.4}$$

$$\alpha(i(t)) = min(\tau, confidence(i(t))) \tag{2.5}$$

In order to manage sensor faults, the weight $\alpha(i(t))$ is dependent on the confidence of the prediction $confidence(i(t))$. It is recommended a maximum value equal to τ in order to take into account the temporal component.

The weighted average of the acquired information is then made available from the Multimodal sensor aggregation node with a fixed frequency F_{sens}.

2.2.4 Multimodal re-identification

In order to further personalize the conversation with the interlocutor, the multimodal sensors aggregation node relies, if needed, on the output of a multimodal re-identification node. This ROS node is able to exploit the descriptors extracted by the speaker biometrics recognition node (audio) and by the face embedding node (video) to re-identify

known people and insert new identities in the gallery set. For safety reasons, this module works with and stores only high-level descriptors. In this way, it is possible to avoid to manage raw data containing sensible (and not needed) details about the interlocutor. To limit the memory necessary for the storage of the known identities and avoid dangerous leaks, the multimodal re-identification module should include an expiration mechanism for embeddings. The expiration time can be set based on the application requirements and on the available hardware resources.

2.2.5 Behaviour manager

The social robot must adapt its behavior to the state of the environment. To this aim, the aggregated state of the scene computed by the multimodal sensors aggregation node is passed to the behaviour manager node. This node contains the application logic of the social robot and is typically based on a Finite State Machine (FSM); despite this choice, Reinforcement Learning [148] algorithms or Behaviour Trees [149] can be used in alternative without changing the previous nodes.

This module is responsible for starting and keeping the engagement with the interlocutor. Moreover, it relies on the dialogue manager node, described in the following, for understanding the intents through natural language processing algorithms and retrieving the related natural language response. Based on the recognized intent, the behaviour manager node can also execute a required action through the actuators of the social robot.

Finally, the behaviour manager may provide useful additional information to the multimodal re-identification module. For instance, when the interlocutor says his/her name to the robot, this information can be stored and then used for re-identification and personalization. To be more precise, the dialogue manager node retrieves these information from the utterance and the behaviour manager node sends it,

together with the interlocutor's embeddings, to the multimodal re-identification node.

2.2.6 Dialogue manager

Considering the type of conversation that the social robot must have with the human, the dialogue manager node can be designed according to a task-Oriented dialogue system [38], which consists of three main components: Natural Language Understanding (NLU), Dialogue State Tracking (DST), Dialog Policy (DP).

The first component recognizes the intention of the interlocutor by analyzing the pronounced utterance. Moreover, the NLU module extracts the entities of interest within the utterance to accomplish a task; for instance, if the human asks to reproduce a song, the entity could be the title of the song.

Once obtained the intention and entities present in the sentence, the DST module creates a conversation state that can be represented as reported in Figure 2.5. The conversation state includes common conversation information: the current sentence intention Int_c, the entities Ent_c and the history of the previous N conversational turns $Hist_N$ (*i.e.* user intentions and robot answers).

In order to personalize the conversation, it is also useful to add soft-biometrics information about the person the social robot is talking with. Finally, the DP module is responsible to retrieve the answer for the interlocutor based on the current conversation state. The conversational state is based on the information inferred from the user utterances and then allows the social robot to ask additional details, if required, to retrieve the most suited answer. Moreover, due to the fact that also the final user information are encoded in the conversational state, the social robot based on the architecture can select the most appropriate answer for the specific user.

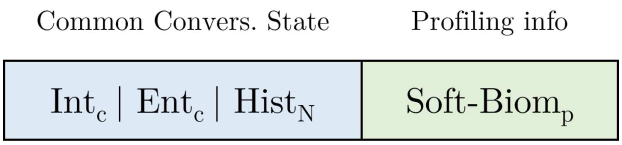


Figure 2.5: Conversation state representation. It includes the common information required to identify the correct answer, namely the current intent of the interlocutor and the related entities (Int_c and Ent_c, respectively) and the history of the last N conversational turns ($Hist_N$). To personalize the answer to retrieve for the specific user, the soft biometrics of the interlocutor are part of the conversation state.

2.2.7 I/O services manager

In order to provide an interface for external services like the reading of sensors data from IoT devices, the dialogue manager node can communicate with the Input/Output services manager node. It acts as an interface to interact with other available IoT services, but it can also provide data from external sensors to the multimodal sensors aggregation node, in order to consider an out-of-robot perspective of the environment. Furthermore, it may expose an interface to external devices that are interested to monitor the social robot and the state of the environment. To this aim, it receives high-level and privacy-safe data from the multimodal sensors aggregation node and from the behaviour manager node. Additionally, the I/O services manager node allows to store and retrieve information from a local database. It can be used, for instance, to store information about the users to further personalize the dialogue and, consequently, make more emphatic dialogues.

Figure 2.6: The social robot realized for the SICUREZZA 2021 exhibition in Rho, Milan, Italy.

2.3 Architecture implementation

The proposed architecture was used to create a social robot able to attract the attention of people attending the SICUREZZA exhibition in Rho, Milan, Italy. The goal of the social robot, housed in the booth of a company selling artificial intelligence products, was to encourage people to enter the booth having a conversation with them, answering specific questions about the company or entertaining people with generic sentences and pleasantries.

In particular, the social robot was asked to greet at the beginning of the conversation, possibly encouraging people to interact if not engaged in other conversations and to say goodbye at the end. The social robot had to give affirmative and negative answers and to answer specific questions about the company, *i.e.* related to partners, sellers, sectors, functionalities, installation constraints, performance, and so on. In addition, the social robot had to focus on its interlocu-

tor without considering other people and be able to personalize the conversation based on the recognized identity, gender, and age.

This section describes the social robot that has been implemented according to the proposed architecture and that meets all the application requirements.

2.3.1 Sensors and actuators

The social robot has been built on Pepper from SoftBank Robotics. Pepper is a humanoid robot which has been designed for social interaction, with the aim to intrigue and attract people and customers (Figure 2.6). Instead of using its own hardware, as shown in Figure 2.7, an Intel Real Sense D435 camera and a ReSpeaker Microphone Array v2.0 has been mounted on its head. The former is a RGB and depth camera that, thanks to the large field of view and the global shutter sensor, is one of the most suited solutions for robotics applications. The latter is an embedded microphone array able to dynamically adapt to the background noise and provide a more clean audio stream. This feature is extremely useful in an exhibition, commonly characterized by loud environmental noise.

On the back of the robotic platform is additionally mounted a NVIDIA Jetson Xavier NX embedded system to allow the execution of the various ROS nodes. In particular, the chosen embedded system is equipped with 384 CUDA cores and 48 tensor cores to allow high parallelism while keeping limited power consumption. All the additional equipment required for the processing device has been mounted on the social robot with 3D-printed supports.

Being the video and audio input nodes independent of the specific camera and microphone, only the functionalities for acquiring video and audio stream from these sensors has been implemented. To this aim, the APIs provided by the builders have been adopted. As for

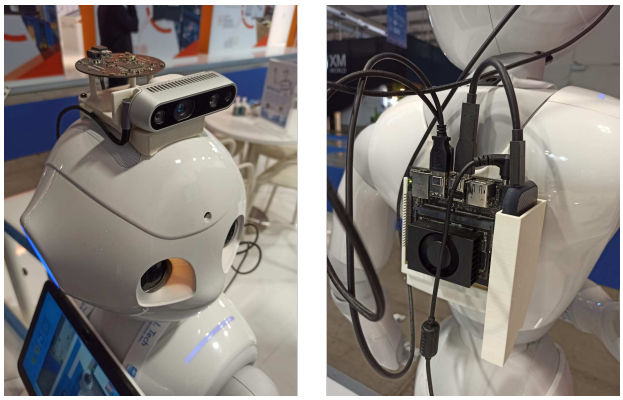


Figure 2.7: The social robot, based on Pepper from Softbank Robotics, was equipped with an Intel Real Sense D435 and a ReSpeaker Microphone Array v2.0, both mounted on its head. The processing device of the social robot is a NVIDIA Jetson Xavier NX mounted on the robot back through a 3D printed support.

the actuators module, I used the official libqi-python library[1] to send voice and movement commands to the social robot. In particular, Pepper allows to use its on-board multilingual speech synthesis system (Italian was chosen since the exhibition was in Italy) in combination with coordinated human-like movements to interact with people; these functionalities have been implemented in the module.

2.3.2 Video and audio analysis

These modules rely on the solutions available in the literature which allow to obtain a good trade-off between accuracy and computational load.

In the face detection node, UltraFace [150], a lightweight neural

[1] https://github.com/aldebaran/libqi-python

network designed for edge computing devices, has been adopted. It is pre-trained on the popular Widerface dataset, on which it achieves a remarkable accuracy. The size of the model is around 1MB, requiring only 90-100 MFlops.

Each detected face is given as input to the soft biometrics recognition node, in charge to perform gender recognition, age estimation and emotion recognition with neural networks efficient and robust to the image variations happening in real environments (blur, noise, brightness). In particular, two MobileNet v3 proposed for gender recognition in [151] and for age estimation in [152] have been chosen because they demonstrated robustness to image corruptions. As for emotion recognition, a ResNet-18 solutions designed in [23] for social robots has been considered.

The neural network adopted in the object detection node is a version of YOLO-X [153] pre-trained on the COCO dataset, which currently holds one of the best performance on this benchmark despite the reduced number of parameters and the high processing speed.

For both face and object tracking the algorithm defined in [154] has been adopted; it is an overlap tracking method achieving a good trade-off between accuracy and computational complexity.

In the Sound Source Localization node, the proposed social robot relies on the algorithm already available in the ReSpeaker Microphone Array v2.0, which is optimized for determining the speech direction of arrival and is effective even when the speaker is far from the microphone. The algorithms are not publicly available and the underlying methodology is not known. In addition, this choice allows to reduce the computational load on the processing device, since the algorithm is executed on board the microphone.

The WebRTC [155] has been chosen to be used in the voice activity detection node since this model was specifically developed by Google to run in real-time on devices with reduced computational capabilities. The algorithm is extremely fast and effective in detecting the voice,

even if it is covered or corrupted by the environmental noise; this feature is crucial in crowded environments, since the signal-to-noise ratio (SNR) may be very low.

The Speech Recognition module is based on the state-of-the-art Conformer architecture [156]. The neural network combines multi-head attention and convolutions to analyse both local and long-range dependencies. In particular, I fine-tuned the small model (small, medium, and large are available) for the Italian language to obtain a good trade-off between speed and transcription accuracy.

2.3.3 Multimodal sensors aggregation

The multimodal sensors aggregation node is implemented in the social robot application by following the guidelines reported in Section 2.2.3. It takes into account the position of the user computed by the sound source localization node and by the face-tracking algorithm. The aggregation follows the formula reported in Equation 2.4. The sound source localization algorithm gives the position on the horizontal plane; hence, only that component of the position is updated when a speech sample is acquired. This implementation relies on the assumption that, during an interaction, the distance between the robot and the interlocutor would not change.

2.3.4 Multimodal re-identification

For the purposes of the exhibition, the social robot has been provided with the capability to re-identify the people it already met. To this aim, the multimodal re-identification node receives the embeddings of each face identified by the multimodal sensor aggregation module and stores a set of face embeddings with a numerical identifier. Because of memory constraints, the maximum number of embeddings stored for each identifier is set to 3, which allows to obtain a good trade-off

between accuracy and memory occupancy. At this point, the node computes the cosine similarity between the acquired embedding and the ones in the gallery.

$$similarity(\bar{a}, \bar{b}) = \frac{\bar{a} \cdot \bar{b}}{||\bar{a}||\ ||\bar{b}||} \tag{2.6}$$

Then, it associates the analyzed embedding with the identity obtaining the highest average similarity (*i.e.* average between the multiple embeddings associated with the same identity); if the similarity is lower than a threshold (0.78 in this case), the face is associated with a new identity. The threshold has been set on a private validation set.

In this application context, the expiration time for the embeddings has been set to one hour for people who are not members of the company (uploaded in the system before the exhibition). In this way, it is possible to personalize the interaction with the people already known by the social robot. More details about this mechanism are reported in the following section.

2.3.5 Behaviour manager

To design the behaviour manager I considered that the social robot has to accomplish the following tasks: (i) attract people to the booth; (ii) interact through spoken natural language with a person avoiding to be distracted by other people or the environmental noise; (iii) interrupt the conversation if the interlocutor says goodbye or leaves. To this aim, I modeled the behavior manager node as a FSM (depicted in Figure 2.8), according to the state design pattern.

At the start, the social robot is in a *wait* state where it remains until it is called upon by the wakeword "Pepper" or detects the presence of a "new" person within 2 meters through the use of RGB camera characteristics and facial size in the captured image. In the first case, the social robot moves to the state *interact* and answers according

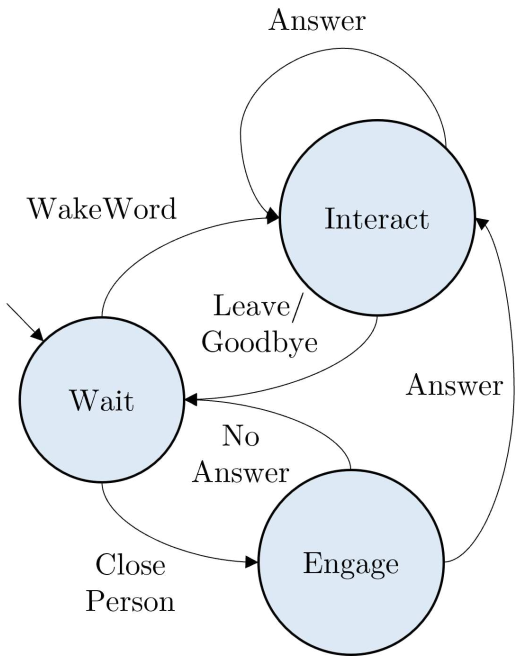


Figure 2.8: Finite state machine implemented in the behaviour manager node. It consists of three states: wait, engage, and interact. In the first state, the social robot waits that a person is nearby to start an interaction; as soon as a person is closer than 2 meters to it, the robot moves into the second state and tries to start a conversation. If the person answers to the robot, it goes in the third state, otherwise it returns in the wait state. The interaction can be also started by a person through a wakeword. Finally, when a person says goodbye or he/she leaves the scene, the robot comes back to the wait state.

to the input sentence; otherwise, the social robot changes its state into *engage* and tries to attract the attention of the person with the aim to start an interaction. In the *engage* state, if the person does not answer the robot, it returns in the *wait* state. To prevent the social robot from being annoying with known people, the multimodal re-identification node is exploited to avoid to proactively start an interaction (*i.e.* change the state from *wait* to *engage*) with known people, *i.e.* belonging to the booth of the company or with whom the social robot already interacted in the previous hour (due to the expiration time). To this aim, in the *interact* state the behaviour manager stores an anonymous identity for the current interlocutor if it does not exist already; otherwise, the expiration time is updated. During the *interact* state, the robot focuses on its interlocutor by keeping the eye contact with him/her (*i.e.* follows the interlocutor's face).

The algorithm that allows to follow the interlocutor computes the angular distance between the position of the face in the image and the center of the image and moves the head of the social robot accordingly. The same interlocutor can be followed, thanks to the possibility of re-identifying a person through his/her facial attributes. The information related to each interlocutor are anonymized through numeric identifiers. Moreover, when one hour is passed from the interaction (*i.e.* the time configured for the social robot to avoid the interaction with the same person) the embeddings representing the expired known interlocutor are deleted. Finally, in the *interact* state, if the interlocutor says goodbye or does not talk with the social robot for 30 seconds, it returns in the *wait* state.

2.3.6 Dialogue manager

The dialogue manager node has been designed with the aim to support the intents and the corresponding entities that the social robot must be able to recognize. According to the requirements of the application,

first of all, I defined intents and entities, which are summarized in Table 2.1.

The ultimate goal of the social robot is to effectively communicate and provide information to its human interlocutors. This involves the robot's ability to comprehend common conversation intents such as greetings, goodbyes, affirmations, and denials. Additionally, the robot must be able to explain its functions so that humans are better informed on what they can ask of it. Another key aspect of the robot's functionality is its ability to understand and answer questions pertaining to the company, including its description and history, partners, selling sectors, products, reliability, and purchasing procedures. The robot can recognize and respond to entities such as partner names, product names, and selling sectors, which allows it to provide more detailed information to those with a specific interest. The dialogue manager node contains the correct information for each combination of intent and entities, enabling the robot to provide accurate and appropriate answers. Finally, the social robot is able to manage a generic intent (OutOfIntent) not related to the ones presented above; in that case, the robot extracts the entities within the sentence through the Spacy library[2] and recognizes the subject of the sentence. Then, it retrieves the information that best matches the input utterance by sending a query to the I/O service manager node, detailed in 2.3.7. If no entities are recognized, the robot says that it is not able to manage that intent and presents again its functionalities.

The dialogue manager node has been implemented through the RASA[3] conversational AI framework, which provides state-of-the-art pipelines for natural language processing. The model adopted for natural language understanding is based on multitask transformers [157]. With this neural network, it is possible to address the tasks of intent

[2]https://spacy.io/
[3]https://rasa.com/

Table 2.1: Intents and entities managed by the social robot. For each entity type, the number of distinct values that it can assume is also reported.

Intents	**Entities**	
Name	**Name**	**Values count**
Greet	Partners	21
Goodbye	Products	18
Affirm	Sectors	4
Deny		
AskPartners		
AskCompany		
AskProducts		
AskSectors		
AskSellers		
AskReliability		
AskInstallation		
AskFunctions		
OutOfIntent		

and entity recognition with a model of limited size. The use of a single model, moreover, allows to speedup the inference time and, therefore, a more natural spoken interaction. Also, the dialogue policy component has been implemented through the RASA framework [158]. In particular, a transformer has been adapted to encode the proposed conversational representation and retrieve the most-suited answer for the specific user the social robot is interacting with.

2.3.7 I/O services manager

To manage general questions from the users that are not related to the company and its products, the I/O services manager node provides the social robot access to the Wikidata knowledge base. In particular, starting from the utterance subject extracted by the dialogue manager node, the I/O services manager node accesses its Wikipedia page and retrieves the related summary description. In this way, the social robot has the capability to answer, even if in a simple and not always an accurate way, to questions on each topic and to entertain the interlocutor in a generic (chit-chat) conversation.

2.4 Human validation

In this section, I report the results of the validation of the realized social robot application. As discussed in Chapter1, performance evaluation is one of the biggest challenges in this research field.

To this purpose, it has been asked to 161 of the people who interacted with the social robot during the exhibition to evaluate their HRI experience. In the following paragraph, it is described the survey that has been submitted to the users via the tablet of the social robot, while in Section 2.4.2 the results of the evaluation are reported and commented.

2.4.1 Survey description

The survey that has been submitted to the users interacting with the social robot has been designed with the two-fold purpose to evaluate its capabilities and to outline possible future improvements. It consists of only five questions, in order to avoid annoying people (that may thus answer without reading the questions to quickly complete the survey) and to obtain reliable answers.

The five questions are reported below:

- *D1*: How engaging was the conversation with the robot?
- *D2*: How do you rate the ability of the social robot to understand the questions?
- *D3*: How fast was the response of the social robot?
- *D4*: How appropriate were the answers of the social robot?
- *D5*: How would you rate the social robot overall (reliability, performance)?

The answer to *D1* gives an important feedback about the design of the dialogue system because it evaluates the capability of the social robot to engage a natural conversation with the human. The answer to *D2* allows to judge the effectiveness of the audio analysis and of the dialogue manager node, especially the accuracy of the voice activity detection, the speech recognition, and the natural language understanding algorithms. The answer to *D3* provides us with an useful estimation of the latency perceived by the user, that can be crucial for a positive experience with the social robot. The answer to *D4* evaluates the performance of the video analysis module to extract the soft biometrics used for the personalization of the conversation, as well as the ability of the dialogue manager to recognize intents and entities.

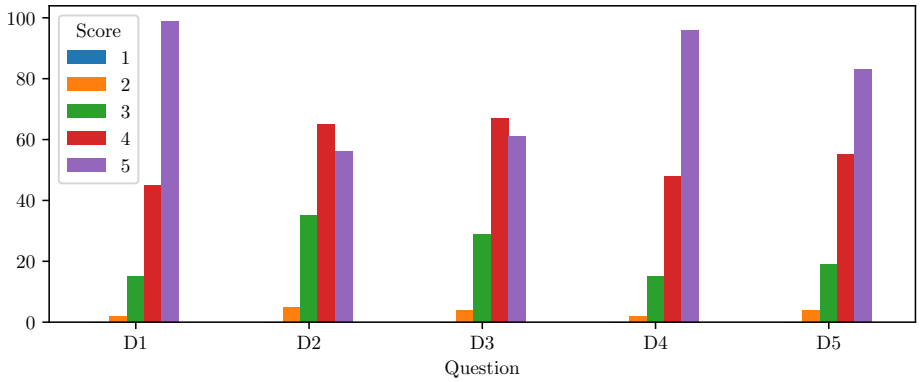


Figure 2.9: Histogram of the answers to the survey evaluating the user experience. The chart represents the distribution of the scores for each question obtained through the survey. The scores vary in the range from 1 to 5.

Finally, the answer to *D5* allows to have an overall evaluation of the HRI experience.

For each question, the subject can assign a score from 1 to 5; the higher the score assigned, the more positive the judgment on that particular aspect.

2.4.2 Results

The results of the survey are reported in Figure 2.9, which represents the distribution of the answers given by the users in the form of a histogram for each question. The values from 1 to 5 are represented from left to right. The higher the bar, the greater the number of occurrences of the corresponding score for that question.

It can be noted that all the histograms have their maximum in the right part of the chart (score equal to 4 or 5), where, in general, most of

Table 2.2: Summary of the answers to the survey. For each question (from D1 to D5) the following statistics are computed: mean, standard deviation (std), min, max and quartiles (25%, 50% and 75%).

	D1	**D2**	**D3**	**D4**	**D5**
mean	4.50	4.07	4.15	4.48	4.35
std	0.72	0.83	0.80	0.72	0.78
min	2	2	2	2	2
25%	4	4	4	4	4
50%	5	4	4	5	5
75%	5	5	5	5	5
max	5	5	5	5	5

the values are concentrated; it means that for the majority of the users the experience was positive. This is an impressive result if considering that the evaluation has been carried out into a very challenging environment, characterized by loud background noise. In addition, the people who attend the SICUREZZA exhibition are technically skilled and have a good knowledge of computer vision and real-time systems; this consideration allows us to exclude a bias towards positive answers due to the "enthusiasm" related to the interaction with a social robot.

The results can give further insights if a more detailed analysis of the answers is performed. Detailed statistics are reported in Table 2.2. It can be noted as there are no answers with score 1; the minimum value is 2 for each question, while the average value is always greater than 4. For each question, more than 75% of the users rated the experience with a score greater than or equal to 4, while more than 25% of them gave the maximum score.

Comparing the results obtained by the different questions, one can note that the scores for questions *D1*, *D4* and *D5* have a mode (*i.e.* the most answered score) equal to 5; on the other hand, questions *D2*

and *D3* obtained a mode equal to 4. The same trend can obviously be observed in the average values, which are slightly lower for *D2* and *D3*, although always higher than 4. The result is reasonable, considering that the noisy and crowded environment constitutes a severe challenge for the audio analysis node.

To better investigate the performance of the two main components involved in *D2*, namely the speech recognition node and the dialogue manager node, a log of the transcriptions of the speech recognition node and of the results of the corresponding intent and entity recognition has been collected during the exhibition; in addition, visual feedback on the tablet of the social robot has been added with the transcription of the speech recognition node in order to have an immediate qualitative result. In this way, I collected all the elements necessary to understand if the cause of a wrong understanding was an error of the speech recognition node or of the dialog manager node. This analysis results in the fact that most of the errors were due to wrong transcriptions by the speech recognition node; this issue was commonly due to people from the booth that were talking with customers just behind the social robot and, therefore, added a lot of noise to the speech signal acquired by the microphone.

In conclusion, the experiment proved the effectiveness of the proposed architecture and the implemented social robot, which obtained a very positive rate from a skilled pool of people. Moreover, the result of the survey allowed to identify the nodes that can be further improved and deserve more attention in real environments: first, audio models have to be more robust to environmental background noise, focusing only on the interesting sounds; moreover, the perception pipeline has to be further improved to obtain instantaneous awareness of surroundings. These goals will be the main subject of the following chapters.

Chapter 3

Robust and lightweight audio representation learning

Based on

DENet: a deep architecture for audio surveillance applications.
Greco, A., Roberto, A., Saggese, A., & Vento, M. - Neural Computing and Applications, 2021.

DEGramNet: Effective audio analysis based on a fully learnable time-frequency representation
Foggia, P., Greco, A., Roberto, A., Saggese, A., & Vento, M. - SUBMITTED TO Neural Computing And Applications.

3.1 Background

At the light of the results obtained in Chapter 2 and the analysis of the state of the art carried out in Chapter 1.3.2, in this chapter I aim to design an audio deep learning model robust to environmental noise and able to run in real-time on social robots' embedded devices.

To this aim, I start my analysis from SincNet [159], a raw-waveform CNN architecture for speaker recognition tasks. The main characteristic of SincNet is that it substitutes the first convolutional layer with a trainable band-pass filterbank. This choice results in the ability to learn the frequencies-of-interest (FOI) for the considered sounds. The capabilities of SincNet are particularly important for the purposes of the thesis, in fact it is possible for the model to ignore all sounds that are distributed out of the learned frequencies. Moreover, the use of parametric filters drastically reduces the number of parameters to train if compared to standard convolutional layers and, therefore, allows to generalize well even when dealing with small datasets.

However, although SincNet learns the most relevant frequencies for recognizing the events of interest, it has not the capability to identify overlapping noise in the learned frequencies. For this reason, in Section 3.2 I propose an attention branch called Denoising-Enhancement Layer (DELayer) to overcome the identified limitation. The DELayer

dynamically determines the frequency components to attenuate and the ones to amplify to improve the classification in presence of overlapping background noise. To validate the proposed layer DENet is proposed, an end-to-end raw-waveform model for Sound Event Detection in surveillance applications. The particular task has been chosen considering that in social robotics applications, the audio signals can be very noisy due to various factors, such as background noise, overlapping sounds, and reverberation. This is where surveillance sound event detection benchmarks can be particularly useful. These benchmarks provide standardized datasets with realistic background noise and a range of common sound events, allowing researchers to evaluate the robustness and accuracy of their models in challenging real-world scenarios. The proposed model achieved an improvement in terms of recognition rate w.r.t. the SincNet counterpart of about 5% at a very low SNR (5dB) proving its robustness to background noise.

The robustness of the proposed DENet comes with additional computational requirements requiring the model to batch multiple frames to work in real-time on CPU processors. Having assessed the capabilities of the proposed DELayer, we proposed a spectrogram-based alternative, namely DEGram, to overcome this limitation. DEGram is a learnable audio representation layer composed of two components: SincGram and Time-Frequency DELayer. The first component, as SincNet, allows to learn the frequencies of interest in the frequency domain thanks to a novel formulation of the layer. The latter component inherits the capabilities of the DELayer while drastically reducing the computational requirements and applying the attention mechanism also in the time domain. Being a spectrogram-like representation, it is also possible to exploit shallow CNN architecture (as previously discussed in Chapter 1.3.2) and, therefore, to further reduce the computational requirements. The proposed representation achieved state-of-the-art performance on the VGGSound dataset (for Sound Event Classification) and comparable accuracy with a complex

and special-purpose approach based on network architecture search over the VoxCeleb dataset (for Speaker Identification). Moreover, DEGram allowed to achieve high accuracy with lightweight neural networks that can be used in real-time on embedded systems, making the solution suitable for Social Robotics applications.

3.2 Noise robustness in end-to-end convolutional networks

3.2.1 Methodology

The architecture of the proposed system is shown in Figure 3.1. The audio stream is partitioned into frames, which are in turn grouped in overlapping sequences (Figure 3.1a). Hereinafter, I will refer at the $j-th$ frame of the $i-th$ sequence with x_{ij}. After the sequencing stage (Section 3.2.1.1), the frames of each sequence are classified by using a *many-to-many* bidirectional recurrent model, called Denoising-Enhancement Network (DENet). The network takes as input directly the raw waveform. The architectural details of the proposed network are reported in Table 3.1 and Table 3.2. The overall architecture, in terms of convolutional and dense layers has been inherited by SincNet [159]. Anyway, with respect to the original architecture, I introduced the novel DELayer, with the aim to increase the discriminative power of the architecture; moreover, I have also added a BGRU for explicitly evaluating the temporal information.

In the first layer, DENet filters the most important frequencies for the task that it is dealing with. Then, through the proposed DELayer, it is able to *dynamically* identify and filter that bands which do not contain noise and, therefore, are useful for the classification, by independently analyzing each one through an attention neural network. After enhancing and denoising the signal, the *time-local* feature extrac-

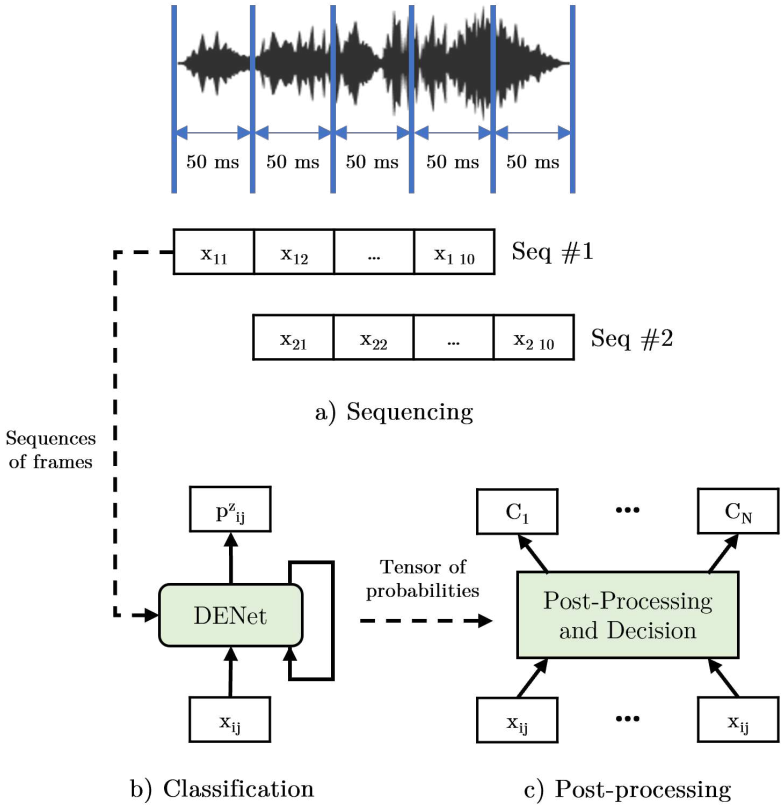


Figure 3.1: System architecture. The input audio signal is divided into non-overlapped frames of size 50 ms; a sequence is composed by 10 frames (a). Each sequence is shifted of 1 frame w.r.t. the previous one. Then, DENet (b) computes the probabilities p_{ij}^{z} of each class z for each frame belonging to the analyzed sequence (many-to-many classification). Finally, a median decision filter (c) is applied over the class labels of the N extracted frames; the class probability of each frame is computed by averaging all the classifications of the overlapped sequences.

Table 3.1: DENet Convolutional Architecture. In the *Info* column the report the parameters of each layer, *i.e.* the number and the length of the convolutional and pooling filters (Filters - Len), the dropout rate (Rate), and the number of units (Units) in the BGRU and fully connected layers.

Layer	**Info**
SincNet Layer	Filters: 80 - Len: 251
Denoising-Enhancement Layer	Table 3.2
Max Pooling	Len: 3
Convolutional Layer	Filters: 80 - Len: 5
Max Pooling	Len: 3
Spatial Dropout	Rate: 0.1
Normalization Layer	
Convolutional Layer	Filters: 80 - Len: 5
Max Pooling	Len: 3
Spatial Dropout	Rate: 0.1
Normalization Layer	
Bidirectional GRU Layer	Units: 2048
Dropout	Rate: 0.3
Fully connected	Units: 1024
Dropout	Rate: 0.4
Fully connected	Units: 512
Dropout	Rate: 0.3
SoftMax Layer	

Table 3.2: DELayer - Attention branch operation. In the *Info* column the parameters of each layer are reported, *i.e.* the number and the length of the convolutional and pooling filters (Filters - Len) and the number of units (Units) in the fully connected layers.

Layer	**Info**
Convolutional Layer	Filters: 30 - Len: 7
Convolutional Layer	Filters: 30 - Len: 7
Convolutional Layer	Filters: 10 - Len: 7
Fully connected	Units: 128
Fully connected	Units: 64
Fully connected	Units: 1

tion (at frame level) is performed by two additional one-dimensional convolutional layers followed by a MaxPooling.

In order to fully exploit the temporal information behind the audio stream, I stacked after the convolutional part of DENet a Bidirectional Gated Recurrent Unit which performs the extraction of the *time-global* features starting from the time-local ones of each frame in the sequence. The class probabilities of each frame in the analyzed sequence are computed by a fully connected neural network. Given the $i-th$ sequence, the output of the network consists in a matrix of probabilities, in which each element p_{ij}^{z} corresponds to the probability that the $j-th$ frame belongs to the class z (Figure 3.1b). Finally, the event-label $C_{1...N}$ of each of the N frames extracted from the audio stream is predicted by post-processing the probabilities resulting from the network (Figure 3.1c).

Following on the previously described architecture, mainly three stages in the proposed system can be identified: *sequencing* (Section 3.2.1.1), *classification* (Section 3.2.1.2) and, *post-processing* (Section 3.2.1.3). Each stage is extensively analyzed in the following sections.

3.2.1.1 Sequencing stage

During a preliminary *sequencing* step (Figure 3.1a), the audio stream acquired by the microphone is partitioned into non-overlapped frames of duration 50 ms. Then, we use the Hamming windowing function to smooth the frames in order to avoid abrupt changes at the boundaries that can cause distortions in the spectrum.

To take into account the temporal evolution of the audio signal, the input of our recurrent neural network is a sequence of 10 frames. The audio sequences are obtained by applying a sliding window of 500 ms with a shift of 50 ms, namely 1 frame. This setting has been proved in [160] to be a good choice for the considered classes of events; using a shift smaller than the size of the window, in fact, it is possible to detect those events localized at the boundaries more effectively.

3.2.1.2 Classification

In this subsection, the main building layers of the proposed DENet are presented. For the sake of clarity, I show in Figure 3.2 the reactions of the proposed layer to samples belonging to different categories of audio events. Such samples belong to the test set of the MIVIA Audio Events dataset, used in this chapter for benchmarking purposes. More details about the dataset and the training procedure will be presented in Section 3.2.2.1 and 3.2.2.3, respectively.

SINCNET

Each audio sound is characterized by a spectrum which describes the distribution of the power over the frequencies. It is almost always true that the energy of a sound is located in a restricted part of the spectrum. Therefore, these frequencies, that we have previously called frequencies of interest (FOI), are crucial to detect and to distinguish the analyzed events; on the other hand, the rest of the spectrum is a

source of noise for the classification stage.

The aim of the SincNet Convolutional Layer, recently proposed in [159], is to extract time-local feature maps which represent the FOI. Differently from state-of-the-art CNN layers, in which all the kernel weights have to be learned, the SincNet layer is parametric. In particular, each filter consists of a sinc function in which only the cut-off frequencies are learned from the data. More formally, the first convolutional layer can be defined as follows:

$$x'_{bij} = x_{ij} * g_b[f_1, f_2] \tag{3.1}$$

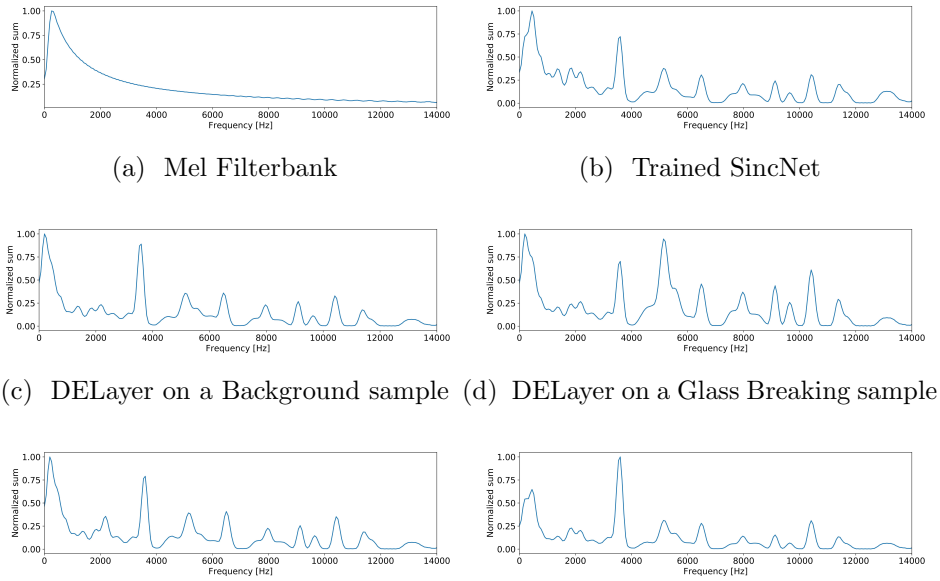
$$g_b[f_1, f_2] = 2f_2 sinc(f_2) - 2f_1 sinc(f_1) \tag{3.2}$$

where x'_{bij} is the filtered output, hereinafter component, g_b is the $b-th$ pass-band function that depends from the cut-off frequencies f_1 and f_2, and $sinc(f_1)$ and $sinc(f_2)$ represent two sinc signals centered in f_1 and f_2, respectively.

As proposed by the authors, I initialize the cut-off frequencies of SincNet according to the Mel-scale filter-bank (Figure 3.2a). This scale has been chosen instead of a random initialization since it demonstrated to reproduce the non-linear human ear perception of sound; it is more discriminative at lower frequencies and less at higher frequencies.

Starting from these weights, this layer is able to learn the FOI directly from the raw data of the class of events we are interested for our application, without the need for any intermediate frequency-based representation. In addition, the reduced number of parameters to train allows to avoid overfitting even with small datasets; this is a very important and not negligible feature, especially in the case of audio analysis, since the size of the available datasets is typically quite limited.

Differently from the original layer, in the proposed model I add a L2 regularization factor to the bandwidths of the SincNet layer. In



(a) Mel Filterbank

(b) Trained SincNet

(c) DELayer on a Background sample

(d) DELayer on a Glass Breaking sample

(e) DELayer on a Gunshot sample

(f) DELayer on a Scream sample

Figure 3.2: Comparison between the cumulative frequency responses (*i.e.* the normalized sum of the amplitudes of the filters) of the Mel Filterbank (a), the SincNet Layer (b) after the training phase, and the outputs of the DELayer applied on Background (c), Glass Breaking (d), Gunshot (e), and Scream (f) samples. The attention in the DELayer assigns different weights to the components at the various frequencies, in order to attenuate the noisy ones (denoising) and amplify those relevant for the classification (enhancement).

this way, I encourage the learning of filters strictly centered in the frequencies most relevant for the classification, thus reducing the possibility of overfitting. This aspect is clearly evident in Figure 3.2b: the cumulative frequency response of the SincNet layer after the training phase demonstrates that, differently from the Mel initialization, there are different and strict peaks centered in the FOI.

DENOISING-ENHANCEMENT LAYER

SincNet allows to automatically learn the FOI, but it is not able to adapt the weights of each band-pass filter (*i.e.* the relevance of each frequency) to deal with the overlapping noise typically present in real environments. Starting from this consideration, I propose a new attention layer that takes as input the output of the SincNet layer and compute at runtime (*i.e.* for each frame) a vector of weights to assign to the components at the various frequencies. From now on we will refer to the weights vector as *attention map*.

In particular, the proposed layer dynamically determines the components affected by noise to attenuate with a small weight (denoising) and the components relevant for the classification to amplify with a bigger weight (enhancement); for this reason, I named it *Denoising-Enhancement Layer*, hereinafter *DELayer*. The effect of the DELayer applied on Background (B), Glass Breaking (GB), Gunshot (GS) and Scream (S) samples, is clearly evident in Fig 3.2c, 3.2d, 3.2e, and 3.2f; it can be noted that the cumulative frequency response of the trained SincNet is quite different from the ones obtained after the application of the DELayer, since the learned attention maps attenuate or amplify the components at the various frequencies for each sample. Moreover, the layer has a different response for each class of the input signal.

In more detail, the output of the SincNet layer consists of several band-pass filtered versions of the input signal. For each component, an attention neural network (*attention branch*) is used to compute a non-

negative weight c_{bij}; the result for the whole spectrum is an attention map, which has a weight c_{bij} for each component. Then, a softmax normalization is applied over all the estimated weights; in this way, the resulting sum is equal to 1.

Finally, the output of the layer is computed as the sum of its inputs, weighted through the attention map.

Formally, the DELayer can be described as follows:

$$y_{ij} = \sum_{b=1}^{B} x'_{bij} \; w(x'_{bij}) \tag{3.3}$$

where y_{ij} represents the output signal, B is the number of sinc filters (*i.e.* the number of extracted signals) in the previous layer, and w represents the weighting function based on the attention branch output, defined as follows:

$$w(x'_{bij}) = softmax(c_{bij}) \tag{3.4}$$

$$softmax(\alpha_z) = \frac{\exp(\alpha_z)}{\sum_{\alpha_k \in \overline{\alpha}} \exp(\alpha_k)} \tag{3.5}$$

where α_z represents the $z-th$ value of the attention map $\overline{\alpha}$. In this particular case, $\overline{\alpha}$ is composed by the positive coefficients computed by the attention branch for each input component x'_{bij}. For the sake of readability, the dependency between $w(x'_{bij})$ and $x'_{kij}, \forall k \neq b$ has not been reported in both equations.

The rationale of this procedure can be traced back to the Fourier series when generalizing the sum from single frequency to frequency band signals. DENet considers the input signal as the sum of the event of interest with an additive random noise; this assumption is reasonable in the case of audio inputs in which the background noise is overlapped with the events to detect. According to this observation, the proposed training procedure allows the DELayer to learn

how to dynamically compute the Fourier coefficients necessary for reconstructing the most important frequency components of the event of interest.

BIDIRECTIONAL GATED RECURRENT UNIT

SincNet and DELayer allow to learn time-local features, namely these layers independently analyze each audio frame. To consider the temporal evolution of the audio signal, a *Bidirectional GRU* is stacked after the convolutional layers of the proposed DENet.

Working at the feature level, DENet is able to learn a new temporal representation which takes into account the evolution over time of the components at the FOI. In addition, the bidirectional analysis of the audio stream allows the output layer to get information from the previous and from the next states simultaneously. This is particularly important when dealing with audio events of different duration, *e.g.* gunshots, and screams. In this way, DENet drastically reduces the misclassifications that happen at the beginning of the event-of-interest. These errors are mainly due to the partial knowledge about the temporal evolution of the events to detect. Practically, the bidirectional process acts as a sort of delayed reasoning after the event occurrence and can improve the classification of all the frames overlapping with the target.

Furthermore, I chose a GRU instead of a LSTM in order to reduce the complexity of the proposed model. In fact, this module is characterized by fewer parameters and allows at the same time to reduce the number of operations and increase the generalization capability when dealing with small datasets. These two advantages are important in our case since we need to perform the analysis in real-time and we have a reduced number of available training samples.

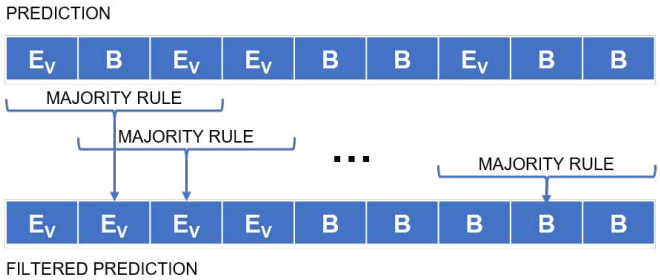


Figure 3.3: Median decision filter applied to the predictions over the frames at binary-class level (E_V event, B background). It applies a sliding majority rule to remove spurious misclassifications.

3.2.1.3 Post-processing stage

At this stage of our system, the output of DENet is a tensor of probabilities in which each value p_{ij}^{z} represents the probability that the frame j in the sequence i belongs to the class z. Due to the overlap between consecutive sequences, for each frame, we have multiple different predictions. In order to exploit the redundant information, an aggregation function has been applied to all the classifications referring to the same frame to obtain a single class label. In particular, the system makes a class-wise average of the probabilities and chose the label with the highest probability.

At the end, to mitigate the occurrences of single-frame False Positive (mainly due to very short events, like gunshots) and False Negative, I adopt the median filter depicted in Figure 3.3 over the decisions taken for each class (binary-class filtering). The median decision filter updates the labels by using a sliding majority rule of size 5; the value of this parameter is a good trade-off between the size of the classified frame (*i.e.* 50 ms) and the mean duration of the shortest event (*i.e.* gunshot which has an average length of 440 ms).

The use of a Bidirectional Layer and the mean aggregation function introduces a delay in the decision of just 1 second, which can be considered acceptable for surveillance applications.

3.2.2 Experimental framework

3.2.2.1 Datasets

I conducted the experiments to assess the performance of DENet on the MIVIA Audio Events [161] and the MIVIA Road Events [162] datasets. These databases are widely adopted by the scientific community for benchmarking purposes in the field of audio event detection in surveillance applications.

The MIVIA Audio Events dataset focuses on surveillance applications, thus it includes glass breaking (GB), gunshot (GS), and screams (S). The class Background (B) is also included. More details on the dataset are shown in Table 3.3; it consists of about 30 hours of recording and includes audio files acquired with different SNRs, in the range from 5 dB to 30 dB; in this way, it is possible to verify the robustness of the methods with respect to environmental noise and to different distances of the source from the microphone.

Differently from the previous database, the MIVIA Road Events dataset is focused on audio events for road surveillance applications, containing events such as tire skidding (SK) and car crashes (CC). Also in this dataset, the Background (B) class is included. The audio files are divided into 4 folds containing 100 events each, in order to account for cross-validation experiments. More details about the dataset are reported in Table 3.4.

The recordings are sampled at 32,000 Hz and quantized at 16 bits per PCM sample. Each audio file has a different background so that several different real situations are simulated.

For both the databases, the training and the test sets have already

Table 3.3: Number of events and total duration of background (B), glass breaking (GB), gunshot (G) and scream (S) recordings in the Mivia Audio Events Dataset.

Type	**Training set**		**Test set**	
	Events	**Duration (s)**	**Events**	**Duration (s)**
B	-	$58,372$	-	$25,037$
GB	$4,200$	$6,025$	$1,800$	$2,562$
GS	$4,200$	$1,884$	$1,800$	744
S	$4,200$	$5,489$	$1,800$	$2,445$

Table 3.4: Number of events and total duration of background (B), car crash (CC) and tire skidding (SK) recordings in the Mivia Road Events Dataset.

Type	**Events**	**Duration (s)**
B	-	$2,732$
CC	200	326
SK	200	522

been defined; thus, we randomly divided the training samples from the validation ones with a ratio of 70%-30%.

3.2.2.2 Evaluation metrics

The performance of DENet has been computed by following the experimental protocol proposed in [162].

In particular, the performance indices are useful for the final user in a real application and are, thus, *event-based*. An event is considered recognized if at least one of the frames which overlaps with the event is properly classified. On the other hand, an event is considered missed if all the frames which overlap with the event are classified as background. The remaining events are considered classification errors. Then, these values are normalized by the number of events in the ground truth, so obtaining the Recognition Rate (RR), the Miss Rate (MR), and the Error Rate (ER).

As an important additional performance index, which takes into account the capability of the system to distinguish the events of interest from the background, I also compute the False Positive Rate (FPR). A false positive is an event detected by the network when only a background sound is present. When a false positive is detected in two consecutive frames, only one error is counted because it causes a single alarm in a real surveillance system. Finally, the FPR is computed by normalizing the number of false positives with the total number of background frames.

3.2.2.3 Implementation details

The DENet architecture has been trained by using the well-known RMSprop optimizer, in which each parameter is updated separately. In particular, I have used a learning rate $lr = 10^{-4}$ and built batches of 128 samples. The hyper-parameters have been tuned through a

grid-search on the validation set. I optimized various architectural properties (length of the filters, number of units) and training settings (optimization algorithm, learning rate, batch size).

As for the training strategy, on the MIVIA Audio Events dataset the network is trained from scratch (TS), using the Glorot initialization procedure for convolutional and dense layers, while the Mel-filterbank bands and center frequencies have been used for the SincNet one. The Glorot or Xavier initializer is a standard de facto for CNNs; it computes the weights with a truncated normal distribution with average 0 and standard deviation equal to $\sqrt{\frac{2}{a-b}}$, where a and b are the number of input and output units of the weight tensor. On the other hand, the chosen initialization for SincNet should amplify the frequencies of the audio signal that are suited for retaining the speech and for discarding all the other components, especially the background noise. It does not allow to start from features which perfectly represent the sounds of interest, but the convergence is surely better and easier since the weights are optimized for a similar task.

On the MIVIA Road Events dataset, in order to evaluate the possibility to transfer the learned weights between different audio event detection problems, I trained the DENet both from scratch and with a fine-tuning (FT) procedure. In more detail, in the latter case, I started from the weights learned by DENet on the MIVIA Audio Events dataset and fine tuned it on the training set of the MIVIA Road Events dataset. Therefore, on this dataset I can evaluate also the effect of transfer learning. This is a common choice in computer vision tasks when no wide datasets are available for the problem of interest; indeed, most of the CNNs are trained for a variety of image classification tasks by using the original weights computed on ImageNet, in order to start from an effective representation. In spite of this, it is not a common approach in audio analysis, in which the fine tuning can have a beneficial effect due to the reduced amount of

training data. By adopting this strategy, I expect an increase in performance and a reduction of the convergence time, since the learning procedure of the network starts from a representation of the sound that is already suited for audio event detection and the data can be sufficient for an effective fine-tuning.

3.2.3 Experimental results

The results of the experiments performed on the MIVIA Audio Events dataset are reported in Table 3.5. The proposed DENet achieves a state-of-the-art recognition rate equal to 0.975, overcoming the recent method based on the trainable COPE filters applied on gammatonegrams [103] (0.960), the CNN SoundNet [163] (0.933) and two approaches based on handcrafted features [31] [162] (0.886 and 0.867). It is worth mentioning that also the standard method based on SincNet [160] is able to outperform all the other approaches (0.971). However, the proposed DELayer and the temporal information gathered with the BGRU introduced in this chapter allow to further increase the recognition rate (increase of 0.004) and to reduce the miss rate (0.014 vs 0.019), keeping at the same time the false positive rate unchanged (0.029).

Analyzing the confusion matrices of the classifications performed by SincNet and DENet on the MIVIA Audio Events dataset, reported in Table 3.6, it can be observed that the improvement is mainly achieved on the sounds with a longer duration, namely glass breaking, and scream, for which the miss rate is almost null (while SincNet has a MR of 0.044 on screams). This improvement is slightly paid on gunshot events (miss rate of 0.042) since these are sounds with reduced duration. The experimental evidence demonstrating the effectiveness of the median filter can be seen in Figure 3.4 and Figure 3.5. The reason for this effect is due to the median filter's ability to reduce the false positive rate while only marginally affecting the recognition

Table 3.5: Comparison of the proposed method with the state of the art approaches on the MIVIA Audio Events dataset in terms of Recognition Rate (RR), Miss Rate (MR), Error Rate (ER) and False Positive Rate (FPR). The methods are ordered for descending recognition rate.

Method	**RR**	**MR**	**ER**	**FPR**
$DENet$	**0.975**	0.014	0.011	0.029
$SincNet$ [160]	0.971	0.019	0.010	0.029
$COPE$ [103]	0.960	0.031	0.009	0.043
$SoundNet$ [163]	0.933	0.007	0.060	0.223
$Haar$ [31]	0.886	0.099	0.014	0.014
$HF + BoW + SVM$ [162]	0.867	0.107	0.026	0.031

Table 3.6: Comparison in terms of confusion matrices and miss rate (MR) obtained by applying SincNet and DENet on the MIVIA Audio Events dataset.

		SincNet				*DENet*			
		Detected class				Detected class			
		GB	GS	S	MR	GB	GS	S	MR
True class	GB	0.999	0.000	0.000	0.001	0.997	0.003	0.000	0.000
	GS	0.019	0.963	0.005	0.013	0.018	0.935	0.005	0.042
	S	0.005	0.001	0.950	0.044	0.005	0.002	0.992	0.001

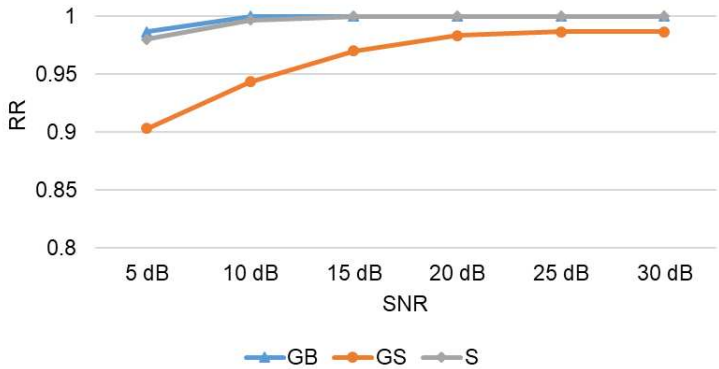
rate. This filtering method may lead to the exclusion of short-duration events, especially in environments with high noise levels.

The proposed architecture demonstrates other two important points of strength for real audio surveillance applications: the robustness to the noise and the generalization capability.

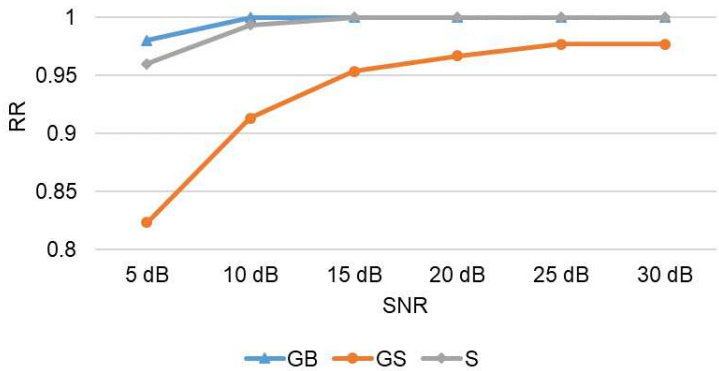
The results reported in Table 3.7 show that DENet is able to better preserve the performance in very noisy conditions; with $SNR = 5\ dB$ it outperforms SincNet in terms of recognition rate (0.921 vs 0.873), miss rate (0.039 vs 0.098) and false positive rate (0.027 vs 0.029). In general, the performance of the proposed network is more stable since its standard deviation among the different SNRs is almost halved in terms of recognition rate (0.025 vs 0.044) and reduced to one-third in terms of miss rate (0.012 and 0.036). This robustness and the stability of the results as the environmental noise changes are definitely crucial capabilities in real audio surveillance applications.

It is essential to recognize that the result obtained holds critical significance within the context of social robotics, the primary topic of discussion in this thesis. It is worth noting that the considered benchmark assumes an uniform distribution of signal-to-noise ratios (SNRs), which may not be the case in actual robotic scenarios where high levels of noise are prevalent. Therefore, developing a model that performs well under these conditions will result in a model that is more effective in real-world scenarios.

As for the generalization capabilities, experimental analysis on a different dataset was necessary. The results achieved by DENet on the MIVIA Road Events dataset are reported in Table 3.8, both in terms of average and standard deviation among 4 experiments, since the experimental protocol requires a 4-fold cross-validation. The performance of the proposed solution on this database are even more encouraging. In fact, the network trained from scratch outperforms almost all the others in terms of recognition rate (0.975), excluding the one based on MobileNet and gammatonegrams proposed in [164]



(a) Without median filter



(b) Median filter window size = 5

Figure 3.4: Effect of the median filter on the recognition rate (RR) for each event class at different SNRs (from 5 to 30 dB). The filter may penalize the recognition rate of the short-duration events, such as the gunshot (GS), especially at small SNRs.

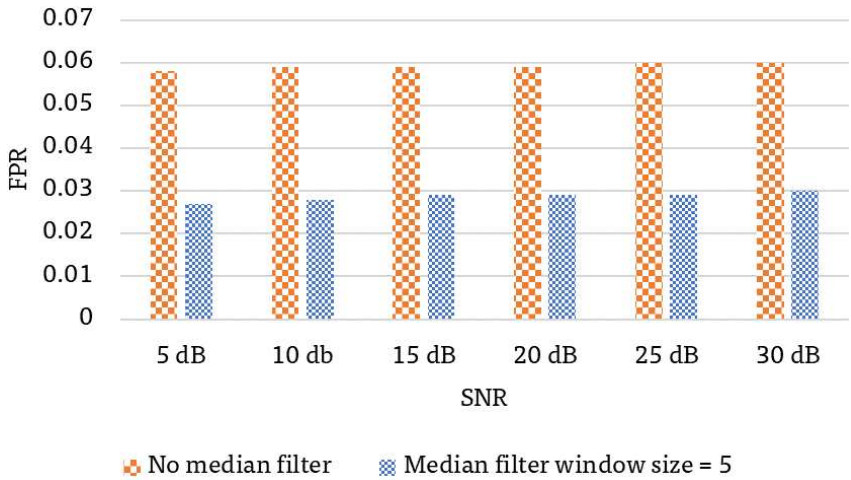


Figure 3.5: Effect of the median filter on the false positive rate (FPR) for each event class at different SNRs (from 5 to 30 dB). The filter allows to reduce the FPR of 50% in all the cases.

Table 3.7: Results of SincNet and DENet on the MIVIA Audio Events dataset at different SNR values (from 5 to 30 dB). In the last two rows I report the mean and the standard deviation of recognition rate (RR), miss rate (MR) and false positive rate (FPR) computer over all the SNRs.

	SincNet			*DENet*		
SNR	**RR**	**MR**	**FPR**	**RR**	**MR**	**FPR**
5 dB	0.873	0.098	0.029	0.921	0.039	0.027
10 dB	0.977	0.016	0.029	0.969	0.020	0.028
15 dB	0.992	0.002	0.029	0.984	0.011	0.029
20 dB	0.994	0.000	0.029	0.989	0.007	0.029
25 dB	0.994	0.000	0.029	0.992	0.004	0.029
30 dB	0.994	0.000	0.029	0.992	0.004	0.030
Mean	0.971	0.019	0.029	0.975	0.014	0.029
Std	0.044	0.036	0.000	0.025	0.012	0.001

(0.995). It is noteworthy the 28.4% of performance increase with respect to the standard method based on SincNet [160] (0.975 vs 0.773), which certifies the effectiveness of the DELayer and of the BGRU for achieving better generalization capabilities. In addition, as expected, the fine-tuning allows to obtain a further improvement of the recognition rate (0.998), so making the DENet trained with this procedure the state of the art even on this dataset (0.003 better than [164]). It is also important to note that DENet achieves on the MIVIA Road Events dataset a null error rate (the others have an error rate of 0.005 [164], 0.012 [103], 0.002 [165], 0.007 [166] and 0.027 [160] in the best cases); it means that each audio event detected is also properly classified.

The sensitivity and accuracy in audio event detection and classification achieved with the adoption of the DELayer and of the BGRU are partially paid in terms of FPR. Indeed, we notice an increase of this index from 0.010 to 0.043 with respect to the original network based on SincNet [160]. However, this modification not only allows to achieve a FPR comparable to that obtained by state-of-the-art methods but implies a substantial increase in the recognition rate (0.975 vs 0.773) bringing at the same time the error rate to 0 (from 0.027). Therefore, such improvements are far superior to the slight increase in sensitivity and, thus, justify and reward the architectural choices.

3.2.3.1 Processing time evaluation

In Table 3.9 is reported the processing time of the proposed network as a function of the buffer size, *i.e.* the number of sequences simultaneously classified. In particular, I evaluated the system with 1, 5, 10, and 20 sequences. We averaged the time required to predict the event probabilities over 100 evaluations for each batch size. We conducted the experiments on both CPU and GPU architectures, an Intel(R) Xeon(R) W-2133 CPU with 62 GB of RAM memory and a Nvidia Titan X Pascal GPU with 12 GB of dedicated memory, respectively.

Table 3.8: Comparison of the proposed method with other state of the art approaches on the MIVIA Road Events dataset. The methods are ordered for descending mean recognition rate (RR).

Method		**RR**	**MR**	**ER**	**FPR**
$DENet(FT)$	Mean	0.998	0.002	0.000	0.043
	Std	0.004	0.004	0.000	0.016
$MobileNet(FT)$ [164]	Mean	0.995	0.000	0.005	0.037
	Std	-	-	-	-
$DENet(TS)$	Mean	0.975	0.025	0.000	0.021
	Std	0.026	0.026	0.000	0.010
$MobileNet(TS)$ [164]	Mean	0.965	0.010	0.028	0.067
	Std	-	-	-	-
$COPE$ [103]	Mean	0.940	0.048	0.012	0.040
	Std	0.043	0.049	0.013	0.018
bof [165]	Mean	0.820	0.178	0.002	0.029
	Std	0.078	0.081	1.000	0.025
bof_{MFCC} [166]	Mean	0.803	0.190	0.007	0.077
	Std	0.116	0.116	0.010	0.059
$SincNet$ [160]	Mean	0.773	0.200	0.027	0.010
	Std	0.080	0.073	0.029	0.008

Table 3.9: Processing time (seconds) of the proposed method on both CPU and GPU. The CPU is an Intel(R) Xeon(R) W-2133 CPU @ 3.60 GHz with 62 GB of RAM memory, while the GPU is a Nvidia Titan X Pascal GPU with 12 GB of dedicated memory.

	CPU		*GPU*	
Buffer Size	**Total**	**Frame**	**Total**	**Frame**
1	0.1135	0.1135	0.0697	0.0697
5	0.1821	0.0364	0.0710	0.0142
10	0.2609	0.0261	0.0753	0.0075
20	0.4217	0.0211	0.0887	0.0044

Considering a shift equal to 50 ms between two following sequences, the system must conclude the processing of the previous buffer within this time interval; however, the higher the size of the buffer, the higher the classification delay. It can be observed that the best trade-off for respecting the real-time constraint is obtained with a buffer size equal to 5 for both the GPU and CPU architectures; with this setting, a waiting time of 0.25 seconds, obtained by multiplying the shift time (50 ms) with the buffer size (5), is negligible for surveillance applications. However, this is not true in the context of Social Robotics.

3.3 Lightweight audio convolutional networks

3.3.1 Methodology

The architecture of the proposed method is depicted in Figure 3.6. In the first layer, the spectrogram is extracted from the input audio signal. Then the DEGram representation allows to select the more rel-

evant frequency bands from the spectrogram through the SincGram learnable filters; afterward, for each audio sample the attention mechanisms attenuate the additive background noise and amplify the useful time-frequency components (more details are reported in Section 3.3.1.1). Once the time-frequency representation has been computed, standard Convolutional Neural Networks and/or Recurrent Neural Networks can be used for addressing a specific audio analysis task. In the proposed DEGramNet (detailed in Section 3.3.1.2) the ECA-Net architecture [167] has been chosen as CNN for the audio classification task.

3.3.1.1 DEGram: Denoising-Enhancement Spectrogram

The DEGram representation results from two layers: the former, called SincGram, learns from the raw data the relevant frequency bands for the audio analysis task at hand; the latter, namely TF-DELayer, applies a time-frequency attention module to dynamically remove additive background noise and enhance those time-frequency components of the input representation useful for the classification.

SINCGRAM

The SincGram layer is composed of M band-pass filters that can be learned during the training process and are represented in the frequency domain. In this way, the layer can learn which are the relevant frequency bands for the specific audio analysis task, like SincNet. It is known that a perfect band-pass filter corresponds to a rectangular window in the frequency domain, which is a not differentiable function; therefore, it cannot be optimized through gradient-based approaches. To overcome this problem, I approximated the rectangular window with the Butterworth window [168].

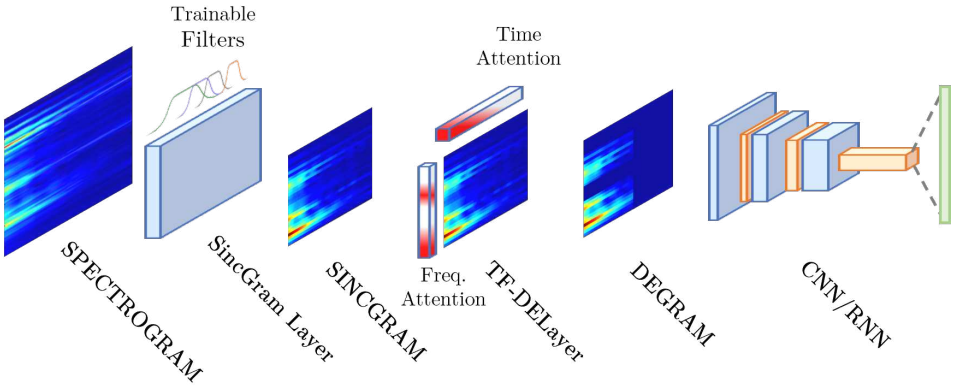


Figure 3.6: DEGram-based CNN architecture, audio representations are labeled in uppercase and layers in lowercase. Computed the spectrogram of the audio signal, the SincGram representation is then obtained by filtering the spectrogram through trainable band-pass filters (SincGram Layer). The DEGram representation is finally computed by applying a temporal and frequency attention layer able to focus on the relevant features of the input audio signal. Upon the learned audio representation, all the state-of-the-art CNNs and RNNs can be used in order to deal with different audio analysis tasks.

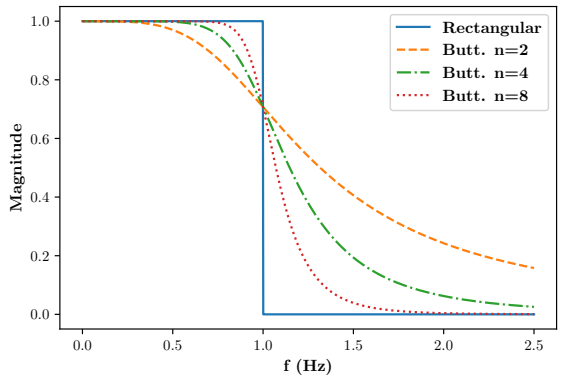


Figure 3.7: Magnitude spectrum of the rectangular and Butterworth windows. The Butterworth window has been computed for three different orders, *i.e.* 2, 4, and 8. As the order increases, the Butterworth window better approximates the rectangular window.

The resulting equation of our filters is thus:

$$g(f, \alpha, \beta, n) = \frac{1}{\sqrt{1 + (\frac{f-\alpha}{\beta})^{2n}}} \tag{3.6}$$

where g is the impulsive response of the filter centered at frequency α, and having scaling factor β and order n; f represents the frequency domain independent variable. Figure 3.7 shows how an increasing window order results in a better approximation of the rectangular window. On the other hand, a more shaped window results in more flat gradients and then it suffers of the vanishing gradient problem.

Considering that the filtering operation can be computed as a multiplication in the frequency domain, the SincGram layer can be formulated as follow:

$$SincGram(X, \bar{\alpha}, \bar{\beta}) = X \cdot W(\bar{\alpha}, \bar{\beta}) \tag{3.7}$$

where $X \in \mathbb{R}^{T \times F}$ is the input spectrogram composed of T time steps and F frequency bins; $\bar{\alpha}, \bar{\beta} \in \mathbb{R}^M$ are the weights of the layer, whose size is equal to the number of filters to train; $W(\bar{\alpha}, \bar{\beta}) \in \mathbb{R}^{F \times M}$ is the resulting filtering matrix, in which each column represents a band-pass filter computed over the frequency bins of the spectrogram. For the sake of readability, the dependency between the SincGram layer and the order n of its filters is not reported because the order is a hyper-parameter and, thus, it is not learned during the training procedure.

The proposed SincGram only needs to learn the center frequencies and the scaling factors of the filters and, therefore, considerably reduces the number of parameters to optimize. Being a linear transformation of the spectrogram (as shown in Equation 3.7), SincGram inherits a receptive field invariant to the sampling rate (unlike the raw-waveform models). When the sampling rate increases, it is common to increase the number of frequency bins F to balance the frequency resolution of the spectrogram itself; this operation results in a heavier network when only the spectrogram is used, since the feature maps are bigger. On the other hand, the resolution of the energies computed by SincGram can be extended without changing the input shape, which is only function of the time steps T and the number of filters M.

Finally, SincGram explicitly represents the time and frequency components of the input signal. In this way, the output representation is well suited for those tasks in which there is the need to temporally align the inputs and the outputs of the neural network architecture (*e.g.* audio event detection and speech recognition).

TF-DELAYER

As previously discussed for SincNet, also SincGram is not able to deal with additive noise in those components, which are common in real noisy environments. In fact, the weight associated to each frequency band is fixed by the following layers within the neural network ar-

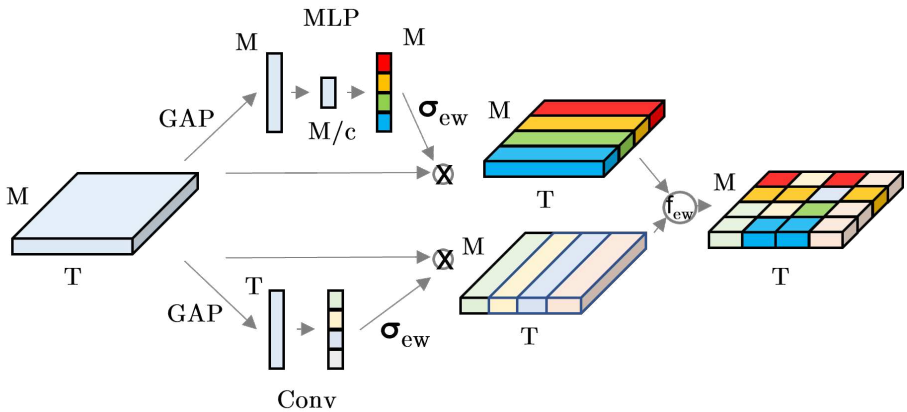


Figure 3.8: TF-DELayer computations on the input representation composed by T time steps and M filters. The temporal and frequency attention are independently computed. The frequency attention scores are computed by passing the squeezed representation along the time dimension to a bottleneck MLP. The temporal attention is computed in the same way but replacing the bottleneck MLP with a Convolutional layer to manage the variable length of the input signal and squeezing the frequency dimension. The sigmoid function is applied to normalize the obtained scores in the range [0,1]. Finally, the scaled temporal and frequency features are combined in the DEGram representation by adopting an element-wise aggregation function (*e.g.* max and average).

chitecture. Even if the previously proposed DELayer allows to reach good performance, the need for an auxiliary neural network to compute the band weights entangles the optimization and slows down the model inference. Furthermore, the attention mechanism dynamically identifies only the frequency bands with higher importance without any consideration of the temporal information.

For these reasons, I propose the Time-Frequency DELayer (TF-DELayer, shown in Figure 3.8) which combines time and frequency attention maps to enhance those useful frequency bands only in the time slots in which the sound of interest is located. Moreover, the improved layer computes the attention weights taking into account all the frequency bands together, in order to better identify the most relevant ones.

The TF-DELayer adapts the spatial and channel two-dimensional attention [169] to a time-frequency mono-dimensional representation; therefore, it consists of two parallel branches for frequency and time attention, respectively.

The frequency attention branch can be formalized as follows:

$$Freq_{att}(s) = \sigma_{ew}(MLP(GAP(s))) \times s \tag{3.8}$$

It squeezes the input SincGram s along the time component by aggregating the information of each frequency band through a Global Average Pooling (GAP) layer, to reduce the complexity of the representation. Then, a bottleneck multilayer perceptron (MLP) is used to compute the attention coefficients. It is characterized by a reduction coefficient c, which allows to further reduce the number of parameters. The attention coefficients are then projected in the range $[0, 1]$ through the element-wise sigmoid function σ_{ew}. Finally, the filtered frequency bands are computed by multiplying the resulting attention coefficient with the original input. It is important to point out that the MLP can be used thanks to the fact that the number of filters

in the SincGram representation is fixed and does not depend on the input length and the sampling rate.

With respect to the previous DELayer, the TF-DELayer considerably reduces the computational complexity of the frequency attention branch thanks to the squeezing operation introduced by the GAP layer. This operation not only speeds up the training and the inference time but also reduces the number of parameters to train, making the optimization easier and allowing the model to better deal with small datasets. Moreover, the way in which the frequency bands to enhance are identified is *orthogonal* with respect to the previous layer in which each component is analyzed independently; in the new layer, the GAP layer exploits the inter-correlations between the frequency bands to better identify the most relevant ones.

The time attention mechanism can be formalized as follows:

$$Time_{att}(s) = \sigma_{ew}(Conv1D(GAP(s))) \times s \tag{3.9}$$

Also, this module squeezes the GAP of the input SincGram s to reduce the complexity of the layer itself but, in this case, the aggregation is computed at the frequency level. Since different audio samples can have variable lengths, the attention maps are computed through a mono-dimensional convolutional layer (Conv1D) followed by a sigmoid activation function. A padding is applied to the input in order to obtain a number of attention weights equivalent to the time steps of the input. The time-filtered representation is computed as a multiplication between the attention weights and the input SincGram. Adding this attention branch, the TF-Layer not only enhances relevant frequency bands while attenuating background noise but also focuses the attention only on the time slots in which the sound of interest is located.

The final DEGram representation is computed starting from the two filtered representations through an element-wise aggregation func-

tion f_ew:

$$DEGram(s) = f_{ew}(Freq_{att}(s), Time_{att}(s)) \tag{3.10}$$

This function aims to combine the two attention contributions to obtain the final representation.

3.3.1.2 DEGramNet

Having available the low-level representation of the audio signal, one needs to compute high-level features to address a particular audio analysis task. As shown in Figure 3.6 and discussed in Section 3.3.1.1, the proposed DEGram is a 2-dimensional time-frequency representation and, thus, it can be combined with all the well-known Convolutional and Recurrent Neural Networks. This is a crucial feature of the method, that allows researchers to use the proposed learnable representation with any neural network architecture.

In the proposed DEGramNet, I adopt the Efficient Channel Attention Network (ECA-Net) as a reference CNN architecture to prove the effectiveness of the representation. ECA-Net [167] has been chosen for two main reasons. First, the network is composed of *Residual Blocks* [170][171] which learn a residual function of the input in contrast to VGG-like architectures [172], as shown in Figure 3.9. Residual blocks proved to be easier to optimize and, moreover, they allow to alleviate the problem of the vanishing gradient in very deep neural networks. This effect is achieved thanks to the direct connection with the input signal which allows a direct back-propagation of the gradients themselves.

The second feature of ECA-Net is the *Efficient Channel Attention module*. Differently from the TF-DELayer, the ECA module is applied on the feature maps (*i.e.* channels). In particular, it aims to improve the hidden representations produced by the neural network by explicitly modeling the inter-correlations between the feature maps

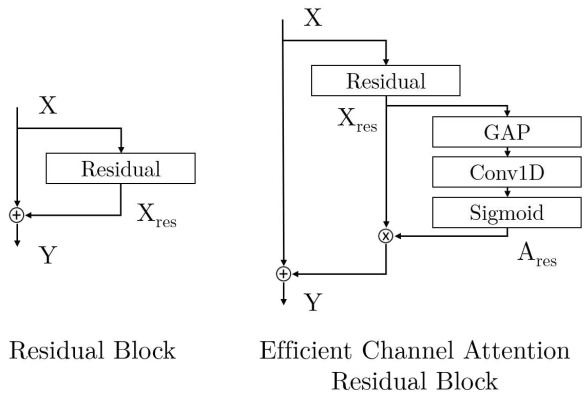


Figure 3.9: Comparison between Residual Block and Efficient Channel Attention Residual Block.

in each convolutional layer. It has been proved that this channel-wise feature re-calibration allows to increase the generalization capabilities of the standard residual networks which are commonly adopted for audio analysis tasks [169] [167] [173].

From the computational point of view, the ECA module works similarly to the frequency attention mechanism reported in Equation 3.8, but the inter-correlations between the different feature maps are computed locally through the use of a 1-dimensional convolutional layer instead of the bottleneck MLP. Even if the choice of replacing the MLP with a convolutional layer implies a sort of locality of the inter-correlations between the feature maps, it allows to efficiently compute the attention weights and to better scale with the network depth, avoiding the explosion of the model complexity.

The kernel size k of the convolutional filter is computed as follows:

$$k = \left| \frac{log_2(C) + 1}{2} \right|_{odd} \tag{3.11}$$

where C is the number of feature maps. This choice helps to reduce the number of hyperparameters to tune.

The final architecture of DEGramNet is summarized in Table 3.10.

3.3.2 Experimental framework

In order to prove the generalization capability of DEGramNet and to validate the assumptions done for the proposed DEGram representation, I performed experiments on two audio classification tasks characterized by different sound of interest: Sound Event Classification and Speaker Identification. In particular, in order to benchmark the robustness of the proposed DEGramNet to environmental noise, I adopted two large-scale dataset acquired in unconstrained conditions: VGGSound [174], for the task of Sound Event Classification, and VoxCeleb1 [175], for the task of Speaker Identification. This section describes the datasets (Section 3.3.2.1), the metrics adopted to evaluate the performance of DEGramNet (Section 3.3.2.2), and the implementation details necessary to reproduce our experimental framework (Section 3.3.2.3).

3.3.2.1 Datasets

VGGSOUND

VGGSound [174] is a large-scale audio-visual dataset recently proposed for the task of sound event classification in the wild, *i.e.* in uncontrolled conditions. For each sample, the object emitting the sound is visible in the corresponding video. VGGSound contains 309 classes of sound events ranging from people sounds to nature and animals. The dataset is composed of about 200k video clips downloaded from Youtube and acquired in the wild, allowing to achieve a two-fold result: a substantial increase of the task difficulty due to different challenging

Table 3.10: DEGramNet convolutional architecture. Each convolutional block corresponds to an ECA residual block. The global average polling layer upon the convolutional architecture allows the network to deal with variable length audio samples.

Layer Name	**Residual Block Convolutions**
SincGram	-
TF-DELayer	-
conv1	7×7, 64, stride 2
maxpool1	3×3, stride 2
conv2	$\begin{bmatrix} 1\times1 \text{ conv, } 64 \\ 3\times3 \text{ conv, } 64 \\ 1\times1 \text{ conv, } 256 \end{bmatrix} \times3$
conv3	$\begin{bmatrix} 1\times1 \text{ conv, } 128 \\ 3\times3 \text{ conv, } 128 \\ 1\times1 \text{ conv, } 512 \end{bmatrix} \times8$
conv4	$\begin{bmatrix} 1\times1 \text{ conv, } 256 \\ 3\times3 \text{ conv, } 256 \\ 1\times1 \text{ conv, } 1024 \end{bmatrix} \times36$
conv5	$\begin{bmatrix} 1\times1 \text{ conv, } 256 \\ 3\times3 \text{ conv, } 256 \\ 1\times1 \text{ conv, } 1024 \end{bmatrix} \times3$
	average pool, n_classes-d fc, softmax

Table 3.11: Statistics of the VoxCeleb1 dataset for Speaker Identification. Fields characterized by three value represents max/avg/min statistics respectively.

# of POIs	1,215
# of male POIs	690
# of videos per POI	36 / 18 / 8
# of utterances per POI	250 / 123 / 45
Length of utterances (s)	145.0 / 8.2 / 4.0

acoustic environment and noisy acquisitions and the removal of the bias due to the acquisition conditions (*e.g.* the recording device).

VGGSound has a pre-defined training-test set division and contains audio samples that are 10-seconds long. The audio samples in the test set are uniformly distributed with respect to the sound classes, while the training set is highly unbalanced.

VOXCELEB1

VoxCeleb1 [175] is a large-scale audio-visual speech dataset for the tasks of Speaker Identification and Speaker Verification. The dataset consists of more than 100k utterances acquired from 1,215 celebrities in the wild. Even in this case, VoxCeleb1 has been acquired starting from YouTube video clips and, thus, contains a wide range of recording conditions, ranging from outdoor stadiums to quiet studio interviews. The speakers within VoxCeleb1 are characterized by a wide range of accents, ages, and ethnicity; moreover, the dataset is gender-balanced, with 55% of male speakers. The summary of the dataset is reported in Table 3.11.

VoxCeleb1 has a pre-defined training-validation-test set division; all the sets are unbalanced. The utterances within the dataset have a variable duration, ranging from less than 1 to 20 seconds.

3.3.2.2 Evaluation metrics

In order to compute the performance of DEGramNet, I adopted the same evaluation protocol proposed in the papers of the considered datasets [174] [175]. In particular, the metrics are: mean Average Precision (mAP), macro-averaged Area Under the receiver operating characteristic (ROC) Curve (AUC) and the equivalent d-prime class separation index, Top-1, and Top-5 Accuracy. For VoxCeleb1, only the Top-1 and Top-5 Accuracy scores have been computed.

The mAP score (Equation 3.12) is the mean of the Average Precision (AP) scores computed for each class.

$$mAP = \frac{1}{N}\sum_{i}^{N} AP_i \tag{3.12}$$

The AP [176] is, in turn, the average percentage of correctly detected samples within the ranked list of the detections by varying the size of the list (from 1 to the number of detections).

The AUC [177], as suggested by the name, is the area under the ROC curve, that draws the True Positive Rate and False Positive Rate as functions of the detection threshold. The AUC score is computed for each class and then averaged in order to obtain a priors-invariant metric score. The relative d-prime index represents the separability between classes and is computed as:

$$d' = \sqrt{2}F^{-1}(AUC) \tag{3.13}$$

where F^{-1} is the inverse cumulative distribution function for a Gaussian unit.

Finally, the Top-1 and Top-5 Accuracy scores are computed as the percentage of correctly classified samples considering the top one and the top five classes with higher predicted probability, respectively.

3.3.2.3 Implementation details

The DEGramNet architecture has been trained through the Adam optimization algorithm. The learning rate used to train the neural network is equal to 3e-4, while the batches consist of 64 samples. These hyperparameters have been selected with a Bayesian Optimization approach. During the training procedure, the learning rate has been reduced by a factor of 10 if the accuracy on the validation set did not decrease for 7 consecutive epochs, in order to avoid the presence of a plateau in the error function. Moreover, an Early Stopping strategy has been adopted to not overfit the training data; the patience has been set to 10. To avoid the specialization of the network over the most represented classes, I balanced the dataset at the beginning of each epoch by randomly under-sampling the majority of classes without replacement.

I set the order of the Butterworth window within the SincGram representation to 4 while the compression coefficient is set to 0.0625. Finally, I fixed the number of trainable filters to 64. These hyperparameters have been optimized through a grid search, while the number of spectrogram bins has been set to 257 as done in [174].

I adopted different regularization strategies within the training of DEGramNet. In order to obtain a neural network able to deal with audio samples of different lengths, I created at each training iteration a batch with a time length randomly chosen in the range $[4, 6]$ seconds. To this aim, I replicated the samples shorter than the selected window and cut the longer ones. Finally, I augmented the training set using the SpecAugment procedure [178]. It is worth mentioning that the augmentation is applied within the network itself after the computation of the DEGram representation to better regularize the remaining part of the CNN.

3.3.3 Experimental results

In this section, I report the experimental results obtained on VGGSound and VoxCeleb1 by the proposed DEGramNet, compared with the ones achieved by other state-of-the-art algorithms optimized for the same tasks. Furthermore, I carried out an ablation study on the contribution of each additional step of the proposed representation (*i.e.* SincGram and DEGram) considering, for the sake of comparability with other methods in the literature, the well-known ResNet architecture [170] [171] (Section 3.3.3.1). Moreover, an in-depth analysis of the learned representation has been conducted by visually comparing the input obtained with Spectrogram, SincGram and DEGram. Finally, the processing time has been evaluated on a server with the NVIDIA Titan XP GPU and the NVIDIA Jetson Xavier NX embedded system (Section 3.3.3.2) to investigate the usability of the proposed representation in real applications.

I report in Table 3.12 and Table 3.13 the results obtained by DEGramNet on VGGSound and VoxCeleb1. I compared the proposed model with other methods evaluated on the same datasets. These results have been reported from [174, 179, 180] and [175] in order to make a fair comparison.

The proposed DEGramNet achieved state-of-the-art performance on the VGGSound dataset with a mAP of 0.547 and a Top-1 accuracy equal to 0.526. The proposed method outperformed all the standard CNN architectures for audio analysis based on Residual Networks. DEGramNet gained 0.4% in terms of AUC and 0.083 in terms of d-prime compared to the deeper ResNet (*i.e.* ResNet50, with AUC and d-prime equal 0.973 and 2.735, respectively), demonstrating a strong capability to distinguish positive and negative samples despite the high variability of the dataset. The gap increases up to 0.9% and 0.191 for smaller architectures as ResNet18. The proposed method also showed better ranking performance, gaining from 1.7% (w.r.t.

Table 3.12: Comparison of the results achieved by DEGramNet and other state-of-the-art methods on the VGGSound test set (Sound Event Classification) in terms of mAP, AuC, d-prime, Top-1 and Top-5 Accuracy (Acc.). The best results for each performance metric are highlighted in bold.

Method	**Ref.**	**mAP**	**AUC**	**d-prime**	**Top-1 Acc.**	**Top-5 Acc.**
ResNet-18	[174]	0.516	0.968	2.627	0.488	0.746
ResNet-34	[174]	0.529	0.972	2.703	0.505	0.758
ResNet-50	[174]	0.532	0.973	2.735	0.510	0.764
SlowFast	[179]	0.544	0.974	2.761	0.525	**0.781**
DEGramNet	-	**0.547**	**0.977**	**2.818**	**0.526**	0.774

Table 3.13: Comparison of the results achieved by DEGramNet and other state-of-the-art methods on the VoxCeleb1 test set (Speaker Identification) in terms of Top-1 and Top-5 Accuracy. The best results are highlighted in bold.

Method	**Ref.**	**Top-1 Acc.**	**Top-5 Acc.**
ResNet-18	[180]	0.795	0.910
VGG-M	[175]	0.805	0.921
ResNet-34	[180]	0.813	0.945
DEGramNet	-	0.866	0.950
AutoSpeech (NAS)	[180]	**0.876**	**0.960**

ResNet50) to 3.5% (w.r.t. ResNet18) in terms of Top-5 accuracy. The result is further confirmed by the mAP metric, that considers the overall ranking performance and not only the Top-5 predictions. In this case, DEGramNet gained from 1.5% if compared to ResNet50 (0.532), up to 3.1% with respect to ResNet18 (0.516).

DEGramNet outperformed the SlowFast architecture, the method achieving so far the best performance on the VGGSound test set, on four metrics over five (except for Top-5 Accuracy). Although SlowFast performs a multi-stream analysis over the Mel-Spectrogram representation and requires a complex architectural design, it was outperformed by DEGramNet on the Top-1 accuracy (gaining 0.1%), AUC and d-prime scores (gaining 0.3% and 0.057, respectively). Therefore, these results prove that the proposed CNN is slightly more accurate on that task by only learning an effective audio representation and without any additional design effort. Moreover, even if DEGramNet obtained a lower Top-5 accuracy score (-1%), it proved to better rank classes in general by obtaining a higher mAP score (0.3%).

Looking at the results obtained on the VoxCeleb1 dataset, the outcomes are quite similar. Even in this case, DEGramNet outperformed all the standard CNN architectures in speaker identification, obtaining a Top-1 Accuracy of 0.866. In particular, the proposed network gained 5.5% with respect to ResNet34 (0.813) and 6.1% if compared with VGG-like neural networks (0.805). Moreover, I compared the results with AutoSpeech, which is a CNN architecture specifically optimized with a Network Architecture Search (NAS) approach on the VoxCeleb1 dataset for the task of Speaker Identification and Verification. We can observe that DEGramNet obtained comparable results to this special-purpose method without any effort, either hand-crafted or computational, in the design step; therefore, our method demonstrated to be a general-purpose architecture for audio analysis tasks. Indeed, this result proves the adaptability of DEGram to different tasks, being the representation able to learn in a single training step

Table 3.14: Comparison of the results obtained on the VGGSound test set by ResNet18trained with Spectrogram, SincGram, DESpectrogram (DESpect.) and DEGram. The best results for each performance metric and for each CNN are highlighted in bold. The CNNs trained with the proposed representations always outperform the corresponding architecture trained with Spectrogram.

Method	**Ref.**	**mAP**	**AUC**	**d-prime**	**Top-1 Acc.**	**Top-5 Acc.**
ResNet-18	[174]	0.516	0.968	2.627	0.488	0.746
Sincgram-ResNet18	-	0.519	0.974	2.747	0.496	0.753
DESpect.-ResNet18	-	0.507	0.972	2.706	0.470	0.740
DEGram-ResNet18	-	**0.526**	**0.975**	**2.765**	**0.507**	**0.763**

the frequency bands of interest for the specific problem at hand.

3.3.3.1 Ablation study

In order to further analyse the positive contribution of the proposed representation with respect to the standard Spectrogram, I conducted an ablation study to evaluate the impact of the single design choices. In particular, I performed a quantitative and qualitative analysis by firstly removing the attention mechanism and then the SincGram filtering layer. I also performed experiments with a representation combining Spectrogram and DELayer, hereinafter called DESpectrogram; this variant allows to measure the positive contribution of Sincgram to the proposed representation. In order to make a comparison independent from the specific dataset chosen for training the methods, I compared the results obtained by the four representations given in input to the ResNet-18 model on the VGGSound and VoxCeleb1 test sets. We reported the results of existing spectrogram-based CNNs from the literature for a fair comparison [174] [180]. The obtained results are shown in Table 3.14 and Table 3.15.

Table 3.15: Comparison of the results obtained on the VoxCeleb1 test set by ResNet18 trained with Spectrogram, SincGram, DESpectrogram and DEGram. The best results for each performance metric and for each CNN are highlighted in bold. The CNNs trained with the proposed representations always outperform the corresponding architecture trained with Spectrogram.

Method	**Ref.**	**Top-1 Acc.**	**Top-5 Acc.**
ResNet-18	[180]	0.795	0.910
Sincgram-ResNet18	-	0.826	0.934
DESpect.-ResNet18	-	0.827	0.932
DEGram-ResNet18	-	**0.831**	**0.939**

I can immediately note that regardless of the task we are dealing with, the proposed representations allow to substantially improve the performance. The spectrogram-based ResNet18 achieved a Top-1 Accuracy of 0.488 and 0.795 on the VGGSound and VoxCeleb1 datasets. Adding to the network the capability to learn the frequencies of interest (SincGram), the CNN was able to improve the performance by around 1% and 3%. A similar improvement can be seen on the VoxCeleb dataset with the DESpectrogram representation, able to better deal with the noise characterizing the dataset. On the other hand, the DESpectrogram representation obtains the worst result on VGGSound. A possible interpretation of this result is that the TF-DELayer is not able alone to distinguish the frequencies with overlapping noise from those of interest for all the considered sounds (*i.e.* 309 classes). Interestingly, the accuracy increases by a further 1% for both tasks when we use DEGram, combining the SincGram representation (learnable frequencies of interest) and the TF-DELayer (noise robustness). This result confirms the effectiveness of the proposed architecture.

To deduce further insights, I performed a qualitative analysis on the various representations, reported in Figure 3.10. In particular, I visually compared the Spectrogram, SincGram and DEGram representations of a noisy speech audio sample, in order to identify the key elements that allow the proposed solutions to be more effective.

We can note that SincGram stretches the frequencies in the lower part of the spectrum, in which most of the relevant speech information is located. Nevertheless, as expected, also overlapping noise is amplified when the spectrum is processed by the learned filters. The DEGram representation mitigates this issue since it is able to perform denoising through the temporal attention module. This result is evident by observing that all the components not overlapping with the speech signal (temporally centered) are close to 0 (teal colored). Moreover, looking at the upper part of the DEGram representation, one can note smoothed energies with respect to SincGram, proving that the frequency attention mechanism attempted to reduce the noise in that part of the spectrum.

The qualitative analysis confirms the assumptions made in the design of DEGram. The proposed audio representation is actually able to amplify the relevant components of the spectrum and to attenuate the ones affected by noise, focusing only on the temporal steps in which the sound is located.

3.3.3.2 Processing time evaluation

To evaluate the computational impact of the proposed representation on the processing time of an audio analysis system, I compared the latency of the ResNet18 architecture with and without the proposed DEGram representation and the latency of a Spectrogram-based ResNet34. The evaluation has been carried out on two different GPU-enabled devices, namely a server equipped with the NVIDIA Titan XP GPU and a NVIDIA Jetson Xavier NX, an embedded system for

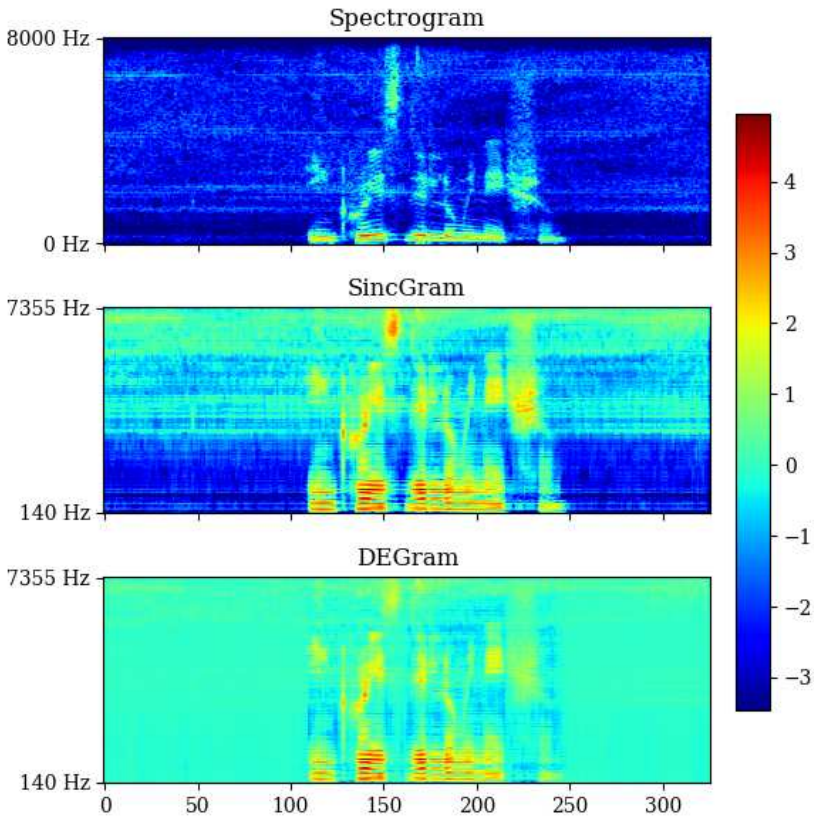


Figure 3.10: Comparison between Spectrogram, SincGram and DEGram applied to a noisy speech sample. SincGram focuses on the most important frequency components (the lower part of the spectrum) and the effect is evident wih respect to Spectrogram. Then, DEGram performs denoising on the frequencies affected by overlapping noise (higher part of the learned spectrum) and focuses on the time slots in which the sound of interest is located.

Table 3.16: Processing time in milliseconds required by ResNet18, DEGram-ResNet18 (DEGram-RN18) and ResNet34 to process audio samples with variable input length (1, 3, 5, 10 seconds) over the NVIDIA Titan XP GPU and the NVIDIA Jetson Xavier NX embedded system. The ResNet18 trained with the proposed DEGram is able to reduce the processing time with respect to the spectrogram-based ResNet34 (achieving lower performance). The proposed architecture reduces the latency of the same CNN architecture on CPU-based architectures.

Device	**Input len.**	**ResNet18**	**DEGram-RN18**	**ResNet34**
Titan XP	*1 sec*	6	7	10
	3 secs	7	7	10
	5 secs	7	8	10
	10 secs	10	9	15
Jetson Xavier NX (GPU)	*1 sec*	165	174	180
	3 secs	172	174	182
	5 secs	177	177	199
	10 secs	201	207	244
Jetson Xavier NX (CPU)	*1 sec*	220	212	284
	3 secs	298	244	408
	5 secs	354	278	499
	10 secs	479	362	745

IoT and Robotics applications. In addition, I evaluated the processing time on the latter device with and without the use of the GPU. The inference time has been computed by averaging the processing time measured over 100 network predictions. The obtained results are reported in Table 3.16.

One can note that the proposed DEGram representation has a negligible overhead on GPU architectures (1 ms on the Titan XP and less than 10 ms on the Jetson Xavier NX) when using the same CNN architecture for inference. On the same devices, DEGram is able to

save up to 37 ms with respect to the ResNet34 architecture; this is not negligible, especially if we consider, as shown in the previous section, that ResNet18 trained with DEGram (Table 3.14 and 3.15) achieves better accuracy than ResNet34 trained with Spectrogram (Table 3.12 and 3.13). The processing speed-up is even higher when the computation is done on a CPU. In this case, the ResNet architecture based on DEGram reduces the processing time up to 110 ms on 10 seconds inputs, thanks to the reduced number of frequency features (from 257 frequency bins to 64 trainable filters). The gap drastically increases comparing the proposed architecture with ResNet34, which requires around twice for computing sequences longer than 3 seconds.

Chapter 4

Multitask audio neural networks for social robots

4.1 Background

In this chapter, I aim to further optimize the audio perception pipeline of a Social Robot architecture. As discussed in Chapter 1.3.2.2, using one deep neural network for each of the tasks that a Social Robot has to perform is not a feasible solution. For this reason, multitask neural networks have been identified as a possible alternative to allow real-time computation on board of the social robots embedded systems. However, MTNs need to be properly designed and trained to achieve satisfactory performance.

In light of the state-of-the-art discussion (Section 1.3.2.2), in this chapter I propose three multi-task neural networks for simultaneously dealing with the following tasks: Speaker Re-Identification, Gender Recognition, Age Estimation, and Emotion Recognition from speech signals. The three different MTN architectures are useful to identify the good trade-off between the computation complexity (*i.e.* the layer sharing level) and the performance obtained on the analysed tasks. The ResNet18 convolutional neural network has been selected as feature extraction backbone due to the strict computational requirements. The considered tasks have never been tackled together due to the absence of a single dataset containing all the labels. To this aim, a training procedure that takes into account the multiple datasets and missing labels through label masking is proposed in the chapter. Moreover, I overcome the problem of loss balancing through the adoption of the GradNorm balancing strategy, a novel approach able to automatically and dynamically tune the weights of the loss functions directly during the training, avoiding huge tuning time. Finally, all the proposed multitask models rely on the robust and efficient DEGram audio representation framework discussed in Chapter 3.

The proposed multitask models have been validated on standard benchmark datasets for the considered tasks: VoxCeleb 1 and 2, IEMOCAP, Mozilla Common Voice and CMU-MOSEI. The multitask mod-

els not only proved to overperform single-task baselines on three tasks over four but also outperformed more complex state-of-the-art models on Gender Recognition, Emotion Recognition, and Age Estimation tasks. This result proves the effectiveness of the proposed training procedure considering the very simple CNN backbone architecture used as a features extractor.

4.2 Methodology

This section describes the architecture of the proposed multitask neural networks, from now on called MTN vA, vB, and vC. Single and multitask models rely on the same convolutional architecture which, thanks to DEGram, is able to learn a time-frequency audio representation directly from data. In Section 4.2.3 I reported the details of the adaptation of the convolutional backbone to the MTN vA, vB, and vC multitask architectures.

4.2.1 Multitask network architectures

The proposed multitask network architectures are reported in Figure 4.1 and detailed in the following subsections.

4.2.1.1 MTN vA

MTN vA is a multitask architecture specifically designed to exploit the inter-dependencies between the involved tasks in a very efficient and simple way. It is composed of a single CNN backbone network which is adopted to extract a shared representation between the tasks. Then, the obtained feature tensor is passed to a Global Average Pooling (GAP) layer to obtain a length-invariant audio representation. Finally, a task-specific Fully Connected (FC) layer is used to infer the label for each of the tasks.

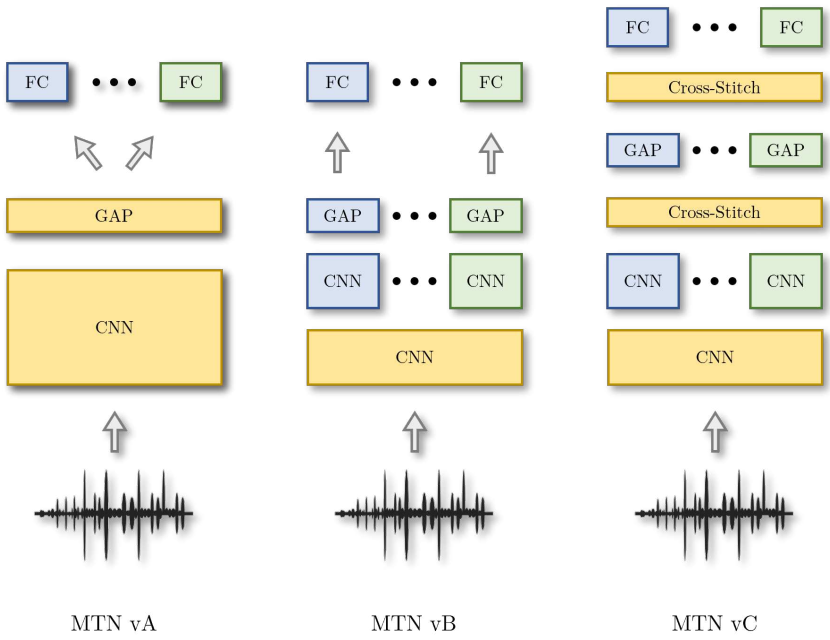


Figure 4.1: Trained multitask networks. Shared layers are colored in yellow while a different color is used to represent each task-related layer. MTN vA relies on a Convolutional Neural Network (CNN) to extract a single representation which is averaged along time using a Global Average Pooling (GAP) layer; the obtained representation is projected in a task-specific subspace to make the related prediction through Fully Connected (FC) layers. Differently from vA, MTN vB is characterized by multiple task-specific layers (also convolutional). Finally, MTN vC linearly combines task-specific features through a Cross-Stitch layer.

The main advantage of MTN vA consists of the need for just a single processing step before the prediction one, obtaining a speedup proportional to the number of tasks. Moreover, also the number of parameters is significantly reduced leading two a twofold effect. Firstly, a simpler model better deals with small-sized training datasets, reducing the risk of overfitting. Second, the reduced number of parameters reduces the size of the model in memory proportional, again, to the number of tasks. The MTN vA generally performs well when a single set of features can be exploited by different tasks, each one performing a regularization action on the others. As a drawback, when each task needs for its specific features, a fixed-sized representation in output from a single backbone may be not sufficient and may cause a predominance of a task with respect to the others.

4.2.1.2 MTN vB

The MTN vB is a multitask network architecture designed to overcome to the limitations of the MTN vA one. Differently from it, the MTN vB is characterized by a smaller shared CNN backbone and deeper task-specific branches. The backbone is in charge to compute low-level audio features (supporting the choice of a smaller model), which are common to the involved tasks. After, each branch creates a disentangled representation at a higher abstraction level starting from the low-level features; the resulting level of abstraction depends on the number of layers in the task-specific branch. For sake of simplicity, the number of layers in each branch is the same for all the tasks and structured as the base architecture of the backbone (more details in Section 4.2.2). Of course, a different choice may result in a better result but relaxing this constraint exponentially enlarges the hyper-parameters search space due to the infinite possible branching combination options.

The MTN vB model inherits from MTN vA the reduced, even if

not drastically, number of parameters and the exploitation of low-level inter-dependencies between tasks through a shared CNN backbone. The increased number of parameters is offset by the possibility for each task to create its own high-level representation without being constrained by the other ones. The main challenge in designing this architecture is to identify the right point in which to branch the task-specific sub-networks. In particular, there is a trade-off between the need for more degree of freedom of some tasks and the regularization effect between the remaining ones.

4.2.1.3 MTN vC

The MTN vC is finally proposed to overcome to issues identified in the design model MTN vB. MTN vC has the same number of task-specific layers as MTN vB. Differently from it, MTN vC inserts a shared Cross-Stitch [181] layer between each couple of task-specific layers. This layer linearly aggregates the feature vectors in output from the task-specific layers at the same abstraction level. The aggregation weights are learned by the layer itself and are independent for each task. In this way, each task-specific branch can get, if needed, features from the others at a higher level and, therefore, apply a regularization action to it. The main difference between MTN vA and MTN vC consist of the fostering of task-specific branch to affect other branches only if needed.

The MTN vC architecture has the same advantages of MTN vB while, at a negligible computational cost, overcoming its disadvantages. The Cross-Stitch layer only models linear inter-dependencies between task-specific features. As a drawback, MTN vC is still sensible to the point in which the branching is performed, which regulates the trade-off between efficiency and obtained performance.

Table 4.1: ResNet18 convolutional architecture. Each convolutional (conv) block corresponds to a residual block. The global average polling layer upon the convolutional architecture allows the network to deal with variable length audio samples.

Layer Name	**Residual Block Convolutions**
conv1	7×7, 64, stride 2
maxpool1	3×3, stride 2
conv2	[3×3 conv, 64; 3×3 conv, 64] ×2
conv3	[3×3 conv, 128; 3×3 conv, 128] ×2
conv4	[3×3 conv, 256; 3×3 conv, 256] ×2
conv5	[3×3 conv, 512; 3×3 conv, 512] ×2
	average pool, n_classes-d fc, softmax

4.2.2 Convolutional neural network backbone

I choose ResNet18 [170] as convolutional neural network to extract audio features from time-frequency representations. The standard ResNet18 architecture is reported in Table 4.1. It relies on a Global Average Pooling (GAP) layer to deal with input utterances of different lengths. It collapses the input features tensor into a fixed-length vector with a size equal to the number of features maps (*i.e.* 512). The main block ResNet18 and residual networks in general is the *residual block*. The residual block learns a residual function of the input in contrast to VGG-like architectures [172]. Thanks to the direct (residual) connections between the input and the output, this kind of model

alleviate the vanishing gradient effect and, therefore, is easier to optimize. Thanks to these properties, residual learning layers are standard in deep learning models to date.

4.2.3 Backbone adaptation to multitask learning

I considered the feature vector in output from the GAP layer as the shared representation of MTN vA. At this point of the model, a different fully connected layer is defined to make a prediction for each of the tasks.

Regarding models vB and vC, I shared the CNN backbone architecture up to the *conv4* residual block (Table 4.1) to obtain a good trade-off between efficiency and the needed abstraction of shared features. This choice is the result of a grid search analysis.

In the MTN vC architecture, a Cross-Stitch layer is inserted before and after the GAP layer. I avoided to modify the residual block because changing the block structure may drastically affect the results[171].

4.2.4 Multitask network loss

One of the main difficulties in training a multitask neural network consists of data acquisition. It is hard to find a training dataset where each sample is labeled with all the tasks one needs to deal with. The training of just the decision layers of the network while freezing feature extraction ones may result in a sub-optimal solution.

To overcome this issue, I trained the multitask models subject of this chapter propagating the loss gradients only for labels available on the current sample. The resulting loss function is reported in Equation 4.1.

$$L_k = \sum_{i=1}^{M} \sum_{t=1}^{N} \alpha_{i,t} L_t(\hat{y}_{i,t}, y_{i,t})$$
$$\alpha_{i,t} = 1 \; if \; y_{i,t} \in Y, 0 \; otherwise \tag{4.1}$$

where L_k is the loss computed at the training step k, M is the number of samples, and N is the number of tasks. $\hat{y}_{i,t}$ and $y_{i,t}$ are respectively the model prediction and the ground truth for the sample i on the task t. L_t is the loss function used for the task t. $\alpha_{i,t}$ is a weight equal to 1 if the label $y_{i,t}$ is contained in the available label set Y, 0 otherwise. This approach allows to deal with multiple datasets even if they are labeled for completely different tasks. It is important, in order to avoid unbalancing, to construct batches representing all the tasks.

Being the value of α just one within the set $[0, 1]$, the resulting loss L can be mainly influenced by the most represented task. Nevertheless, constructing the batch oversampling samples for the unrepresented tasks may lead to the same result. This behaviour is drastically amplified in the case that the losses' nature is different (*e.g.* Mean Square Error and Cross Entropy).

To overcome this behaviour, the models dynamically adjust the weights of each task during the training through the GradNorm algorithm [141]. The updated loss function is reported in Equation 4.2.

$$L' = \sum_{i=1}^{M} \sum_{t=1}^{N} \alpha_{i,t,k} L_t(\hat{y}_{i,t}, y_{i,t})$$
$$\alpha_{i,t,k} = w_{t,k} \; if \; y_{i,t} \in Y, 0 \; otherwise \tag{4.2}$$

where $w_{t,k}$ represents the weight of the task t at the training step k.

To explain the GradNorm optimization algorithm the following quantities have to be defined:

- $\overline{G}_k$ is the average L^2 norm of the gradient computed for each task ($|G_{t,k}|$) at the training step k;

- $\tilde{L}_{t,k}$ is the inverse training rate and computed as ratio between the loss for the task t at the training step k divided by the initial loss;

- $r_{t,k}$ is the relative inverse training rate, obtained by dividing the inverse training rate by the average value computed over all the tasks.

GradNorm updates each task weight $w_{t,k}$ by minimizing the following loss function:

$$|G_{t,k} - \overline{G}_k \cdot r_{t,k}| \tag{4.3}$$

This loss function has two objectives: first, it adapts the weight of the task t to obtain a L^2 norm of the gradient similar for all the tasks; regarding the second objective, let us consider $\tilde{L}_{t,k}$ as an inverse indicator of "how much a model a particular task w.r.t. the initial loss" and, consequently, $r_{t,k}$ is the same index but projected in the range $[0, 1]$. In this perspective, GradNorm reduces the reference norm of the gradients, and then the priority with which a task should be learned, by multiplying it by $r_{t,k}$ for the tasks that are learning faster w.r.t. the others. In this way, the algorithms allows to keep similar learning pace of the considered tasks.

4.3 Experimental framework

In this section, I reported the main details about the experimental setup to allow the reader to reproduce the results. This chapter deals

with the task that a social robot has to perform in order to obtain personalized and emphatic conversations: Re-Identification, Age estimation, Gender and Emotion recognition. The tasks are all approached relying only on the audio modality.

To the best of my knowledge, models performing re-identification are not compared on a standard benchmark; instead, the common approach is to indirectly approach the problem as a Verification one, *i.e.* deciding if two utterances have been pronounced by the same speaker or not. Trivially, it is possible to re-identify people by storing utterances from people we would like to re-identify and, each time an input utterance is available, compare it with the stored reference set. In this chapter, I approached re-identification in the same way. Consequently, the performance is reported for the task of Speaker Verification being the study of multitask networks design and related training algorithms the focus of the chapter.

4.3.1 Datasets

I reported in Table 4.2 the considered datasets together with their statistics and related tasks.

CMU-MOSEI [182] is one of the largest multimodal datasets for Sentiment and Emotion recognition to date. It is composed of more than 23k utterances obtained from YouTube videos. The videos are related to 1000 YouTube speakers. I included CMU-MOSEI in the training set considering the labels for both the tasks of Emotion and Gender recognition.

IEMOCAP[183] is a multimodal emotion recognition dataset acquired from recited interactions. It is composed of more than 10k utterances obtained in controlled environments. In the literature, IEMOCAP has been used for Emotion recognition considering several sub-set of emotion classes. For the purposes of this thesis and according to [184], I considered for IEMOCAP and CMU-MOSEI only

Table 4.2: Statistics of the datasets adopted to train and validate multitask and single task models. The involved task are Speaker Re-Identification (R), Gender Recognition (G), Age Estimation (A), Emotion Recognition (E).

Dataset	**Task**	**# of samples**	**Hours**	**Set**
CMU-MOSEI	G E	23 k	65	Train
IEMOCAP	E	10 k	11	Train, Test
VoxCeleb 2	R A G	1.13 M	2.4k	Train, Test
Common Voice	G	73 k	83	Test
VoxCeleb 1	R G	153 k	352	Test

the following emotions: Neutral, Angry, Happy, and Sad. I included IEMOCAP in both the training and test sets.

Mozilla Common Voice[185] is a large scale crowd-sourced multilingual speech recognition dataset. It is composed of around 73k utterances obtained in non-controlled recording conditions also labeled for the task of gender recognition [186]. I included Mozilla Common Voice as a test set for the task of Gender Recognition. Being a dataset with a samples distribution different from the training one, it makes the model validation more reliable.

VoxCeleb 1 has been already presented in Section 3.3.2.1. *VoxCeleb 2*[175, 187] is a large-scale Speaker Recognition dataset acquired in uncontrolled environments and state-of-the-art benchmarks for speaker-related tasks. It is composed by 1.13M of utterances. Recently, the dataset has been labeled with age labels in [188]. I adopted VoxCeleb 1 as test set and VoxCeleb 2 as training set as commonly done in Speaker Verification benchmarks. There is no identities overlap between the selected sets. I also used VoxCeleb 2 as test set for the age estimation task. Unfortunately, the test set defined in [188] overlaps with the training set of the speaker verification task. For this reason,

I removed from the training set all the identities contained in the test one for a fair performance evaluation.

4.3.2 Evaluation metrics

I reported in this section the evaluation metrics adopted in this chapter to estimate the performance of the trained models for each of the tasks. I considered the most common performance indices used for each of them.

I evaluated the performance on the Speaker Verification task in terms of *Equal Error Rate* (EER). Considering that we are dealing with a binary classification problem where a true positive is a pair of utterances correctly classified as pronounced from the same speaker. The EER can be computed by identifying the threshold value that allows the model to obtain a True Positive Rate equal to the True Negative Rate, *i.e.* the Type-I and Type-II errors are equal. The resulting error value is the EER index.

For both the tasks of Gender and Emotion Recognition, I evaluated the performance in terms of *Accuracy* (Acc).

Finally, I evaluated the performance for the task of Age estimation in terms of *Mean Absolute Error* (MAE). The is computed as the sum of the absolute differences between the model prediction and the ground truth divided by the total number of utterances considered in the evaluation.

4.3.3 Implementation details

I optimized the trained models weights using the Adam optimization algorithm with a learning rate equal to 1e-4. In order to avoid overfitting on the training data I stop the training if the performance validation set does not improve for 10 consecutive epochs. In multi-objective optimization problems the comparison of solutions is com-

monly done using the Pareto front, but in this case this solution is not feasible to define a stopping criteria. For this reason, I considered the performance on the most difficult task, *i.e.* speaker verification, as a reference metric for the Early Stopping. This choice may be a limitation; it will be analyzed in future work. According to the same criteria, I reduced the learning rate of a multiplicative factor of 0.3 to avoid optimization plateau if the performance on the validation set does not improve for 7 consecutive epochs.

I considered the Cross Entropy loss as the objective function for the tasks of Gender and Emotion recognition. For the age estimation task, I used the MAE loss instead. I also tested Root Mean Square Loss but I have seen that GradNorm does not obtain good results when the norm of one loss is very higher w.r.t. others. Finally, I used ArcFace [189] as the objective function for the speaker verification task. ArcFace proved to overperform the triplet loss for the task of Face Verification and, in addition, it avoids triplet-related problems (*e.g.* hard triplet sampling) [190]. I reported the ArcFace objective function in Equation 4.4.

$$L_{ArcFace} = -log \frac{e^{s\ cos(\theta_{y_i}+m)}}{e^{s\ cos(\theta_{y_i}+m)} + \sum_{j=1,j\neq i}^{N} e^{s\ cos\theta_{y_j}}} \tag{4.4}$$

where θ_{y_j} is the angular distance between the output audio embedding and the reference embedding of the speaker i (also the one who pronounced the utterance) and N is the number of speakers in the training set. The hyper-parameters s and m are the norm scaling factor and the loss margin. As suggested by the reference paper, I set them to 64 and 0.5, respectively.

Table 4.3: Comparison of the results obtained by the proposed multitask models compared to the related single task version and the state of the art. The best results are reported in bold. The results of our models that outperform the state of the art are reported in italic. The multitask models results are labeled with an arrow representing if it outperformed the single task model (↑) or not (↓).

Task	Speaker Verification			Gender Recognition	Emotion Recognition	Age Estimation
Dataset	VoxCeleb1			Common Voice	IEMOCAP	VoxCeleb2
Subset	Standard	Pair E	Pair H	-	-	Enriched Test
Model						
State of the Art	**1.45%**	**1.47%**	**2.72%**	94.32%	56.94%	9.44
ST Gender	-	-	-	*94.69%*	-	-
ST Verification	8.97%	8.97%	16.56%	-	-	-
ST Emotion	-	-	-	-	49.05%	-
ST Age	-	-	-	-	-	10.09
MT Ver A	9.54% ↓	9.69% ↓	16.88% ↓	*95.78%* ↑	55.62% ↑	***8.74*** ↑
MT Ver B	10.30% ↓	10.14% ↓	17.26% ↓	***96.65%*** ↑	54.88% ↑	*8.94* ↑
MT Ver C	9.73% ↓	9.60% ↓	17.09% ↓	*96.40%* ↑	***57.54%*** ↑	*8.86* ↑

4.4 Experimental results

I reported the obtained results in Table 4.3. I report in the first line of the table the state-of-the-art results [184, 186, 188, 191] obtained by supervised models regardless of the method's computational requirement for a fair comparison of the performance. Moreover, in addition to the results of the proposed multitask models, I reported the performance achieved by single-task models characterized by the same convolutional neural network backbone.

The proposed multitask models always outperform the state-of-the-art counterparts at least on two tasks over four and, in the case of the model Version C, on three tasks over four. In particular, on the task of Gender Recognition the multitask models version A, B, and C achieve an accuracy score of 95.78%, 96.65%, and 96.40% compared to the 94.32% of the state-of-art. On the other hand, on the Emotion Recognition task they obtain an accuracy score equal to 55.62%, 54.88%, and 57.54%; in this case, only the multitask model version C outperforms the state-of-art, obtaining an accuracy of 56.94%. The mean average error at the state of the art for the age estimation task is 9.44 years; all the proposed multitask models obtained a better performance achieving a MAE index of 8.74, 8.94, and 8.86, respectively. Finally, the proposed multitask models were not able to achieve results comparable with the state-of-the-art on the task of Speaker Verification, obtaining an average EER equal to 9.7% on the Standard and Pair E test sets of VoxCeleb1 and 17% on the Pair H one; on the other hand, the more complex model ResNext [191] is able to better fit the training data, with an average EER of 1.46% and 2.72% respectively on the two sets. This result can be further justified by the lower amount of training data for the speaker verification task due to the overlap between the VoxCeleb2 and VoxCeleb2 Enriched training and test sets.

Looking at the results obtained by single-task models it can be

seen that only the Gender Recognition model is able to narrowly outperform the state-of-the-art with an accuracy score equal to 94.69%. Comparing the results obtained by single-task models on the related task with multitask ones, it is evident that the multitask learning framework helps to obtain better generalization capabilities with respect to the single-task one. The table shows how the Gender Recognition, Emotion Recognition, and Age Estimation tasks always improve their performance in a multitask setup; on the other hand, the speaker verification performance slightly decreases. I speculate that this result is due to the higher complexity of the considered task w.r.t. others and the need to obtain a good representation for all the tasks given a very simple model (ResNet18). Nevertheless, this behaviour is expected due to the adoption of a light model in combination with a multitask loss function which tries to balance the learning pace of the different tasks.

Comparing the different multitask models, it can be seen as the architecture version A, even if characterized by the lowest amount of parameters, allows to obtain good performance compared to the single-task models. In particular, there is an accuracy improvement of around 1% and 6% on the Gender and Emotion Recognition tasks respectively. Moreover, it obtains an improvement in terms of MAE of around 1.5 years. Moving from architecture version A to version B, we only see an improvement on the Gender Recognition task, reaching state-of-the-art performance with an accuracy score equal to 96.65%. On the other hand, the performance on the other tasks decreases. This result suggests that the correlation between the considered tasks is strong and, therefore, it is important to share features between them. This assumption is further confirmed if we look to the results obtained by architecture version C, which is able to outperform the state-of-the-art models on three tasks over four while combining the increasing of parameters to train with the feature sharing through the Cross Stitch layers.

For a deep understanding of the contribution of the proposed learnable audio representation, I reported in Table 4.4 a comparison of the results obtained by training the multitask architecture with and without the proposed layers. For a fair comparison, I carried out experiments using the Mel Spectrogram representation with the same number of filters as the proposed one. The results show how all the multitask models trained with the proposed representation outperform over all the tasks the Mel counterpart, unless of the version A model which obtains worse results only on the Gender Recognition tasks. The proposed representation allows to gain an average EER improvement of 3% on the speaker verification task and a reduction of the MAE around 1 year for the age estimation one. This performance gain proves that the proposed representation effectively deals with the noise characterizing the VoxCeleb1 and VoxCeleb2 datasets.

4.4.1 Processing time evaluation

Finally, I reported in Figure 4.2 the latency of the proposed multitask architecture compared to the time needed to run all the single-task counterparts on the NVIDIA Jetson AGX embedded system. The latency estimation has been made by averaging 1000 prediction timings over audio signals with lengths equal to 3 seconds (common audio length in social robotics applications). As one can expect, the latency of the multitask model version A (0.369 seconds) is around 1/4 of the time needed to run single-task models (1,489 seconds) while achieving big performance gains on three dealt tasks. A similar behaviour can be seen for the multitask model version B, where the earlier branching causes a latency increasing w.r.t. version A of around 0.167 seconds. Finally, the multitask model version C slightly increases the latency w.r.t. model version B with considerable improvements from the point of view of accuracy. Nevertheless, both models version B and C gain around 1 second if compared to single-task models.

Table 4.4: Comparison of the results obtained by the proposed multitask models trained with the learnable representation and with the Mel Spectrogram. The results of the multitask models trained with the proposed representation are labeled with an arrow representing if it outperformed the Mel Spectrogram one (↑) or not (↓).

Task	Speaker Verification			Gender Recognition	Emotion Recognition	Age Estimation
Dataset	VoxCeleb1			Common Voice	IEMOCAP	VoxCeleb2
Subset	Standard	Pair E	Pair H	-	-	Enriched Test
Model						
MT Ver A (Mel)	12.74%	12.64%	19.39%	96.19%	51.16%	9.40
MT Ver B (Mel)	13.04%	13.21%	20.17%	96.57%	51.27%	9.87
MT Ver C (Mel)	12.60%	12.39%	18.90%	96.19%	55.63%	9.54
MT Ver A	9.54% ↑	9.69% ↑	16.88% ↑	95.78% ↓	55.62% ↑	8.74 ↑
MT Ver B	10.30% ↑	10.14% ↑	17.26% ↑	96.65% ↑	54.88% ↑	8.94 ↑
MT Ver C	9.73% ↑	9.60% ↑	17.09% ↑	96.40% ↑	57.54% ↑	8.86 ↑

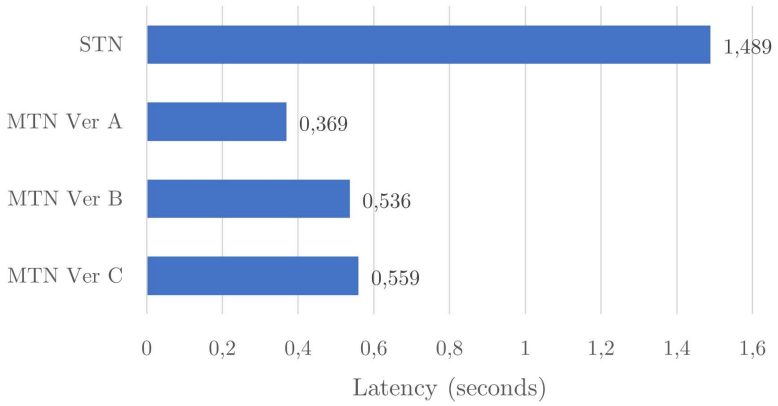


Figure 4.2: Inference latency of the trained multitask networks (MTN Ver A, B, and C) and single-task counterpart (STN). The latency has been computed on an average audio length of 3 seconds over the NVIDIA Jetson AGX Embedded System.

In conclusion, the multitask models offer a good trade-off in terms of accuracy and computational complexity when compared to single-task networks. This is due to their superior accuracy on three out of four tasks, as well as their ability to significantly reduce inference time and memory requirements. Although there is a slight increase of 8% in the task of speaker re-identification, the overall benefits of MTN models make them a suitable choice in social robotics applications.

Chapter 5

Efficient natural language understanding for social robots

Based on

Efficient Transformers for on-robot Natural Language Understanding.
Greco, A., Roberto, A., Saggese, A., & Vento, M. - ICHR 2022.

5.1 Background

Based on the results obtained in Chapter 2 and the state-of-the-art analysis conducted in Chapter 1.3.1, the need to optimize the overall input pipeline, from the perception components up to the conversational ones is evident. This chapter focuses on the Natural Language Understanding module. In its more efficient architecture, the NLU model is composed of a feature extractor transformer and two output layers, respectively for intent and entity recognition. However, this type of neural architecture is computationally heavy due to the quadratic complexity with respect to the input utterance. Moreover, even if it proved to be very accurate over large-scale benchmarks, in a real scenario a social robot has to recognize a specific set of intents and entities designed according to the services it has to provide and the environment in which it works. Therefore, there are no large-scale datasets available to train that transformers models for each possible practical application. In this case, it is common that few people, usually just one person, write several examples of conversations and train the model over these few samples. Unfortunately, these examples are biased by the typical conversation model that those people have in mind and, consequently, it happens that the NLU model is affected by overfitting. It is thus important that NLU models are designed to achieve a good generalization capability even when a large dataset of conversation examples is not available.

For these reasons, in this chapter I investigate the main problems related to the development of NLU models on board of social robots:

(i) the unavailability of data for training the entire model and (ii) the computational and memory constraints of GPU-enabled embedded devices. To this aim, I performed a benchmark analysis by considering four different efficient state-of-the-art transformer architectures for NLU. I carried out experiments on ATIS and SNIPS, two of the most popular NLU datasets [192, 59], in a transfer-learning setup to reproduce the real training conditions. I identified some design strategies justified by the experimental results that allow to better identify the transformer to use so that it is possible to avoid wasting weeks training them all. Finally, I evaluated the impact of the NLU component in the architecture presented in Chapter 2.

5.2 Methodology

This section describes the NLU architectures that I considered in this chapter, namely BERT [193], Distil-BERT [194], ALBERT [195] and SqueezeBERT [196], which are, apart from the reference BERT architecture, the main attempts to reduce the complexity of transformer architectures. I fine-tuned the chosen transformers as a multitask NLU model to reduce the computational complexity of the neural network to deploy on board of the robot. This approach leads to two positive results: first, the number of parameters is halved with respect to using distinct models for the two tasks; moreover, by optimizing the shared parameters, the two tasks can share useful context information and improve their performance.

5.2.1 BERT

The Bidirectional Encoder Representation for Transformers, also known as BERT, is a pre-training strategy proposed in [193]. To date, in the literature, it is common to use BERT to refer to the model that has

been trained using this strategy. The BERT (base) architecture is, in fact, based on the transformer encoder proposed in [197] and is composed of 12 transformer blocks with 12 attention heads, for a total of 110M parameters. BERT is pre-trained by trying to reconstruct an input sentence in which 15% of the words are masked. Differently from task-oriented training previously proposed, BERT directly optimizes in an unsupervised manner the representation coming from the transformer encoder, leading to a better context representation.

5.2.2 Distil-BERT

Distil-BERT [194] is a transformer architecture trained from BERT using the knowledge distillation approach. In this training strategy, a small neural network is trained to reproduce the behaviour of a larger model. In the particular case of Distil-BERT, the transformer architecture is inspired to BERT with three differences: first, the token-type embeddings and the pooler (*i.e.* the task-specific sentence embedding layer) layers have been removed; the number of transformer layers has been reduced of a factor 2; most of the operations, as linear and normalization layers, have been optimized through modern linear algebra tool-kits.

5.2.3 ALBERT

A Lite BERT (ALBERT) [195], is a BERT-like architecture optimized in terms of the number of parameters and memory requirements. The first difference w.r.t. the BERT architecture is within the transformers blocks; in fact, ALBERT shares the parameters in each layer by reducing the number of weights of a factor 9. Moreover, in order to reduce the memory space required by the word embeddings, ALBERT factorizes the embedding matrix. In this way, it reduces the memory requirement from $O(V \times H)$ to $O(V \times E + E \times H)$ where V is the

vocabulary size, H is the size of the embedding and, E is the factorization dimension. Finally, the hidden size of the embeddings within the ALBERT encoder is also reduced from 768 to 128.

5.2.4 SqueezeBERT

SqueezeBERT [196] is a transformer architecture which aims to optimize the BERT performance by adopting grouped convolutions. In particular, SqueezeBERT replaces the three linear layers that are used in the self-attention mechanism with a 1D grouped convolutional layer, configured with 1 as kernel size (equivalent to a linear layer) and 4 groups. In this way, it is able to reduce of a factor 4 the number of arithmetic operations within this layer. The same change is made for the last two layers in the feed-forward network in each transformer block. The general architecture of SqueezeBERT is identical to BERT except for the grouped convolution of the self-attention mechanism and the feed-forward network.

In Table 5.1 I estimated the number of arithmetic operations required by each of the benchmarked models in terms of the number of hidden layers (L) and hidden units (H), the embedding size (E) and the number of attention heads (A). The number of arithmetic operations does not change between BERT and ALBERT that, differently from SqueezeBERT, use the standard self-attention mechanism. On the other hand, the memory required for storing the embeddings of the words varies from 768 floats per word to 128; the memory footprint is further reduced through a factorization approach. Finally, Distil-BERT uses half of the transformer layers with respect to the others.

Table 5.1: Number of hidden layers (L) and hidden units (H), embedding size (E) and number of attention heads (A), which have an impact on the number of operations required by the considered models.

Model	L	H	E	A
BERT	12	768	768	12
Distil-BERT	6	768	768	12
SqueezeBERT	12	768	768	12
AlBERT	12	768	128 (Factorized)	12

5.3 Experimental framework

5.3.1 Datasets

The SNIPS dataset [59] is a NLU benchmark acquired through the SNIPS voice digital assistant. It is composed of 7 intents related to several tasks, like playing a song and book a restaurant. It contains 14,484 sentences equally distributed along the intents and 72 different types of entities. The ATIS dataset [192] is, nowadays, the most popular single-turn benchmark for NLU. It consists of 5,871 sentences concerning flight reservations. The distribution of the intents is strongly unbalanced, increasing the possibility of overfitting. The dataset contains 120 different types of entities.

5.3.2 Implementation details

Large-scale datasets are often unavailable when dealing with social robotics applications; for this reason, the models have been trained through a fine-tuning approach by freezing the weights of the transformer encoder and optimizing task-specific features through 2 additional transformer layers. In this way, it is possible to evaluate the

Table 5.2: Experimental results over SNIPS and ATIS reported in terms of intent accuracy (Int Acc) and entity F1 score (Ent F1 Score). The average latency per sentence on the NVIDIA Jetson Xavier NX is reported together with the relative speedup with respect to the BERT model. The up (down) arrows indicate that a higher (lower) value for that metric corresponds to a better performance score. The different models are sorted in descending order of the number of parameters.

	SNIPS		ATIS				
Transformer	Int Acc ↑	Ent F1 Score ↑	Int Acc ↑	Ent F1 Score ↑	Latency (ms) ↓	Speedup ↑	MParams ↓
BERT (TS)	**0.986**	**0.970**	**0.975**	**0.961**	-	-	110
BERT	0.974	0.922	0.953	0.951	310	1.00x	110
Distil-BERT	0.957	0.933	0.968	0.942	190	1.63x	66
SqueezeBERT	0.966	0.911	0.942	0.948	140	2.21x	51
ALBERT	0.971	0.904	0.968	0.941	**100**	**3.10x**	**12**

generalization capabilities of the pre-trained models when these are adopted in real applications.

The weights of the models have been optimized using Adam and a learning rate equal to 0.001. To deal with the unbalanced distribution of the ATIS dataset, I balanced the number of samples for each intent within a batch. This strategy acts as a sort of regularization to increase the penalty over under-represented intents. To regulate the trade-off between exploration and exploitation of the optimization algorithm, I linearly increased the batch size between 64 and 256 at each epoch, considering a maximum number of epochs equal to 100. Finally, to avoid overfitting, I early stopped the optimization if the performance over the validation set did not improve for 5 consecutive epochs. The hyper-parameters of the fine-tuning model have been optimized through a grid search approach.

The performance of the trained model on the intent classification task has been computed in terms of classification accuracy. Regarding the entity recognition task, the performance metric is the F1 score, which jointly evaluates the precision and recall of the trained models. In order to compute the average latency per sentence of each model, I evaluated the time needed for making inference over the ATIS test set and we divided the results by the number of sentences. The time evaluation has been carried out on a NVIDIA Jetson Xavier NX.

5.4 Experimental results

The results of our experiments are reported in Table 5.2. To carry out a fair comparison with state-of-the-art models based on transformers, I report the results obtained in [54] by training BERT as a joint model. Differently from the experimental setup, all the weights of the model have been optimized (independently) on the two datasets. For this reason, the model is reported as *BERT (TS)*, where TS means Trans-

fer Learning. Comparing only the fine-tuned models, it can be seen that the intent accuracy score has a very low variance, around 1%, without an evident correlation between the number of parameters and the obtained performance. On the other hand, a consideration can be made about the entity recognition task: the BERT model outperforms almost all the other transformers. From this result, one can conclude that a more complex architecture is needed to obtain more discriminative features at the word level to generalize over different entity values. This hypothesis seems to be confirmed whether by observing the results of fine-tuning models with BERT (TS) on the SNIPS dataset (F1 Score of 0.970, compared to 0.933 of Distil-BERT). The SNIPS dataset includes a larger vocabulary and, therefore, the model needs more degrees of freedom (*i.e.* more trainable weights) to learn task-specific representations for a bigger input set.

Regarding the latency on the NVIDIA Jetson embedded system, several considerations can be done. Looking at the Distil-BERT and the SqueezeBERT architectures, which obtain a speedup with respect to BERT of 1.63x and 2.21x respectively, one can notice that even if the former is composed of half layers compared to the latter, SqueezeBERT has a 0.58x speed-up. This result suggests that optimizing the operations within the self-attention mechanism allows to better speed-up the model. The most efficient transformer is ALBERT, with a 3.10x speedup with respect to the heavier BERT. ALBERT has an average inference time of 100 ms, which is the time limit to give the interlocutor the feeling of an instantaneous response [198]. This evidence is surprising due to the fact that ALBERT has been optimized in terms of memory requirements instead of the total number of arithmetic operations. Nevertheless, an explanation can be found: the NVIDIA Jetson Xavier NX has a shared memory between the operating system and the GPU and bigger memory requirements lead to more swap operations and related delays. This assumption is supported by the inverse-proportional relation between the number of parameters and

the speed-up.

It can be concluded that when the dialogues related to a task are characterized by a small vocabulary and a limited number of entity values, a fine-tuning approach can reach results comparable to transfer learning while reducing the probability of overfitting. On the other hand, if the model has to deal with more variegate entities, it is suggested to move to a heavier model to mitigate the performance drop; moreover, if enough data are available, a transfer learning approach is preferred. Regarding the model latency, the feature that mainly affects the inference time of a transformer model over an embedded system is the memory. The memory requirement is very important for social robotics applications since the language model must run in combination with other neural networks like speech recognition and computer vision.

For the purposes of the thesis, I choose the ALBERT architecture to be deployed on the final system. ALBERT achieves a speedup of 3.10x and comparable intent recognition performance with respect to BERT while limiting the entity recognition performance drop to 2%.

5.5 NLU impact evaluation

In order to provide a view of the impact of the experiments on the final systems, I evaluated the average response time of our prototype from the speech signal pronounced by the users to the response of the social robot. Of course, the response time is highly influenced by the design choices done for the other modules present in the processing pipeline (*e.g.*, one can use streaming or batch speech recognition models). Furthermore, the time needed for the computation also depends on the length of the sentence. As anticipated in Chapter 2, the final system is composed of a batch speech recognition model preceded by a Voice Activity Detection (VAD) model. I excluded the time needed

for the VAD to decide that the sentence ended from the response time. Finally, in order to compute the average response time, I considered sentences with a duration of 4 seconds and composed of 10 words, that is the worst case in common social applications.

In this setup, the total amount of time from the users' speech signal to the response is 465 ms; only 21.5% of the time, namely 100 ms, is spent by the NLU module based on the ALBERT model. On the other hand, the NLU module based on BERT required 210 ms more; it means that our analysis allows to reduce the relative impact of the NLU module from (310/675) 45.9% to (100/465) 21.5% with a negligible effect on the performance.

Chapter 6

Conclusions

The main objective of this thesis is the study of social robotics architectures able to provide emphatic interactions with human users. Studies prove that the capability of social robots to personalize the conversation and perceive the interlocutor's emotions are the key behaviours that allow them to be emphatic. To these purposes, social robots rely on multiple sensory modalities to acquire information about their interlocutor robustly and accurately. It is important to point out that this kind of robot commonly operates in challenging environments, *e.g.* dynamic lighting conditions and loud environmental noise. In addition to these challenges, social robots must converse in real-time with humans to give them the feeling of natural interaction. Considering the application context, this result is only possible if the computation is performed on board of the social robot platform, which makes the task harder due to the implicit computational and power constraints.

This thesis tackles these requirements in the context of Deep Learning. In particular, a novel software architecture optimized for multimodal real-time interactions has been proposed as a general-purpose solution for social robots. The realization of a robotic prototype allowed to identify the main issues perceived by humans about state-of-the-art algorithms related to human-robot interaction when deployed together in a real application. In light of this result, in this thesis, I advanced the state-of-the-art by proposing and validating novel auditory and natural language understanding algorithms optimized to be executed on robotic embedded systems while keeping high accuracy.

As a *first contribution*, I proposed a general software architecture for social robots. The proposed architecture includes all the software modules that allow to meet the main requirements of a social robot: first, a dialogue manager able to personalize the human-robot interaction by exploiting the biometrics perceived by the sensors of the social robot; second, a multimodal sensor aggregation module able to exploits the information acquired by different types of sensors to in-

crease the robustness to environmental noise; finally, parallel processing pipelines that, properly designed and implemented, ensure real-time performance. A social robot prototype based on the proposed architecture has been realized and deployed in the SICUREZZA exhibition for three days. 161 people who interacted with the robot evaluated their experience by answering 5 questions with a score between 1 and 5. The maximum score was achieved for more than 40% of the answers and the average rate was between 4 and 5. This result acquires more relevance considering that the people who attended the conference were technically skilled and, therefore, their feedback is reliable. The survey also allowed to investigate the feeling of humans about the performance of the state-of-art algorithms available on the proposed prototype. This analysis results in the need for audio algorithms more robust to environmental noise and more efficient human utterances processing pipelines.

Based on the considerations made above, the *second contribution* of this thesis consists of two learnable audio representations to achieve the following goals: robustness to environmental noise and efficiency.

To achieve the first goal, I propose a novel convolutional layer, namely Denoising-Enhancement Layer (DELayer), able to denoise and enhance the input signal by combining an attention module (DELayer) with the SincNet layer; the final representation is thus able to attenuate the frequency components affected by environmental noise and amplify the ones relevant for the classification. The experimental results demonstrate that the end-to-end model based on the proposed DELayer, namely DENet, is definitely more effective than the existing methods in detecting and recognizing audio events of interest on the MIVIA Audio Events and on the MIVIA Road Events benchmarks. The novel CNN achieves state-of-the-art performance on both datasets and further analyses show its robustness to noise, its stability among different environmental conditions and its generalization capabilities.

The latency evaluation of the DENet model pointed out the need

for a more optimized audio model to be run on CPU devices without batching the predictions and, therefore, increasing the speed of the model itself. For this reason, I also proposed DEGram, a learnable spectrogram-based representation. Being a spectrogram-like representation, DEGram allows the use of shallower CNN with comparable accuracy. Consequently, it reduces the computational requirements of the final system while keeping the robustness of DENet. DEGram is the result of the combination of two novel layers: SincGram and a Time-Frequency version of the DELayer, namely TF-DELayer. The former, like SincNet, is able to learn the frequencies of interest for the audio analysis problem we are dealing with, extracting task-specific features through trainable band-pass filters. The latter can denoise the input signal from environmental background noise using a time-frequency attention mechanism, focusing on the part of the input signal in which the sound of interest is temporally located and on the frequency components not affected by noise. Differently from the previous DELayer, the TF-DELayer uses a squeeze and excitation attention mechanism to drastically reduce the attention overhead. The experimental analysis demonstrated the effectiveness of DEGramNet, an audio CNN based on the proposed DEGram. DEGramNet was able to achieve state-of-the-art results on the VGGSound dataset (Sound Event Classification) and comparable results with a complex Network Architecture Search approach on the VoxCeleb1 dataset (Speaker Identification), proving to be a general-purpose architecture for audio analysis tasks. Moreover, the ablation study proved that the proposed DEGram representation was more effective than Spectrogram, being able to improve the performance of the same CNN architecture trained with both the representations and to achieve better accuracy with respect to deeper models. Finally, the reduction of the number of features computed by DEGram allows the same network to reduce the inference time on CPU architectures, making the model work in real-time on embedded systems CPUs.

As a *third contribution*, I proposed three multitask neural network models able to perform the following tasks simultaneously: Gender Recognition, Emotion Recognition, Age Estimation, and Speaker Re-Identification. I proved that it is possible not only to reduce the computational requirements w.r.t. the single-task counterparts but also to obtain a better generalization of the model itself. The experiment allowed to identify the best architecture to use in very limited setups and when higher accuracy is required. In particular, each multitask architecture is characterized by a different trade-off between feature sharing and model parameters. All the architecture have in common the CNN backbone, *i.e.* ResNet18, and the audio representation, *i.e.* DEGram, that I proved in the previous contribution to be performing and efficient. In this thesis, I also proposed a training loss that allows to take into account different datasets even if they don't have the labels for all the tasks. Moreover, the GradNorm algorithm has been used to avoid unbalancing in the performance of the different tasks. The proposed architectures have been compared with state-of-the-art algorithms and single-task counterparts on standard benchmarks for the considered tasks, *i.e.* VoxCeleb1 and VoxCeleb2 (Speaker Re-Identification and Age Estimation), Mozilla Common Voice (Gender Recognition) and IEMOCAP (Emotion Recognition). The trained models outperformed state-of-art models on three tasks over four proving their effectiveness. Moreover, the multitask approach proved to improve the performance of Gender Recognition, Emotion Recognition, and Age Estimation w.r.t. the single task models, validating the hypothesis that the multitask paradigm not only reduces the computational requirements but also improves the generalization capabilities of the model.

As a *fourth contribution*, I further reduced the response latency of the social robot by identifying the best transformer architecture for Natural Language Understanding in the context of Social Robotics. In particular, I evaluated the trade-off between the accuracy and processing time of different multitask transformer architectures by con-

sidering the problems related to the unavailability of data for training the model with a good degree of generalization and the limited computational and memory resources of GPU-enabled embedded devices. I carried out experiments by comparing transformers characterized by different optimization approaches like knowledge distillation and grouped convolutions. On one hand, I proved that by using transformer models it is possible to achieve good performance even using a fine-tuning approach (performance indices over 90%) and this was not a foregone conclusion. I also demonstrated that the results are highly influenced by the task (*i.e.*, imbalance in the sentences, vocabulary size, and the number of entities). To address this issue, I identified some design strategies that allow to better identify the transformer to use, so that it is possible to avoid wasting weeks training them all. On the other hand, I evaluated the processing time of these transformers over an embedded system commonly used for robotic applications, namely the NVIDIA Jetson Xavier NX. In this way, I added to the previous considerations additional insights to filter out non-suitable models for the architecture and problem at hand. The analysis allowed to reduce the impact of the NLU model on the average response time by around 20-25%.

Overall, in this thesis I realized a real prototype of a Social Robot able to be emphatic with humans through the expression of personalization capabilities. The proposed prototype also allowed to identify the main gaps in state-of-art sensor processing algorithms as they are perceived by humans. In this thesis, I address the main issue by proposing novel audio representations very robust to environmental noise and capable to gain the performance of shallow neural networks commonly running on Social Robots with a negligible computational cost. In addition, I also addressed the optimization of the computational requirements for the audio analysis and NLU components of the social robots by proposing two multitask models able to reduce the computation latency by 75% and 33% w.r.t. earlier models, re-

spectively.

6.1 Outlook

Based on the conclusions of the thesis, here are some possible future research directions:

- Investigating more advanced and sophisticated dialogue managers that can learn and adapt to the user's emotional state, leading to more personalized interactions between the social robot and the human user. This can involve exploring reinforcement learning approaches to train the dialogue manager based on the emotion perceived by the user through the different modalities (i.e. audio, text, and video).

- Further developing the proposed software architecture for social robots to increase its scalability and adaptability to different application contexts. This can involve incorporating new sensory modalities or optimizing existing ones to enhance the robustness of the system.

- Designing and implementing new multimodal fusion techniques that can effectively combine and utilize the information acquired from different sensors, resulting in more accurate and robust detection and recognition of audio and visual events. This can involve exploring techniques such as attention-based fusion, graph-based fusion, or recurrent neural networks.

- Investigating novel audio representation learning techniques that can efficiently process audio signals in challenging environments. This can involve exploring different types of audio representations like, for example, end-to-end time-frequency features learned in the deep network itself.

- Developing novel algorithms and architectures for natural language understanding that can handle more complex and diverse input from human users, such as understanding sarcasm or irony. This can involve exploring techniques such as transfer learning, meta-learning, or active learning to improve the performance of the natural language understanding module.

Bibliography

[1] P. Nemitz, "Constitutional democracy and technology in the age of artificial intelligence," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 376, no. 2133, p. 20180089, 2018.

[2] C. Coombs, D. Hislop, S. K. Taneva, and S. Barnard, "The strategic impacts of intelligent automation for knowledge and service work: An interdisciplinary review," *The Journal of Strategic Information Systems*, vol. 29, no. 4, p. 101600, 2020.

[3] V. S. Gunge and P. S. Yalagi, "Smart home automation: a literature review," *International Journal of Computer Applications*, vol. 975, no. 8887-8891, 2016.

[4] J. S. Edu, J. M. Such, and G. Suarez-Tangil, "Smart home personal assistants: a security and privacy review," *ACM Computing Surveys (CSUR)*, vol. 53, no. 6, pp. 1–36, 2020.

[5] G. Matthews, P. A. Hancock, J. Lin, A. R. Panganiban, L. E. Reinerman-Jones, J. L. Szalma, and R. W. Wohleber, "Evolution and revolution: Personality research for the coming world of robots, artificial intelligence, and autonomous systems," *Personality and individual differences*, vol. 169, p. 109969, 2021.

[6] J. J. Craig, *Introduction to robotics: mechanics and control*. Pearson Educacion, 2005.

[7] Y. Liu, Z. Tian, Y. Liu, J. Li, F. Fu, and J. Bian, "Cognitive modeling for robotic assembly/maintenance task in space exploration," in *Advances in Neuroergonomics and Cognitive Engineering*. Springer International Publishing, 6 2017, pp. 143–153.

[8] R. Alami, R. Chatila, A. Clodic, S. Fleury, M. Herrb, V. Montreuil, and E. A. Sisbot, "Towards human-aware cognitive robots," in *The fifth international cognitive robotics workshop (the AAAI-06 workshop on cognitive robotics)*, 2006.

[9] C. Breazeal, *Designing Sociable Robots*. Cambridge, Mass: The MIT Press, 2004.

[10] A. Henschel, G. Laban, and E. S. Cross, "What makes a robot social? a review of social robots from science fiction to a home or hospital near you," *Current Robotics Reports*, vol. 2, no. 1, pp. 9–19, 2 2021.

[11] B. A. Maxwell, "Building robot systems to interact with people in real environments," *Autonomous Robots*, vol. 22, no. 4, pp. 353–367, 1 2007.

[12] B. P. E. A. Vásquez and F. Matia, "A tour-guide robot: Moving towards interaction with humans," *Engineering Applications of Artificial Intelligence*, vol. 88, p. 103356, 2 2020.

[13] B. R. N. Ramachandran and J. C. Lim, "User validation study of a social robot for use in hospital wards," in *Companion of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 3 2021, pp. 215–219.

[14] P. Pennisi, A. Tonacci, G. Tartarisco, L. Billeci, L. Ruta, S. Gangemi, and G. Pioggia, "Autism and social robotics: A systematic review," *Autism Research*, vol. 9, no. 2, pp. 165–183, 10 2015.

[15] J. López, D. Pérez, E. Zalama, and J. Gómez-Garcia-Bermejo, "BellBot - a hotel assistant system using mobile robots," *International Journal of Advanced Robotic Systems*, vol. 10, no. 1, p. 40, 1 2013.

[16] J. Broekens, M. Heerink, and H. Rosendal, "Assistive social robots in elderly care: a review," *Gerontechnology*, vol. 8, no. 2, pp. 94–103, 4 2009.

[17] V. Vigilante, "Intelligent embedded systems for facial soft biometrics in social robotics," Ph.D. dissertation, University of Salerno, 3 2021.

[18] S. Pleshkova, A. Bekiarski, S. S. Dehkharghani, and K. Peeva, "Perception of audio visual information for mobile robot motion control systems," in *Computer Vision in Control Systems-2*. Springer, 2015, pp. 135–167.

[19] M. A. Lee, Y. Zhu, P. Zachares, M. Tan, K. Srinivasan, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg, "Making sense of vision and touch: Learning multimodal representations for contact-rich tasks," *IEEE Transactions on Robotics*, vol. 36, no. 3, pp. 582–596, 6 2020.

[20] M. McTear, "Conversational AI: Dialogue systems, conversational agents, and chatbots," *Synthesis Lectures on Human Language Technologies*, vol. 13, no. 3, pp. 1–251, 10 2020.

[21] Z. Zhang, R. Takanobu, Q. Zhu, M. Huang, and X. Zhu, "Recent advances and challenges in task-oriented dialog systems," *Science China Technological Sciences*, vol. 63, no. 10, pp. 2011–2027, 9 2020.

[22] Y. Ma, K. L. Nguyen, F. Z. Xing, and E. Cambria, "A survey on empathetic dialogue systems," *Information Fusion*, vol. 64, pp. 50–70, 2020.

[23] A. Greco, A. Roberto, A. Saggese, M. Vento, and V. Vigilante, "Emotion analysis from faces for social robotics," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, IEEE. IEEE, 10 2019, pp. 358–364.

[24] K. Wang, N. An, B. N. Li, Y. Zhang, and L. Li, "Speech emotion recognition using fourier parameters," *IEEE Transactions on affective computing*, vol. 6, no. 1, pp. 69–75, 1 2015.

[25] T. M. Moerland, J. Broekens, and C. M. Jonker, "Emotion in reinforcement learning agents and robots: a survey," *Machine Learning*, vol. 107, no. 2, pp. 443–480, 2018.

[26] B. Xiao, N. Monath, S. Ananthakrishnan, and A. Ravi, "Play duration based user-entity affinity modeling in spoken dialog system," in *Proc. Interspeech 2018*. ISCA, 9 2018, pp. 2057–2061.

[27] C. Gao, W. Lei, X. He, M. de Rijke, and T.-S. Chua, "Advances and challenges in conversational recommender systems: A survey," *AI Open*, vol. 2, pp. 100–126, 2021.

[28] Z.-H. Tan, N. B. Thomsen, X. Duan, E. Vlachos, S. E. Shepstone, M. H. Rasmussen, and J. L. Højvang, "isociobot: a multimodal interactive social robot," *International Journal of Social Robotics*, vol. 10, no. 1, pp. 5–19, 2018.

[29] A. Greco, A. Saggese, M. Vento, and V. Vigilante, "Performance assessment of face analysis algorithms with occluded faces," in *International Conference on Pattern Recognition*, Springer. Springer International Publishing, 2021, pp. 472–486.

[30] L. Luo, W. Huang, Q. Zeng, Z. Nie, and X. Sun, "Learning personalized end-to-end goal-oriented dialog," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01. Association for the Advancement of Artificial Intelligence (AAAI), 7 2019, pp. 6794–6801.

[31] A. Saggese, N. Strisciuglio, M. Vento, and N. Petkov, "Time-frequency analysis for audio event detection in real scenarios," in *2016 13th IEEE international conference on advanced video and signal based surveillance (AVSS)*. IEEE, 2016, pp. 438–443.

[32] Z. Du, L. He, Y. Chen, Y. Xiao, P. Gao, and T. Wang, "Robot cloud: Bridging the power of robotics and cloud computing," *Future Generation Computer Systems*, vol. 74, pp. 337–348, sep 2017.

[33] Z. Ma, Y. Liu, X. Liu, J. Ma, and F. Li, "Privacy-preserving outsourced speech recognition for smart iot devices," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8406–8420, 2019.

[34] C. Savaglio, M. Ganzha, M. Paprzycki, C. Bădică, M. Ivanović, and G. Fortino, "Agent-based internet of things: State-of-the-art and research challenges," *Future Generation Computer Systems*, vol. 102, pp. 1038–1053, 2020.

[35] A. K. Tanwani, R. Anand, J. E. Gonzalez, and K. Goldberg, "RILaaS: Robot inference and learning as a service," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4423–4430, jul 2020.

[36] P. Foggia, A. Greco, G. Percannella, M. Vento, and V. Vigilante, "A system for gender recognition on mobile robots," in *Proceedings of the 2nd International Conference on Applications of Intelligent Systems - APPIS '19*. ACM, 1 2019, pp. 1–6.

[37] J. Gao, M. Galley, and L. Li, *Neural Approaches to Conversational AI*. now Publishers Inc, 2019.

[38] S. J. Young, "Probabilistic methods in spoken–dialogue systems," *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 358, no. 1769, pp. 1389–1402, 4 2000.

[39] T.-H. Wen, D. Vandyke, N. Mrkšić, M. Gasic, L. M. R. Barahona, P.-H. Su, S. Ultes, and S. Young, "A network-based end-to-end trainable task-oriented dialogue system," in *15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017-Proceedings of Conference*, vol. 1. Association for Computational Linguistics, 2017, pp. 438–449.

[40] S. Young, "Still talking to machines (cognitively speaking)," in *Eleventh Annual Conference of the International Speech Communication Association*. ISCA, 9 2010.

[41] E. Simonnet, S. Ghannay, N. Camelin, Y. EstÃ¨ve, and R. D. Mori, "Asr error management for improving spoken language understanding," in *Interspeech 2017*, 2017.

[42] R. Liu, B. Sisman, G. Gao, and H. Li, "Expressive TTS training with frame and style reconstruction loss," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1806–1818, 2021.

[43] W. Shi and Z. Yu, "Sentiment adaptive end-to-end dialog systems," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2018, pp. 1509–1519.

[44] L. Shang, Z. Lu, and H. Li, "Neural responding machine for short-text conversation," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, 2015, pp. 1577–1586.

[45] G. Tur and R. De Mori, *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons, 2011.

[46] S. Louvan and B. Magnini, "Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey," in *Proceedings of the 28th International Conference on Computational Linguistics*. International Committee on Computational Linguistics, 2020, pp. 480–496.

[47] G. Mesnil, X. He, L. Deng, and Y. Bengio, "Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding," in *Interspeech*. ISCA, 8 2013, pp. 3771–3775.

[48] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013.

[49] N. T. Vu, P. Gupta, H. Adel, and H. Schutze, "Bi-directional recurrent neural network with ranking loss for spoken language understanding," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE. IEEE, 3 2016, pp. 6060–6064.

[50] B. Liu and I. Lane, "Attention-based recurrent neural network models for joint intent detection and slot filling," in *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016*. ISCA, 9 2016, pp. 685–689.

[51] T. Saha, N. Priya, S. Saha, and P. Bhattacharyya, "A transformer based multi-task model for domain classification, intent detection and slot-filling," in *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–8.

[52] K. Yao, B. Peng, G. Zweig, D. Yu, X. Li, and F. Gao, "Recurrent conditional random field for language understanding," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE. IEEE, 5 2014, pp. 4077–4081.

[53] X. Zhang and H. Wang, "A joint model of intent determination and slot filling for spoken language understanding," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, 2016, pp. 2993–2999.

[54] Q. Chen, Z. Zhuo, and W. Wang, "Bert for joint intent classification and slot filling," *arXiv preprint arXiv:1902.10909*, 2019.

[55] Y. Wang, Y. Shen, and H. Jin, "A bi-model based rnn semantic frame parsing model for intent detection and slot filling," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 2018, pp. 309–314.

[56] C.-W. Goo, G. Gao, Y.-K. Hsu, C.-L. Huo, T.-C. Chen, K.-W. Hsu, and Y.-N. Chen, "Slot-gated modeling for joint slot filling and intent prediction," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Association for Computational Linguistics, 2018, pp. 753–757.

[57] C. T. Hemphill, J. J. Godfrey, and G. R. Doddington, "The ATIS spoken language systems pilot corpus," in *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*. Association for Computational Linguistics, 1990.

[58] H. Bonneau-Maynard, S. Rosset, C. Ayache, A. Kuhn, and D. Mostefa, "Semantic annotation of the french media dialog corpus," in *Ninth European Conference on Speech Communication and Technology*. ISCA, 9 2005.

[59] A. Coucke, A. Saade, A. Ball, T. Bluche, A. Caulier, D. Leroy, C. Doumouro, T. Gisselbrecht, F. Caltagirone, T. Lavril *et al.*, "Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces," *arXiv preprint arXiv:1805.10190*, 2018.

[60] E. Bastianelli, A. Vanzo, P. Swietojanski, and V. Rieser, "Slurp: A spoken language understanding resource package," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 7252–7262.

[61] S. Schuster, S. Gupta, R. Shah, and M. Lewis, "Cross-lingual transfer learning for multilingual task oriented dialog," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, 2019, pp. 3795–3805.

[62] J. D. Williams, A. Raux, and M. Henderson, "The dialog state tracking challenge series: A review," *Dialogue & Discourse*, vol. 7, no. 3, pp. 4–33, 4 2016.

[63] M. Kotti, V. Diakoloukas, A. Papangelis, M. Lagoudakis, and Y. Stylianou, "A case study on the importance of belief state representation for dialogue policy management," in *Interspeech 2018, 19th Annual Conference of the International Speech Communication Association, Hyderabad, India, 2-6 September 2018*. ISCA, 9 2018, pp. 986–990.

[64] M. Henderson, "Machine learning for dialog state tracking: A review," in *Proceedings of The First International Workshop on Machine Learning in Spoken Language Processing*, 2015.

[65] N. Mrkšić, D. Ó. Séaghdha, T.-H. Wen, B. Thomson, and S. Young, "Neural belief tracker: Data-driven dialogue state tracking," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2017, pp. 1777–1788.

[66] C.-S. Wu, A. Madotto, E. Hosseini-Asl, C. Xiong, R. Socher, and P. Fung, "Transferable multi-domain state generator for task-oriented dialogue systems," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019, pp. 808–819.

[67] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski,

S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2 2015.

[68] Z. Lipton, X. Li, J. Gao, L. Li, F. Ahmed, and L. Deng, "Bbq-networks: Efficient exploration in deep reinforcement learning for task-oriented dialogue systems," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[69] B. Peng, X. Li, L. Li, J. Gao, A. Celikyilmaz, S. Lee, and K.-F. Wong, "Composite task-completion dialogue policy learning via hierarchical deep reinforcement learning," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2017, pp. 2231–2240.

[70] J. O. S. T. SCHATZMANN, K. A. R. L. WEILHAMMER, M. A. T. T. STUTTLE, and S. T. E. V. E. YOUNG, "A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies," *The knowledge engineering review*, vol. 21, no. 2, pp. 97–126, 2006.

[71] S. Jung, C. Lee, K. Kim, M. Jeong, and G. G. Lee, "Data-driven user simulation for automated evaluation of spoken dialog systems," *Computer Speech & Language*, vol. 23, no. 4, pp. 479–509, 10 2009.

[72] X. Li, Z. C. Lipton, B. Dhingra, L. Li, J. Gao, and Y.-N. Chen, "A user simulator for task-completion dialogues," *arXiv preprint arXiv:1612.05688*, 2016.

[73] L. E. Asri, J. He, and K. Suleman, "A sequence-to-sequence model for user simulation in spoken dialogue systems," in *Proc. Interspeech 2016*. ISCA, 9 2016, pp. 1151–1155.

[74] J. D. Williams, K. Asadi, and G. Zweig, "Hybrid code networks: practical and efficient end-to-end dialog control with supervised and

reinforcement learning," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2017, pp. 665–677.

[75] M. Henderson, I. Vulić, D. Gerz, I. Casanueva, P. Budzianowski, S. Coope, G. Spithourakis, T.-H. Wen, N. Mrkšić, and P.-H. Su, "Training neural response selection for task-oriented dialogue systems," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019, pp. 5392–5404.

[76] J. Deriu, A. Rodrigo, A. Otegi, G. Echegoyen, S. Rosset, E. Agirre, and M. Cieliebak, "Survey on evaluation methods for dialogue systems," *Artificial Intelligence Review*, vol. 54, no. 1, pp. 755–810, 2021.

[77] S. Young, C. Breslin, M. Gašić, M. Henderson, D. Kim, M. Szummer, B. Thomson, P. Tsiakoulis, and E. T. Hancock, "Evaluation of statistical POMDP-based dialogue systems in noisy environments," in *Situated Dialog in Speech-Based Human-Computer Interaction*. Springer International Publishing, 2016, pp. 3–14.

[78] M. A. Walker, D. J. Litman, C. A. Kamm, and A. Abella, "PARADISE," in *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 1997, pp. 271–280.

[79] M. Valera and S. Velastin, "Intelligent distributed surveillance systems: a review," *IEE Proceedings-Vision, Image and Signal Processing*, vol. 152, no. 2, p. 192, 2005.

[80] H. Yan, M. H. Ang, and A. N. Poo, "A survey on perception methods for human–robot interaction in social robots," *International Journal of Social Robotics*, vol. 6, no. 1, pp. 85–119, 2014.

[81] A. Mesaros, T. Heittola, T. Virtanen, and M. D. Plumbley, "Sound event detection: A tutorial," *IEEE Signal Processing Magazine*, vol. 38, no. 5, pp. 67–83, 2021.

[82] H. Mahdi, S. A. Akgun, S. Saleh, and K. Dautenhahn, "A survey on the design and evolution of social robots-past, present and future," *Robotics and Autonomous Systems*, p. 104193, 2022.

[83] A. Mathur, A. Isopoussu, F. Kawsar, N. Berthouze, and N. D. Lane, "Mic2mic: Using cycle-consistent generative adversarial networks to overcome microphone variability in speech systems," in *Proceedings of the 18th International Conference on Information Processing in Sensor Networks*. ACM, 4 2019, pp. 169–180.

[84] T. Wan, Y. Zhou, Y. Ma, and H. Liu, "Noise robust sound event detection using deep learning and audio enhancement," in *2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, IEEE. IEEE, 12 2019, pp. 1–5.

[85] W. Xie, A. Nagrani, J. S. Chung, and A. Zisserman, "Utterance-level aggregation for speaker recognition in the wild," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5791–5795.

[86] A. Greco, A. Saggese, M. Vento, and V. Vigilante, "Sorenet: a novel deep network for audio surveillance applications," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, IEEE. IEEE, 10 2019, pp. 546–551.

[87] H. Purwins, B. Li, T. Virtanen, J. Schluter, S.-Y. Chang, and T. Sainath, "Deep learning for audio signal processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 2, pp. 206–219, 5 2019.

[88] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[89] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational analysis of sound scenes and events.* Springer, 2018.

[90] A.-L. Giraud, C. Lorenzi, J. Ashburner, J. Wable, I. Johnsrude, R. Frackowiak, and A. Kleinschmidt, "Representation of the temporal envelope of sounds in the human brain," *Journal of neurophysiology*, vol. 84, no. 3, pp. 1588–1598, 2000.

[91] M. Kos, Z. Kačič, and D. Vlaj, "Acoustic classification and segmentation using modified spectral roll-off and variance-based features," *Digital Signal Processing*, vol. 23, no. 2, pp. 659–674, 2013.

[92] A. Graps, "An introduction to wavelets," *IEEE computational science and engineering*, vol. 2, no. 2, pp. 50–61, 1995.

[93] S. Furui, "Speaker-independent isolated word recognition based on emphasized spectral dynamics," in *ICASSP'86. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 11. IEEE, 1986, pp. 1991–1994.

[94] Z. Liu, J. Huang, Y. Wang, and T. Chen, "Audio feature extraction and analysis for scene classification," in *Proceedings of First Signal Processing Society Workshop on Multimedia Signal Processing.* IEEE, 1997, pp. 343–348.

[95] R. N. Bracewell and R. N. Bracewell, *The Fourier transform and its applications.* McGraw-Hill New York, 1986, vol. 31999.

[96] D. P. Mitchell and A. N. Netravali, "Reconstruction filters in computer-graphics," *ACM Siggraph Computer Graphics*, vol. 22, no. 4, pp. 221–228, 1988.

[97] P. Podder, T. Z. Khan, M. H. Khan, and M. M. Rahman, "Comparative performance analysis of hamming, hanning and blackman window," *International Journal of Computer Applications*, vol. 96, no. 18, 2014.

[98] F. Phan, E. Micheli-Tzanakou, and S. Sideman, "Speaker identification using neural networks and wavelets," *IEEE Engineering in Medicine and Biology Magazine*, vol. 19, no. 1, pp. 92–101, 2000.

[99] P. Wei, F. He, L. Li, and J. Li, "Research on sound classification based on svm," *Neural Computing and Applications*, vol. 32, no. 6, pp. 1593–1607, 2020.

[100] C. Junli and J. Licheng, "Classification mechanism of support vector machines," in *WCC 2000-ICSP 2000. 2000 5th International Conference on Signal Processing Proceedings. 16th World Computer Congress 2000*, vol. 3. IEEE, 2000, pp. 1556–1559.

[101] E. Fosler-Lussier, "Markov models and hidden markov models: A brief tutorial," *International Computer Science Institute*, 1998.

[102] S. Laqtib, K. E. Yassini, and M. L. Hasnaoui, "A deep learning methods for intrusion detection systems based machine learning in manet," in *Proceedings of the 4th International Conference on Smart City Applications*, 2019, pp. 1–8.

[103] N. Strisciuglio, M. Vento, and N. Petkov, "Learning representations of sound using trainable COPE feature extractors," *Pattern Recognition*, vol. 92, pp. 25–36, 8 2019.

[104] D. Yu and L. Deng, "Deep learning and its applications to signal and information processing [exploratory dsp]," *IEEE Signal Processing Magazine*, vol. 28, no. 1, pp. 145–154, 2010.

[105] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M. P. Reyes, M.-L. Shyu, S.-C. Chen, and S. S. Iyengar, "A survey on deep learning," *ACM Computing Surveys (CSUR)*, vol. 51, no. 5, pp. 1–36, 2019.

[106] L. Torrey and J. Shavlik, "Transfer learning," in *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. IGI Global, 2010, pp. 242–264.

[107] R. Tolimieri and M. An, *Time-frequency representations.* Springer Science & Business Media, 1997.

[108] M. Espi, M. Fujimoto, K. Kinoshita, and T. Nakatani, "Exploiting spectro-temporal locality in deep learning based acoustic event detection," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2015, no. 1, pp. 1–12, 2015.

[109] D. Gabor, "Theory of communication. part 1: The analysis of information," *Journal of the Institution of Electrical Engineers-part III: radio and communication engineering*, vol. 93, no. 26, pp. 429–441, 1946.

[110] A. Satt, S. Rozenberg, and R. Hoory, "Efficient emotion recognition from speech using deep learning on spectrograms." in *Interspeech*, 2017, pp. 1089–1093.

[111] Y. Zeng, H. Mao, D. Peng, and Z. Yi, "Spectrogram based multi-task audio classification," *Multimedia Tools and Applications*, vol. 78, no. 3, pp. 3705–3722, 2019.

[112] I. Nelken, "Processing of complex sounds in the auditory system," *Current opinion in neurobiology*, vol. 18, no. 4, pp. 413–417, 2008.

[113] A. Natsiou and S. O'Leary, "Audio representations for deep learning in sound synthesis: A review," in *2021 IEEE/ACS 18th International Conference on Computer Systems and Applications (AICCSA)*. IEEE, 2021, pp. 1–8.

[114] S. S. Stevens and J. Volkmann, "The relation of pitch to frequency: A revised scale," *The American Journal of Psychology*, vol. 53, no. 3, pp. 329–353, 1940.

[115] V. Hohmann, "Frequency analysis and synthesis using a gammatone filterbank," *Acta Acustica united with Acustica*, vol. 88, no. 3, pp. 433–442, 2002.

[116] R. D. Patterson, I. Nimmo-Smith, J. Holdsworth, and P. Rice, "An efficient auditory filterbank based on the gammatone function," in *a meeting of the IOC Speech Group on Auditory Modelling at RSRE*, vol. 2, no. 7, 1987.

[117] K. Palanisamy, D. Singhania, and A. Yao, "Rethinking cnn models for audio classification," *arXiv preprint arXiv:2007.11154*, 2020.

[118] A. Greco, A. Roberto, A. Saggese, and M. Vento, "Which are the factors affecting the performance of audio surveillance systems?" in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 7876–7883.

[119] K. Choi, G. Fazekas, M. Sandler, and K. Cho, "Convolutional recurrent neural networks for music classification," in *2017 IEEE International conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2017, pp. 2392–2396.

[120] T. Kim, J. Lee, and J. Nam, "Sample-level CNN architectures for music auto-tagging using raw waveforms," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Neural Information Processing Systems (NIPS). IEEE, apr 2018.

[121] J. Naranjo-Alcazar, S. Perez-Castanos, I. Martin-Morato, P. Zuccarello, F. J. Ferri, and M. Cobos, "A comparative analysis of residual block alternatives for end-to-end audio classification," *IEEE Access*, vol. 8, pp. 188 875–188 882, 2020.

[122] T. Kim, J. Lee, and J. Nam, "Comparison and analysis of SampleCNN architectures for audio classification," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 2, pp. 285–297, may 2019.

[123] Z. Zhu, J. H. Engel, and A. Hannun, "Learning multiscale features directly from waveforms," *Interspeech 2016*, pp. 1305–1309, 2016.

[124] J. Lee, J. Park, L. Kim, and J. Nam, "Sample-level deep convolutional neural networks for music auto-tagging using raw waveforms," in *The 14th Sound and Music Computing Conference*. SMCNetwork, 2017.

[125] R. Zazo, P. S. Nidadavolu, N. Chen, J. Gonzalez-Rodriguez, and N. Dehak, "Age estimation in short speech utterances based on lstm recurrent neural networks," *IEEE Access*, vol. 6, pp. 22 524–22 530, 2018.

[126] R. S. Alkhawaldeh, "Dgr: gender recognition of human speech using one-dimensional conventional neural network," *Scientific Programming*, vol. 2019, 2019.

[127] Z. Huang, M. Dong, Q. Mao, and Y. Zhan, "Speech emotion recognition using cnn," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 801–804.

[128] A. Roberto, A. Saggese, and M. Vento, "A challenging voice dataset for robotic applications in noisy environments," in *International Conference on Computer Analysis of Images and Patterns*. Springer, 2019, pp. 354–364.

[129] R. Caruana, "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997.

[130] S. Wang, S. Yin, L. Hao, and G. Liang, "Multi-task face analyses through adversarial learning," *Pattern Recognition*, vol. 114, p. 107837, 2021.

[131] Z. Tang, L. Li, and D. Wang, "Multi-task recurrent model for speech and speaker recognition," in *2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*. IEEE, 2016, pp. 1–4.

[132] M. Firdaus, H. Golchha, A. Ekbal, and P. Bhattacharyya, "A deep multi-task model for dialogue act classification, intent detection and slot filling," *Cognitive Computation*, vol. 13, no. 3, pp. 626–645, 2021.

[133] R. Lotfian and C. Busso, "Predicting categorical emotions by jointly learning primary and secondary emotions through multitask learning," *Interspeech 2018*, 2018.

[134] A. Montalvo, J. R. Calvo, and J.-F. Bonastre, "Multi-Task Learning for Voice Related Recognition Tasks," in *Proc. Interspeech 2020*, 2020, pp. 2997–3001.

[135] W. Ding and L. He, "MTGAN: Speaker Verification through Multitasking Triplet Generative Adversarial Networks," in *Proc. Interspeech 2018*, 2018, pp. 3633–3637.

[136] L. Deng and X. Li, "Machine learning paradigms for speech recognition: An overview," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 5, pp. 1060–1089, 2013.

[137] S. Parthasarathy and C. Busso, "Jointly predicting arousal, valence and dominance with multi-task learning." in *Interspeech*, vol. 2017, 2017, pp. 1103–1107.

[138] D. Le, Z. Aldeneh, and E. M. Provost, "Discretized continuous speech emotion recognition with multi-task deep recurrent neural network." in *Interspeech*, 2017, pp. 1108–1112.

[139] J. Kim, G. Englebienne, K. P. Truong, and V. Evers, "Towards speech emotion recognition" in the wild" using aggregated corpora and deep multi-task learning," in *18th Annual Conference of the International Speech Communication Association, INTERSPEECH 2017: Situated interaction*. International Speech Communication Association (ISCA), 2017, pp. 1113–1117.

[140] C. Luu, P. Bell, and S. Renals, "Leveraging Speaker Attribute Information Using Multi Task Learning for Speaker Verification and Diarization," in *Proc. Interspeech 2021*, 2021, pp. 491–495.

[141] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich, "Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks," in *International Conference on Machine Learning.* PMLR, 2018, pp. 794–803.

[142] J. Zhang, Y. Peng, V. T. Pham, H. Xu, H. Huang, and E. S. Chng, "E2E-Based Multi-Task Learning Approach to Joint Speech and Accent Recognition," in *Proc. Interspeech 2021*, 2021, pp. 1519–1523.

[143] S. Sigtia, E. Marchi, S. Kajarekar, D. Naik, and J. Bridle, "Multi-task learning for speaker verification and voice trigger detection," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2020, pp. 6844–6848.

[144] N. Chen, Y. Qian, and K. Yu, "Multi-task learning for text-dependent speaker verification," in *Proc. Interspeech 2015*, 2015, pp. 185–189.

[145] M. Sarma, K. K. Sarma, and N. K. Goel, "Multi-task learning dnn to improve gender identification from speech leveraging age information of the speaker," *International Journal of Speech Technology*, vol. 23, no. 1, pp. 223–240, 2020.

[146] Y. Pan and W.-Q. Zhang, "Multi-task learning based end-to-end speaker recognition," in *Proceedings of the 2019 2nd International Conference on Signal Processing and Machine Learning*, 2019, pp. 56–61.

[147] Y. Liu, L. He, J. Liu, and M. T. Johnson, "Speaker Embedding Extraction with Phonetic Information," in *Proc. Interspeech 2018*, 2018, pp. 2247–2251.

[148] H. Oliff, Y. Liu, M. Kumar, M. Williams, and M. Ryan, "Reinforcement learning for facilitating human-robot-interaction in manufacturing," *Journal of Manufacturing Systems*, vol. 56, pp. 326–340, 2020.

[149] M. Colledanchise and P. Ögren, *Behavior trees in robotics and AI: An introduction.* CRC Press, 2018.

[150] "Ultra-lightweight Face Detection RFB 320," https://github.com/Linzaer/Ultra-Light-Fast-Generic-Face-Detector-1MB.

[151] A. Greco, A. Saggese, M. Vento, and V. Vigilante, "Gender recognition in the wild: a robustness evaluation over corrupted images," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 12, pp. 10 461–10 472, 2021.

[152] ——, "Effective training of convolutional neural networks for age estimation based on knowledge distillation," *Neural Computing and Applications*, pp. 1–16, 2021.

[153] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," *arXiv preprint arXiv:2107.08430*, 2021.

[154] R. Di Lascio, P. Foggia, G. Percannella, A. Saggese, and M. Vento, "A real time algorithm for people tracking using contextual reasoning," *Computer Vision and Image Understanding*, vol. 117, no. 8, pp. 892–908, 2013.

[155] J. Wiseman, "WebRTC VAD," https://pypi.org/project/webrtcvad/.

[156] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented Transformer for Speech Recognition," in *Proc. Interspeech 2020*, 2020, pp. 5036–5040.

[157] T. Bunk, D. Varshneya, V. Vlasov, and A. Nichol, "Diet: Lightweight language understanding for dialogue systems," *arXiv preprint arXiv:2004.09936*, 2020.

[158] V. Vlasov, J. E. Mosig, and A. Nichol, "Dialogue transformers," *arXiv preprint arXiv:1910.00486*, 2019.

[159] M. Ravanelli and Y. Bengio, "Speaker recognition from raw waveform with sincnet," in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 12 2018.

[160] A. Roberto, A. Saggese, and M. Vento, "A deep convolutionary network for automatic detection of audio events," in *Proceedings of the 3rd International Conference on Applications of Intelligent Systems*, 2020, pp. 1–6.

[161] P. Foggia, A. Saggese, N. Strisciuglio, M. Vento, and N. Petkov, "Car crashes detection by audio analysis in crowded roads," in *2015 12th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, aug 2015, pp. 1–6.

[162] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, "Reliable detection of audio events in highly noisy environments," *Pattern Recognition Letters*, vol. 65, pp. 22–28, nov 2015.

[163] Y. Aytar, C. Vondrick, and A. Torralba, "Soundnet: Learning sound representations from unlabeled video," in *Advances in neural information processing systems*, 2016, pp. 892–900.

[164] P. Foggia, A. Saggese, N. Strisciuglio, M. Vento, and V. Vigilante, "Detecting sounds of interest in roads with deep networks," in *Image Analysis and Processing - ICIAP 2019*, E. Ricci, S. Rota Bulò, C. Snoek, O. Lanz, S. Messelodi, and N. Sebe, Eds. Cham: Springer International Publishing, 2019, pp. 583–592.

[165] V. Carletti, P. Foggia, G. Percannella, A. Saggese, N. Strisciuglio, and M. Vento, "Audio surveillance using a bag of aural words classifier," in *IEEE Int. Conf. on Advanced Video and Signal Based Surveillance (AVSS)*, Aug 2013, pp. 81–86.

[166] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, "Audio surveillance of roads: A system for detecting anomalous sounds," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 1, pp. 279–288, jan 2016.

[167] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "ECA-net: Efficient channel attention for deep convolutional neural networks,"

in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2020.

[168] S. Butterworth, "On the theory of filter amplifiers," *Experimental Wireless & the Wireless Engineer*, vol. 7, no. 6, pp. 536–541, 1930.

[169] A. G. Roy, N. Navab, and C. Wachinger, "Concurrent spatial and channel 'squeeze & excitation' in fully convolutional networks," in *International conference on medical image computing and computer-assisted intervention*. Springer, 2018, pp. 421–429.

[170] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[171] ——, "Identity mappings in deep residual networks," in *European conference on computer vision*. Springer, 2016, pp. 630–645.

[172] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[173] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[174] H. Chen, W. Xie, A. Vedaldi, and A. Zisserman, "Vggsound: A large-scale audio-visual dataset," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE. IEEE, may 2020, pp. 721–725.

[175] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," *Telephony*, vol. 3, pp. 33–039, 2017.

[176] C. Buckley and E. M. Voorhees, "Retrieval evaluation with incomplete information," in *Proceedings of the 27th annual international*

conference on Research and development in information retrieval - SIGIR '04. ACM Press, 2004, pp. 25–32.

[177] T. Fawcett, "Roc graphs: Notes and practical considerations for researchers," *Machine learning*, vol. 31, no. 1, pp. 1–38, 2004.

[178] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Interspeech 2019*. ISCA, sep 2019, pp. 2613–2617.

[179] E. Kazakos, A. Nagrani, A. Zisserman, and D. Damen, "Slow-fast auditory streams for audio recognition," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE. IEEE, jun 2021, pp. 855–859.

[180] S. Ding, T. Chen, X. Gong, W. Zha, and Z. Wang, "AutoSpeech: Neural architecture search for speaker recognition," in *Interspeech 2020*. ISCA, oct 2020, pp. 916–920. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2020-1258

[181] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3994–4003.

[182] A. B. Zadeh, P. P. Liang, S. Poria, E. Cambria, and L.-P. Morency, "Multimodal language analysis in the wild: Cmu-mosei dataset and interpretable dynamic fusion graph," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 2236–2246.

[183] C. Busso, M. Bulut, C.-C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J. N. Chang, S. Lee, and S. S. Narayanan, "Iemocap: Interactive emotional dyadic motion capture database," *Language resources and evaluation*, vol. 42, no. 4, pp. 335–359, 2008.

[184] X. Wu, S. Hu, Z. Wu, X. Liu, and H. Meng, "Neural architecture search for speech emotion recognition," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 6902–6906.

[185] R. Ardila, M. Branson, K. Davis, M. Kohler, J. Meyer, M. Henretty, R. Morais, L. Saunders, F. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," in *Proceedings of the 12th Language Resources and Evaluation Conference*, 2020, pp. 4218–4222.

[186] K. Chachadi and S. Nirmala, "Voice-based gender recognition using neural network," in *Information and Communication Technology for Competitive Strategies (ICTCS 2020)*. Springer, 2022, pp. 741–749.

[187] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," *Proc. Interspeech 2018*, pp. 1086–1090, 2018.

[188] K. Hechmi, T. N. Trong, V. Hautamäki, and T. Kinnunen, "Voxceleb enrichment for age and gender recognition," in *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2021, pp. 687–693.

[189] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4690–4699.

[190] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[191] T. Zhou, Y. Zhao, and J. Wu, "Resnext and res2net structures for speaker verification," in *2021 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2021, pp. 301–307.

[192] P. J. Price, "Evaluation of spoken language systems," in *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*, 1990.

[193] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[194] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.

[195] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," in *International Conference on Learning Representations*, 2019.

[196] F. Iandola, A. Shaw, R. Krishna, and K. Keutzer, "Squeezebert: What can computer vision teach nlp about efficient neural networks?" in *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*, 2020, pp. 124–135.

[197] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, 2017, pp. 5998–6008. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[198] R. B. Miller, "Response time in man-computer conversational transactions," in *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*, 1968, pp. 267–277.