UNIVERSITAT DE BARCELONA

FUNDAMENTALS OF DATA SCIENCE MASTER'S THESIS

Non-acted Multi-view Audio-Visual Dyadic Interactions Project

Master thesis: Multitask Learning for Facial Attributes Analysis

---

Author:
Andreu Masdeu Ninot

Supervisor:
Dr. Sergio Escalera
Cristina Palmero
Dr. Julio C. S. Jacques Junior

A thesis submitted in partial fulfillment of the requirements for the degree of MSc in Fundamentals of Data Science

in the

Facultat de Matemàtiques i Informàtica

September 2, 2019

UNIVERSITAT DE BARCELONA

Abstract

Facultat de Matemàtiques i Informàtica

MSc

Non-acted Multi-view Audio-Visual Dyadic Interactions Project

Master thesis: Multitask Learning for Facial Attributes Analysis

by Andreu Masdeu Ninot

Abstract of the project:

Socially-intelligent systems have to be capable of accurately perceiving and inferring the personality and other particularities of different individuals, so as to provide a more effective, empathic, and natural tailored communication. To embody this human likeness into such systems, it is imperative to have a deeper understanding of real human-human interactions first, to computationally model both individual behavior and interpersonal interinfluence. However, there is a lack of publicly-available audiovisual databases of non-acted face-to-face dyadic interactions, which cover the richness and complexity of social communications in real life. In this project, we collected the first of its kind non-acted audio-vidual multi-view dataset of dyadic interactions. The main goals of this dataset and associated research is to analyze human communication from a multidisciplinary perspective (i.e. technological, sociological and psychological) and to research and implement new paradigms and technologies of interpersonal behavior understanding. It is expected to move beyond automatic individual behavior detection and focus on the development of automatic approaches to study and understand the mechanisms of perception of and adaptation to verbal and non-verbal social signals in dyadic interactions, taking into account individual and dyad characteristics. In addition to the collection of more than 80 hours of dyadic interactions including 150 participants performing cognitive tasks designed by the psychologists, this project performed a proof of concept analysis of different technical challenges included in the database: -Setup design, calibration and synchronization of 6 HD cameras, 2 HD egocentric cameras, 2 wrist heart rate monitors, 2 lapel microphones, and 1 ambient microphone -Multi-view joint optimization of hand and body skeleton poses for enhanced hand and body pose recovery -Speaker audio segmentation -Audio-visual spatio-temporal modeling of human emotions -Multi-task face attributes analysis The different contributions are presented and justified in the context of their respective state- of-the-art, evaluated on proper public datasets, and finally tested as a proof of concept evaluation on the recently designed dyadic dataset. Detailed discussion of the implemented work and its associated future research is provided.

Abstract of this Master thesis

In this thesis we explore the use of Multitask Learning for improving performance in facial attributes tasks such as gender, age and ethnicity prediction. These tasks, along with emotion recognition will be part of a new dyadic interaction dataset which was recorded during the development of this thesis. This work includes the implementation of two state of the art multitask deep learning models and the discussion of the results obtained from these methods in a preliminary dataset, as well as a first evaluation in a sample of the dyadic interaction dataset. This will serve as a baseline for a future implementation of Multitask Learning methods in the fully annotated dyadic interaction dataset.

Master project student contribution

The contribution of this master thesis within the whole project is the study of the state of the art in multitask deep learning, specially for face attributes analysis. Training and analysis of two state of the art multitask learning architectures on an external dataset and a cross-database evaluation of these trained models in the Dyadic Interaction Dataset. Comparision between multitask learning models and with single-task learning models. Help and participation during the recordings of the different sessions of the Face-to-face Dyadic Interaction Dataset placing and collecting the setup and attending the participants. Also collaboration in the annotation of this database labelling the emotions of the participants.

Acknowledgements

I would like to thank my supervisors Sergio, Cristina and Julio for his dedication and the opportunity to work in this amazing project. Also, I would like to thank the other members of the research group, specially Javi and Alexa who have put an enormous dedication to this project. And to my colleagues: Aleix, Pablo and Rubén with whom we have shared very special moments together through the development of our thesis.

Also, I would like to thank my family for their support and participation in the recordings, and for always being there. Also, thanks to all the friends who have participated or tried to in the recordings of the dataset. Finally, but not less important, I would like to thank Maria for her constant support.

Contents

Chapter 1

Introduction

In the computer vision and machine learning communities, one ongoing line of research is modelling human interaction and behaviour. Social signal processing and affective computing are two research fields that aim at understanding these interactions by extracting audio-visual features and combining them to build bigger constructs such as personality or dominance. In particular, **dyadic interactions** between individuals is a crucial aspect in studying how human beings react to the environment and with each other.

Exploring and analyzing dyadic sessions may result in a better understanding of human interactions and behaviour. But as for every purpose or plan intended in machine learning there is always one mandatory thing to have: **data**. In particular, data for dyadic interactions between humans is quite rare to find, for instance, there is the **IEMOCAP** dataset [19]. However, data for this sort of interaction analyzed from a psychological perspective does not exist.

One goal of this master thesis was precisely to assist in the creation of a **dyadic interaction dataset** consisting of videos of sessions of dyadic interactions between individuals in different scenarios. Different attributes were captured and labelled from these videos: body pose, hand pose, emotion, age, gender etc. Furthermore, all participants of the dataset were given a personality test survey, which means that each individual personality was labelled. This represented a very ambitious challenge and makes this dataset a very unique object which may be very helpful for the computer vision and machine learning communities in the future.

In particular, many attributes can be extracted from the data. One specific area of interest containing multiple social signs and attributes are the **faces** of the individuals. From analyzing the faces of the participants we can extract their age, gender, ethnicity and emotions. Furthermore, all these attributes seem to be correlated somehow. Therefore, creating models that learn to perform all these tasks jointly seems like a reasonable idea.

From that idea came the second and main goal of my part in this group master thesis: exploring the use of multitask learning architectures for the facial analysis of the aforementioned dataset. **Multitask learning** (MTL) aims to improve the performance on the multiple tasks defined compared to the performance we would obtain if those tasks were learned individually. Therefore, the objective of this work is to show that multitask learning makes sense and does improve the performance of the face analysis tasks which we defined on this dataset.

This master thesis includes a discussion on the creation and contents of the dataset, a chapter discussing the current state of the art in MTL, a chapter describing the multitask learning methods chosen, a chapter analyzing the results obtained using an experimental dataset and a sample of the dyadic interaction dataset and finally a chapter containing the conclusions we reached.

Chapter 2

Dyadic Dataset

2.1 Introduction

Technology providers must endow **socially-intelligent** machines with the capacity of understanding and adapting to different social contexts and individuals, so as to provide a more empathetic, inclusive, tailored communication. To do so, it is necessary to train such systems with computational models that capture the richness and complexity of natural human-human interactions. The Face-to-face Dyadic Interaction Dataset project aims at **analyzing human communication, from a multidisciplinary perspective**, to research and implement new paradigms and technologies of **interpersonal behavior understanding**, by means of a novel annotated database of dyadic face-to-face spontaneous interactions. The database, which would be publicly available for the research community in compliance with GDPR, will consist of audio-visual recordings, personality profiling and sociodemographic data of diverse user populations. The project will entail the development of new automatic models of interpersonal influence understanding and personality traits regression from interaction social signals using novel deep learning techniques, with the ultimate goal of demonstrating the feasibility of developing socially-aware systems able to Face-to-face Dyadic Interaction Dataset the real personality and characteristics of users from verbal and non-verbal social signals, and adapt to them accordingly.

2.1.1 Project

In this new era of ubiquitous intelligent systems becoming more and more seamlessly integrated into our daily life, the future looks daunting for part of the society. "*How should we adapt to this new technology? How can we, as users, learn how to interact with machines?*". Evidences of racially and gender biased artificial intelligence (AI) have also contributed to a skeptical vision of such technologies. While valid, such questions and fears must be tackled from a different perspective, that is, "*how can technology providers train these systems to interact in a more humane way, while accounting for possible society biases?*" The so-called task of humanizing AI deals precisely with these subjects, in order to produce intelligent systems capable of successfully interpreting and reacting to human factors. According to the psychology literature, the way we perceive some social signals, such as eye gaze and facial expressions, is affected by our personality (Ponari et al., 2013), health status (Surguladze et al., 2004) and cultural identities (Riviello and Esposito, 2016). Therefore, **socially-intelligent systems have to be capable of accurately perceiving and inferring the personality and other particularities of different individuals, so as to provide a more effective, empathetic, and natural tailored communication**.

To embody this human likeness into such systems, it is imperative to have a deeper understanding of real human-human interactions first, to computationally model both individual behavior and interpersonal influence. Current literature in computer vision and machine learning for human behavior understanding has mainly focused on research and development of perception, analysis and synthesis methods for individual behavior; however, interpersonal-based tasks such as perception and modelling of the communication flow and the adaptation between communication partners have been largely unexplored from a technology point of view (Vinciarelli et al., 2015). To advance in such areas, the community is in need of **publicly-available annotated datasets of non-acted, spontaneous interactions** among dyads and small groups belonging to different population groups in terms of age, gender, and cultural background. While several acted datasets exist (Busso et al., 2008), natural interactions are preferred, as they cover the richness and complexity of social communications in real life.

As a result, the main goal of Face-to-face Dyadic Interaction Dataset is to **analyze human-human communication, from a multidisciplinary perspective (i.e. sociological, psychological and technological)**, to research and implement new paradigms and technologies of interpersonal behavior understanding by means of a common database of dyadic face-to-face interactions. The purpose is to move beyond automatic individual behavior detection and focus on the development of automatic approaches to study and understand the mechanisms of perception of and adaptation to verbal and non-verbal social signals in dyadic interactions, taking into account individual and dyad characteristics. Our central research question revolves around the **feasibility of developing socially-aware systems able to Face-to-face Dyadic Interaction Dataset the real personality and internal process of an individual by the social signals they convey, as well as understand how such interaction partners perceive and react to those cues directed to them**. To answer our hypothesis, the project will address the following sub-tasks:

1. To design and collect **highly-varied audio-visual, physiology, sociodemographic and personality profiling data from target population, comprising: audio-visual 360º recordings of face-to-face dyadic natural interactions** from third and first-person view cameras and ambient and lapel microphones; heart rate data from wearable monitors; real and apparent personality, temper and current mood profiling through self- and hetero-evaluation questionnaires (filled in by both interaction partners); sociodemographic individual metadata consisting in gender, age, ethnicity, country of origin, country of residence, occupation and education; and dyadic metadata such as relationship among participants (i.e., friends, family, unknown).

2. To manually **annotate the dataset** of audio-visual recordings, ranging from low-level behaviors such as facial expressions, eye gaze, gestures, body poses and utterances, to high-level behaviors such as attention, interest, engagement and overall quality of interaction. Transcription of dialogues would also be included. Further individual and dyadic ground truth would be extracted from the analysis of the profiling data, such as dominance, leadership, agency, communion, and high-order universal phenotypic personality traits (openness to experience, conscientiousness, extraversion, agreeableness, and neuroticism), using standardized questionnaires such as BFI-2 (Soto and John, 2017), HEXACO (Lee et al., 2004), CBQ-short (Putnam and Rothbart, 2006) and EATQ-R (Ellis and Rothbarth, 2001).

3. To explore relationships between ground truth annotations of social signals with behavioral constructs and sociodemographic and profiling data, using modern and flexible statistical multivariate procedures that allow testing complex relations between variables at a time.

4. To research and develop automatic **methods of interpersonal influence understanding and personality traits regression from interaction social signals using novel multimodal deep learning techniques**.

The database is planned to include a sample of 150 Spain-based individuals to better capture social interaction nuances of our society, who would participate in over 200 dyadic interactions. It would consist of several tasks that elicit particular social behaviors, such as joint attention, competition, collaboration, and free conversation, with different levels of cognitive load. Currently, **no annotated dataset of such dimensions and characteristics is publicly available for research purposes**. By releasing the annotated database to the research community, **we encourage data sharing and collaboration among different disciplines, reuse, and repurposing of new research questions, as well as foster international visibility and exposure of Spain-based research**. Data collection and sharing would be guaranteed by ethical committee approval and consent of participants in agreement with GDPR.

In a world that becomes more and more automated, the knowledge and novel techniques that this project would produce would enable **more representative and accurate interaction models**, which in turn would provide us with more empathic agents, tailored to the user needs, social context and characteristics (i.e. sociodemographic group, personality traits, etc.). The project would also serve as a proof of concept for more fair, socially-inclusive, explainable and interpretable automatic models, getting away from the "*black-box*" preconception of AI-based systems, and to automatically detect and **account for possible sources of bias in our society**. The applications are countless, ranging from virtual tutoring and therapy systems to assisting care for the elderly and people with disabilities (e.g. a personalized virtual assistant with compatible features with the elderly), as well as support in job interviews, among others, ultimately **improving our quality of life**.

2.2 Summary

The Face-to-face Dyadic Interaction Dataset contains a total of **194 interaction sessions by 150 different participants**. The recording procedure of each session consists of 5 task done by pairs. This tasks are completely different from each other with different behavior elicitation conditions and cognitive workload.

The duration of each session, on average, is 25 minutes. This means that the entire dataset is made up by 81 hours of interaction audiovisual content. Recruitment and organization of sessions followed a strict criteria. The recording session were programmed with respect to the participants availability, age, gender and relationship among participants, trying to record each participant more than one time and with known and unknown people. The maximum number of sessions per participant is 5 and the minimum allowed age is 4 years old. Fig. 2.1 shows an histogram with the number of sessions done by participant, which indicates that the mean number of sessions is **2.59**.

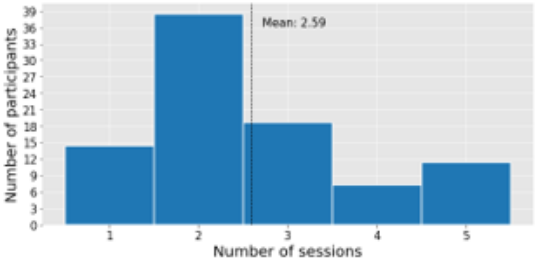

FIGURE 2.1: Histogram of number of sessions done by participant

2.3 Setup

In order to record all the sessions, we use a synchronized setup composed for the following elements:

- 6 HD 720p cameras:
  - 3 AXIS M1124 IP
  - 2 Revotech i712
  - 1 Revotech i706
- 2 HD egocentric cameras Victure AC800
- 2 wrist heart rate monitors FitBit Charge 3
- 2 label microphones Rode Smartlav+
- 1 ambient microphone Olympus ME-33

  The 6 HD cameras are synchronized between them, three by three. We can see a picture of the setting in Figure 2.2. The cameras N1,Z1 and HC are synchronized between them, same for N2,Z2 and GC. With a clock visible for N1 and Z1, we synchronize the rest of them. The wrist heart monitors and egocentric cameras do not appear in the image, they are worn by the subjects: the wrist heart rate monitor is obviously in the wrist, and the egocentric cameras are hanging on the neck.

All cameras except GC are frontal. N1, N2, Z1, Z2 are focusing on of the others subjects, while HC and ZC are general, as we see in Figure 2.3. It also appear the clock in the views Z1 and N1 (two images in the top left), which we use to synchronize these two views and consequently the rest.

2.4 Tasks

The five tasks done by the subjects in the experiments are always the same, but the order of them is changing, except for the talking task which is done always first and the eye gaze check which is done in the last place. It is another variable of the dataset. These tasks realized are the following ones:

- **Free conversation task** (around 5 mins.)

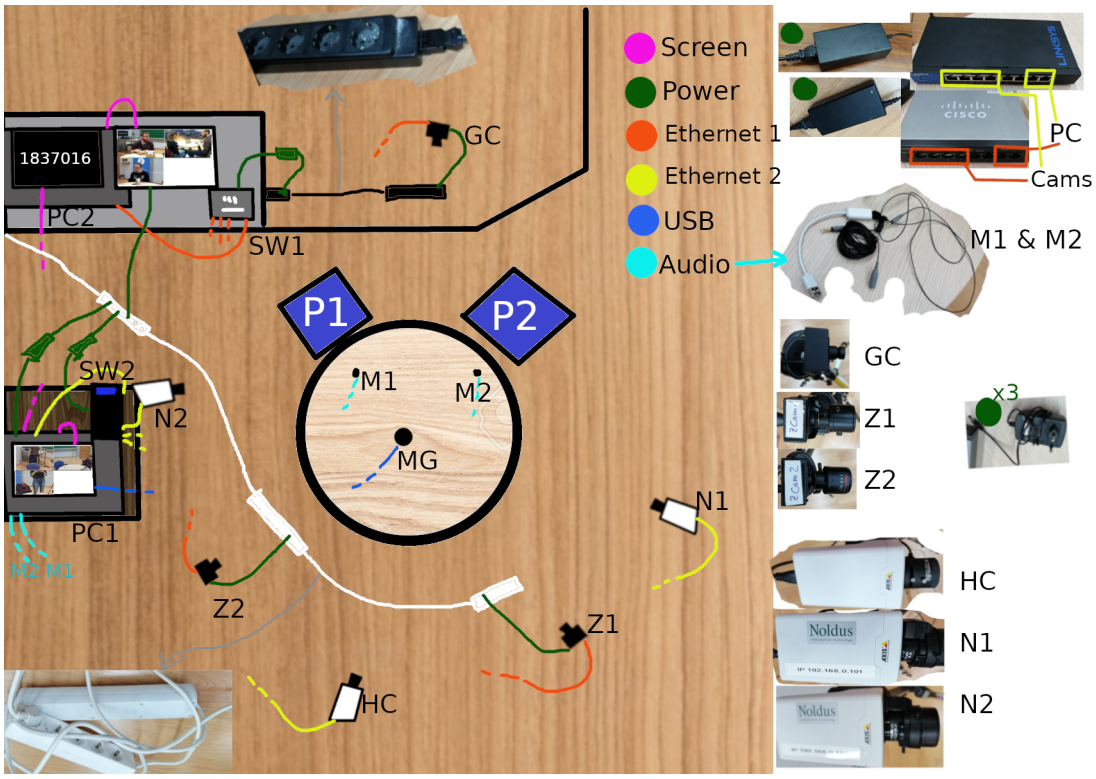

FIGURE 2.2: Picture of the setting of the project. The 6 HD cameras are named N1, N2, Z1, Z2, HC, GC. P1 and P2 are the subjects of the experiment, and M_i are the microphones

FIGURE 2.3: Comparison between the cameras. Frontal 1 correspond to Z1 and Z2 cameras; Frontal 2 to Z1 and Z2. General to HC and GC.

- Talk about any subject (e.g how was your day? hobbies, tv series, etc.), avoiding private information.
- Allows analysis of common conversation constructs, such as turn taking, synchrony, empathy and quality of interaction, among others.

- **'Who am I?' game** (around 7 mins.)
  - Participants use 10 YES or NO questions each to guess the identity of the animal they have on the forehead.
  - 3 difficulty levels (easy - e.g. penguin, medium - e.g. leopard, hard - e.g. albatross).
  - Analyze cognitive processes (e.g. thinking, gaze events).

- **Lego building** (around 5 mins.)
  - Participants build a lego together following a set of instructions.
  - 4 difficulty levels (super easy - for children, easy, medium, hard).
  - Fosters collaboration, cooperation and joint attention.
  - Elicits leader-follower behaviors and subject-object interaction.

- **Ghost blitz game** (around 5 mins.)

  - Participants have to select, among a set of 5 figurines, the figure whose color and shape is not shown in a selected card from a deck of cards.
  - Fosters competitive behavior and interaction with objects.
  - Allows to analyze cognitive processing speed and reflexes.

- **Eye gaze check** (around 1.5 mins.)
  - Participants follow instructions to look *'Elsewhere'*, *'at others face'* or *'at static/moving object'* while moving head and eyes.
  - Elicits different gaze behaviors, such as smooth pursuit.
  - Useful for automatic gaze estimation methods.

2.5 Data and details of participants

Together with the annotations that will be done in the dataset, there is other data which the participants give through some questionnaires. The first one is about personality and temperament. It is self-evaluated for every participant before the first session they do (real data) and hetereo-evaluated after each session between participants (apparent data). This questionnaire differ depending on the age of the participant:

- From 4 to 8 years old: Temperament. **CBQ-short** (Putnam and Rothbarth, 2006 [20]).
- From 9 to 15 years old. Temperament and self-regulation. **EATQ-R** (Ellis and Rothbarth, 2001 [21]).
- For 16+ years old: Personality. **Big-Five BFI-2** (Soto and John, 2017), **Honesty-Humility HEXACO** (Lee et al., 2004).

The other questionnaire to be answered is about current mood: **PEQPN** (Williams et al. 2000). This is self-evaluated before and after session, and hetero-evaluated after session.

Sociodemographic metadata of the participants such as age, gender, ethnicity, country of origin, country of residence, occupation, maximum level of studies and relationship among participants was already know before planning the sessions. The relation between them is done in order to have the maximum variability in the dataset. We see in Figure 2.4 the age of the participants of the study. It ranges from 4 to 84 years old, with mean of 31.47 ± 14.6 and 55% are male. Most part of people is from Young age, since a big amount of them were friends of the researchers of this project.

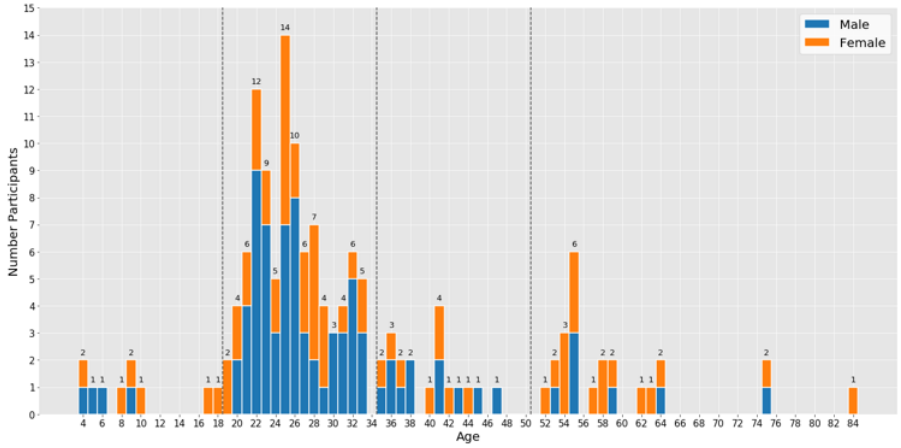

FIGURE 2.4: Histogram of number of participants wrt. age and gender.

In Figure 2.5 it is shown the distribution of the sessions wrt. age, gender and known/unknown people. Again, most of the interactions are done between young people. From the 194 sessions, 44% of them were done between known people (e.g. parent-child, friends, work/study colleagues, couples, etc).

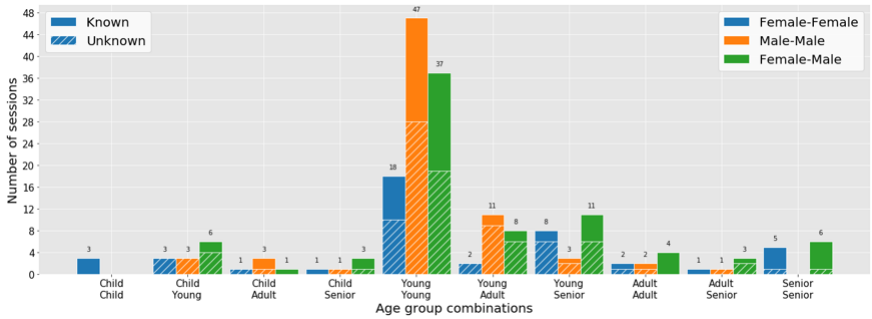

FIGURE 2.5: Distribution of interaction sessions wrt. age, gender and interaction among known/unknown people

The country of origin and sessions done between people from different countries is shown in Figure 2.6. From the 150 subjects, 69% had Spain as country of origin. It was also a considerable number of people from Venezuela (12) and Chile (5). From the 194 sessions, 50% of them were done with both participants from Spain as country of origin.

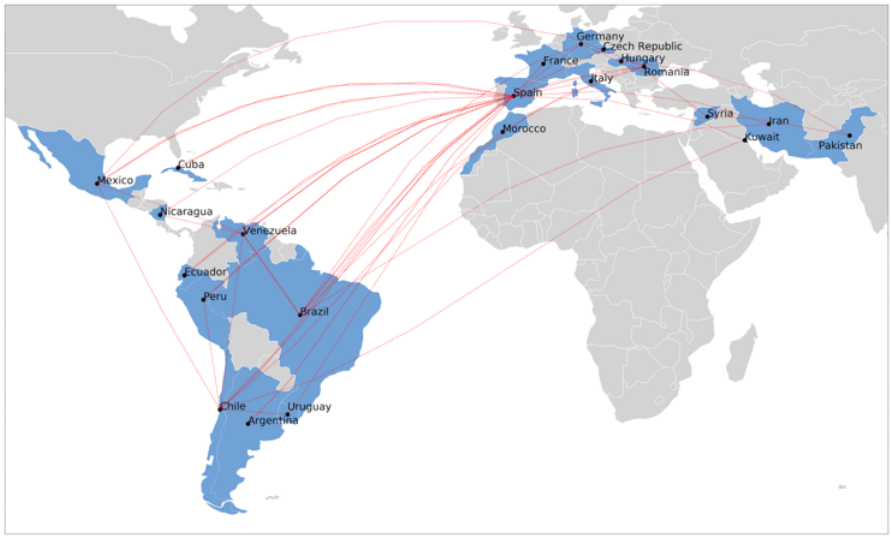

FIGURE 2.6: Country of origin of the participants and sessions done between them.

2.6 Annotations

As for any dataset, the more labels available for it, the more rich it will be. Some labels can be obtained using reliable automated techniques, for example for tasks such as body and hand pose estimation. However, for other labels a ground-truth is required, specially if we aim to make this dataset a reference in dyadic human interaction. In particular, manual human annotation will be required for labelling emotion states and utterances. The protocol for these annotations has been defined and some pilots have been tested.

2.6.1 Emotion Annotation

Emotions will be rated for short video segments (i.e. 3 second intervals), taking into account the whole scene, using all available modalities, as well as the history of the interaction. That is, information of the affective state of a participant can come from speech, language content, gestures, facial expressions, objects of the scene, or even the reaction of the other participant. Annotators will rate three dimensions of emotion, characterized by the Valence-Arousal-Dominance (VAD) model. Furthermore, the annotators will provide a verbal description of what parts of the image are the most important ones to perceive the felt emotion, as well as a verbal description of the emotions they are perceiving.

Videos will be annotated using the Valence-Arousal-Dominance model (VAD, or PAD, from Pleasure) (Russell and Mehrabian, 1977), which consists of a continuous 3-dimensional model of affective space characterized by three dimensions: valence (pleasant-unpleasant), arousal (active-calm), and dominance (in control-submissive). This dimensional model allows to characterize emotions on three dimensions, each of which spans an interval of real-valued numbers indicating the strength and orientation of each dimension. Dimensional approaches as VAD allows us to represent emotions in a more fine-grained way, in contrast to categorical approaches as Ekman's basic emotions (Anger, Disgust, Fear, Happiness, Sadness and Surprise). Ekman's can be mapped to the VAD model as in Figure 2.7. One can observe that

Ekman's categories are unevenly distributed in the VAD space (Buechel and Hahn, 2016).

Each dimension of the VAD model is characterized as follows:

- **Valence**: measures how pleasant or unpleasant one feels about something. It represents the positive or negative dimension of an emotion. For instance, joy and happiness are pleasant emotions, while fear and anger score in the unpleasant side.
- **Arousal**: measures how excited or apathetic one feels about something. It represents the degree or the strength of the emotion, ranging from calm, bored and sleepy to aroused and excited. It may also denote the mental activity, ranging from low engagement to ecstasy). Note that it is not the intensity of the emotion, as grief and depression can be low arousal intense feelings. For instance, while both anger and rage are unpleasant emotions, rage has a higher arousal state. However, boredom, which is also an unpleasant state, has a low arousal value.
- **Dominance**: it measures the extent to which the emotion makes the subject feel in control of the situation, ranging from being in control of one's emotions (in control, empowered, confident) to dominated by them (submissive, oppressed). Note that having a high level of dominance does not mean one is the dominant person of the interaction, as it is not an interpersonal attribute, but an individual one influenced by the situation and context. The authors of the model claimed that this is a necessary dimension to describe emotion as only dominance makes it possible to distinguish "angry" from "anxious," "alert" from "surprised," "relaxed" from "protected," and "disdainful" from "impotent". The first adjective of these pairs denotes a dominant emotion, while the latter denotes a feeling of being controlled by one's emotions.

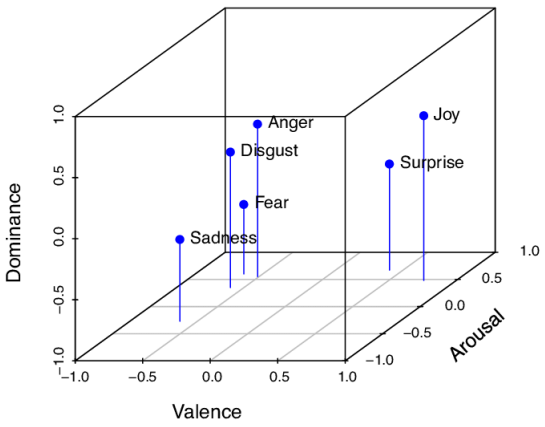

FIGURE 2.7: Mapping from VAD to Ekman's emotions [1]

To assess the three dimensions of valence, arousal, and dominance, the Self-Assessment Manikin (SAM), an affective rating system devised by Lang (1980) will be used. In this system, a graphic figure depicting values along each of the 3 dimensions on a continuously varying scale is used to indicate emotional reactions. As can be seen in Figure 2.8, SAM ranges from a smiling, happy figure to a frowning, unhappy figure when representing the valence dimension. For the arousal dimension,

SAM ranges from an excited, wide-eyed figure to a relaxed, sleepy figure. For the dominance dimension, SAM ranges from a large figure (in control) to a small figure (dominated). Annotators can select any of the 9 figures comprising each scale, which results in a 9-point rating scale for each dimension. Ratings are scored such that 9 represents a high rating on each dimension (i.e., high pleasure, high arousal, high dominance), and 1 represents a low rating on each dimension (i.e., low pleasure, low arousal, low dominance).

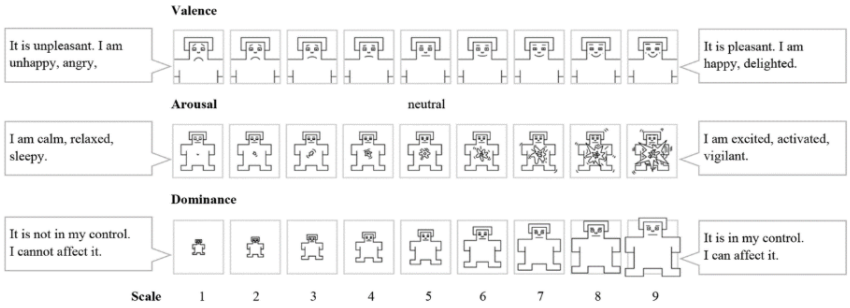

FIGURE 2.8: Self-Assessment Manikin visualization. [1]

2.6.2 Utterances

There is no common consensus regarding what an "utterance" is, and each research community uses it in a different way. There are many proposals, like the one from Traum and Heeman, 1997. In this work, we will follow instructions and conventions from [15] to annotate utterances.

Determining where the boundaries fall between utterances is an important and tricky part when annotating a conversation. Lots of segments of speech qualify as utterances: a word, a short phrase, or a complex sentence with many embedded clauses. Unfortunately, people do not speak with periods, commas, and question marks to let you know when one utterance ends and another one begins.

Definition 1. *An **utterance** can be a word, a phrase, or an entire sentence. It is the smallest unit of speech.*

The following are all potential examples of utterances: (*ok?*), (*uhuhh*), (*the pink one*), (*yeah, well, I thought she was going to, but she never did.*), (*bugs lives outside, honey.*). For this work, we will also label non-verbal vocalizations (e.g., laughing, sobbing, crying, mouth clicking, sighing) as utterances, as they have a communicative meaning. However, other non-communicative sounds, like sneezes and coughs, are not considered utterances. In the following lines, some rules that have been followed through the annotations are presented:

- **Pauses of 2 seconds or less**: If the speaker hesitates to find a word, it is treated as one utterance, unless the speaker pauses for more than two seconds. (e.g., *put it in the – toy box*).
- **Self-corrected speech**: If the speaker interrupts herself to correct herself, it is treated as one utterance. (e.g., *don't do that, Nathan – Jake!*).
- **Self-interrupted speech**: If the speaker interrupts herself to express an entirely different thought, it is treated as two separate utterances. (e.g., *why don't you put it – don't do that*).

- **False starts**: If the speaker has a false start in her speech, it is treated as one utterance. (e.g., *will you – will you go to your room?*).
- **Speaker stumbling over words**: If the speaker stumbles over his or her words while trying to formulate an utterance, still count this as a false start and transcribe the attempts on a single utterance line. To count as a false start, the words must be spoken very quickly or be otherwise clearly an attempt at verbalizing a single thought. (e.g., *no – don't – oh no!*).
- **Utterances never span more than one conversational turn**: When people are taking turns talking, having a conversation, a single utterance will never span over more than one person's turn.
- **Utterances are never more than one complete sentence**: Complete sentences include things like "the man is walking" and "I'm sleepy". If sentences are joined by conjunction words (like "and", "or", "but", "because", "after", "for", "so", "if", "when", etc.), then they are transcribed them together on a single utterance line, but if there are no conjunction words, the sentences are separated utterances.
- **Using semantic and syntactic cohesion to determine utterance boundary**: In general, when phrases are related semantically and grammatically, they should are transcribed together as one utterance if there is less than a two second pause between them. For instance, the phrases "I'll go first" and "and then you can have a turn" are related semantically because after "I'll go first" is the time when you can have a turn, and the conjunction words "and then" connect the phrases to each other grammatically.
- **Using intonational contours to help determine utterance boundary**: If two phrases are part of two totally separate intonational contours, they are transcibed as two separate utterances instead of a single utterance. Intonational contour refers to the pattern the pitch of your voice makes when you utter questions, propositions, and commands. An example is shown in figure 2.9.

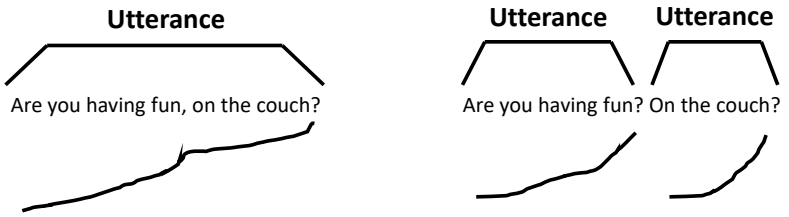

FIGURE 2.9: Examples of intonational contours.

- **Repeated words and phrases**: If a speaker keeps repeating a word or short phrase over and over again (e.g. "no no no, no no, no no no....no!" or "I won't I won't I won't I won't!", or is counting, saying the alphabet, or reciting a list, the intonation and 2 second pause rules are used to find the boundary.

Chapter 3

Multitask learning

Traditionally, in Machine Learning we aim to solve one particular task. For that, we create and train models to perform this specific task. Afterwards, we perform techniques such as transfer learning, hyperparameter tuning until the metrics used to monitorize these models no longer increase. Other times, we may have multiple tasks at hand, but in that scenario we would act similarly by creating one model for each individual task. Generally, we can obtain decent or even very high performance in this way, by only focusing in one single task. However, we might be ignoring relevant information coming from training signals from other tasks, which may improve our metrics . Thus, by sharing features or parameters between related tasks, we can help our models generalize better [7]. This is known as Multitask Learning (MTL).

This approach has been used successfully across all domains of Machine Learning and Deep Learning: natural language processing, audio signal processing, computer vision etc. Multitask learning might be defined in multiple ways, but generally, whenever multiple loss functions are optimized MTL takes place. Zhang and Yang [11] propose a more formal definition for MTL:

Definition 2. *Given m learning tasks $\{T_i\}_{i=1}^m$ where all the tasks or a subset of them are related, multitask learning aims to help improve the learning of a model for T_i by using the knowledge contained in all or some of the m tasks.*

And Caruana [2] summarizes the goal of Multitask learning: "improve generalization by leveraging the domain-specific information contained in the training signals of related tasks".

Multitask learning should not be confused with transfer learning, although their settings are similar, in fact they have different natures. Generally, in MTL all tasks are given the same importance, meaning that we aim to improve performance in all of them. On the other hand, in transfer learning there is one target task which is more important and which is improved using knowledge contained in the so-called source tasks. However, sometimes in MTL some tasks are treated as secondary or auxiliary tasks to increase performance in a primary one. In that scenario, MTL resembles more transfer learning, but even in that case multiple loss functions are being optimized [11].

3.1 MTL motivations and reasons behind

Motivation for MTL can be expressed from both a biological and a machine learning perspective [7]. There are obvious reasons to motivate this approach from the human biology perspective: it is precisely how we learn. We are constantly using information and knowledge from different tasks whenever we learn a new one. This is

true for almost every sort of task: recognizing objects, learning new math operations, playing new sports, etc. Sometimes, we learn basic tasks to master more complex ones.

From a machine learning perspective, MTL can be seen as a form of inductive transfer. In particular, it introduces an inductive bias which is brought by the auxiliary tasks. In machine learning, an inductive bias of a learning algorithm is considered to be as the set of assumptions that the learner uses to predict new outputs. This may seem intuitive, but digging deeper we can find the following reasons of why MTL works. Let's assume we have two tasks A and B which share some common hidden layer representation F [2].

- **Implicit data augmentation**:

  Implicitly, multitask learning increases the number of samples which we use for training. All tasks are somewhat noisy, therefore our goal is to learn a good representation that ignores this data-dependent noise and generalizes well. Different tasks have different noise pattern. Consequently, learning jointly task A and task B enables the model to obtain a more general representation F that suits both tasks through averaging noise patterns.

- **Attention focusing**:

  For very noisy tasks or in situations where data is not abundant and complex, models might have trouble identifying which are relevant or irrelevant features. Solving new tasks can help these models focus their attention on the relevant features, since these new tasks might provide new evidence for the relevance or irrelevance of the features.

- **Eavesdropping**:

  There are features which are easier to learn for a certain task B, while they are difficult to learn for another task A. There can be many reasons behind this: task A might interact with these features in a more complex way or other features are blocking the model to learn these features. Considering MTL, eavesdropping can take place: learning those features through an auxiliary task B. One easy way to do this is through hints: learning a model that directly predicts the most important features.

- **Representation bias**:

  Multitask learning biases the preferences of the model by choosing those that other tasks also prefer. Furthermore, MTL tasks also prefer not to use representations that other tasks prefer not to use. This improves generalization in terms of learning new tasks in the future, since we are in a hypothesis space that performs well for a large number of tasks. Therefore, it will keep performing well for novel tasks as long as they are somehow related.

- **Regularization**:

  MTL can also be seen as a regularizer. As we have seen, multitask learning reduces the risk of overfitting and can be used as a tool to combat it. However, this might represent a special case of regularization as it does not necessarily increase the training error to reduce test error as other techniques do.

3.1.1 Auxiliary tasks

Consequently, taking into account the aforementioned reasons, multitask learning should not only be considered in scenarios where we face and aim to solve well multiple tasks, but also for single-task scenarios. In those situations, where we only care about one particular performance, there are multiple ways to find auxiliary tasks which may help increase performance profiting from the multitask learning benefits [7].

- **Related task:**

  Using related tasks is a typical choice in MTL. According to Caruana, two tasks can be considered as related if similar features can be used for solving them. Some examples of using multiple tasks can be: using facial attributes as auxiliary tasks for predicting human emotion, perform classification as well as object detection in an image, in audio processing jointly predict language and translation etc.

- **Adversarial task:**

  Using related tasks may have a problem: the unavailability of labeled data. But sometimes we have access to data which represents the opposite of what we want to achieve. Using a gradient reversal layer, we can leverage this data using an adversarial loss. This procedure has found success in domain adaptation problems. In that case, the adversarial task is predicting the domain of the input and with the gradient reversal layer the adversarial loss is maximized. This is helpful because it forces the model to learn representations that are not able to distinguish between domains.

- **Hints:**

  As we commented in the eavesdropping mechanism, using hints can be beneficial for learning features which are difficult to obtain using only the original task. One example can be found in Natural Language Processing (NLP), where for sentiment analysis models auxiliary tasks such as predicting which input words are positive or negative are used.

- **Focusing attention:**

  In a similar mechanism as using hints, we can use auxiliary tasks to focus attention on important features that could be ignored. For example, in computer vision and more specifically in facial recognition, one could use the location of facial landmarks as an auxiliary task which could certainly be useful for distinguishing people.

- **Predicting inputs:**

  Caruana [2] showed that there are situations where some inputs are impractical to use as features, but surprisingly these inputs work better as outputs and can still guide the learning of the task.

- **Using the future to predict the present:**

  Sometimes valuable features only become available after predictions must be made. These features cannot be used as inputs as they will not be available during run time. However, we can collect historical data and use them as

auxiliary tasks. Predictions for these tasks will be ignored during runtime but their functionality is to serve as extra knowledge to the model during training.

One example could be found in medical risk prediction, for example in assessing the risk of death of a patient once pneumonia has been diagnosed to decide whether the patient has to be hospitalized or not. There are useful tests which can measure this risk but will only become available after the patient has been hospitalized. Therefore, this test cannot be used as input but it can be used as an auxiliary task using historical data of hospitalized patients.

All these auxiliary tasks share one common aspect: their goal is to help, in a more or less explicit manner, the model to learn more useful representations for the main task. Furthermore, they are somehow related to the original task, which allows the representations to be beneficial. This can be considered as an example of representation learning.

3.2 Multitask learning architectures

Before describing some of the current state-of-the-art architectures for multitask learning, let's discuss first some aspects that need to be considered to create a multitask model. In MTL, there are three big questions to be addressed: when to share, what to share and how to share [11].

The first question is related to choosing multi-task or single-task architectures when there are multiple tasks to be solved. In other words, which tasks should be shared, and which ones should be kept apart. There are few learning approaches for that as currently such decisions are taken by human experts. For some architectures this can have a big impact as sharing between unrelated tasks could become in what is known as 'negative transfer'. Another solution, though very computational expensive, is to formulate this decision as a model selection problem using techniques such as cross validation. Finally, another solution is to use architectures which can degenerate into single-task counterparts. In this scenario, the model decides with its own mechanisms when to share between all tasks considered.

The second question is what to share. The answer to this question will determine the form in which knowledge sharing between tasks could occur. There are mainly three ways to do it: feature-based, instance-based and parameter-based. The first approach tries to learn common features that may be useful for all considered tasks as a way to share knowledge. Instance-based approaches aim at identifying useful data instances in a task for other task and shares the knowledge through them. Finally, parameter-based approaches use model parameters (for example, weights in a neural network) in a task to help learn parameters for other tasks, for example through regularization. The state of the art in MTL focuses mainly in feature-based and parameter-based approaches. Instance-based models are rare and there are few studies on them. However, one example is to weight training data based on density ratios between probabilities that each instance and its label belong to both its own task and a mixture of all tasks considered. This method is known as multitask distribution matching method.

Once we have answered the first two questions, 'how to share' determines specific ways to share knowledge between tasks. For example, in feature-based MTL one primary approach is to learn common features for all tasks. For example, share some hidden layers of an artificial neural network or share the convolutional blocks

of a Convolutional Neural Network (CNN) and then create branches of fully connected layers for each tasks. In parameter-based MTL there are different ways of how to share knowledge. One approach is the low-rank approach, where parameters for each task are assumed to belong in a lower dimensional subspace. In other words, the parameter matrix of these tasks has low rank. Other approaches are the task clustering approach, task relation learning and the decomposition approach.

The following sections will describe some specific methods for MTL, differentiating between feature-based and parameter-based approaches.

3.2.1 Feature-based MTL

As stated previously, it is reasonable to assume that for related tasks we can obtain useful common representations. This common representations can bring improvement on performance, as they might have high expressive power for these multiple tasks. In feature-based MTL there are also two subcategories: feature transformation approach, where the common representation is a linear or non-linear transformation of the original representation, and feature selection approach, where the learned representation are a subset of the original features.

The most iconic example for this first category is known as hard parameter sharing. It is also the most common approach in multitask neural networks. Basically, it consists in sharing all or some hidden layers of a neural network while keeping task specific layers. It is a feature learning approach as it forces the network to learn a common and useful representation in the hidden layers for all tasks. However, it is accomplished by sharing the same parameters, in other words, the weights of the hidden layers, and hence its name. Hard parameter sharing reduces theoretically the risk of overfitting as the number of tasks increase [7]. It is very intuitive, the more task a model has to learn the more general the representation learned must be. Therefore, as the model is trying to perform well in all tasks, it cannot learn specific and noisy features that would overfit the single task.

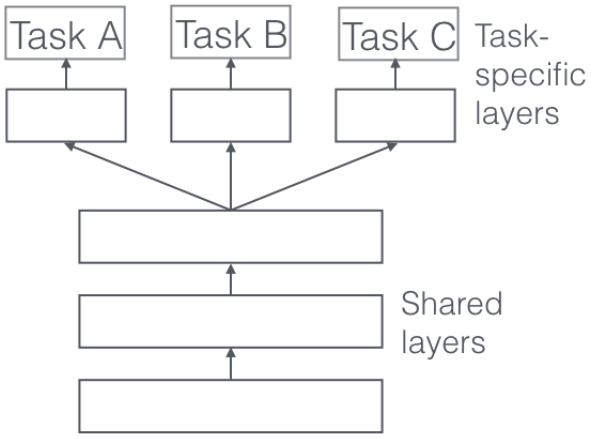

FIGURE 3.1: Hard parameter sharing network for three tasks. [7]

Hard parameter sharing is the standard architecture for MTL and specifically for feature-based MTL. However, there has been some research in discovering better mechanisms for this feature sharing approach. One example are Deep Relationship Networks, employed in computer vision. In computer vision, a common approach for hard parameter sharing is to share all convolutional layers while keeping

task-specific fully-connected layers. Deep Relationship Networks improves this approach by placing matrix priors on the fully connected layers as in a Bayesian model. This approach still relies on a pre-defined structure for sharing.

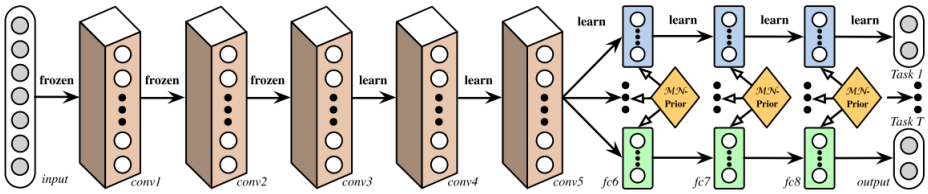

FIGURE 3.2: An example of a Deep Relationship Network. [7]

Other approaches do not rely on fixed structures but instead they learn dynamically the structure during training. This is the case for Fully-Adaptive Feature Sharing. It starts with a hard parameter sharing model and during training it creates new branches greedily promoting grouping of similar tasks. However, this method may not discover an optimal model and also it can create individual branches which does not let the model learn more complex common representations.

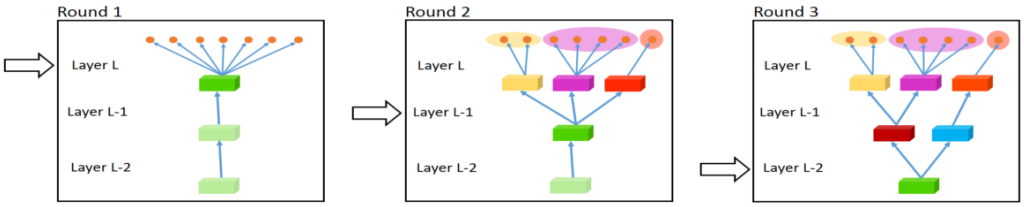

FIGURE 3.3: Evolution of a fully-adaptive feature sharing network. [7]

On the other hand, other approaches start with separate architectures, where for each task the same model is replicated. For example, Cross-stitch Networks [6]. These networks introduce the so-called cross-stitch units that determine when to share features and which features should be shared by learning linear combinations of them. One particular effect of these units, is that they can avoid negative transfer between unrelated tasks. A similar approach, known as Proportional Addition is to learn directly linear combinations of the feature maps. This greatly reduces the amount of parameters but also the complexity of the shared representations.

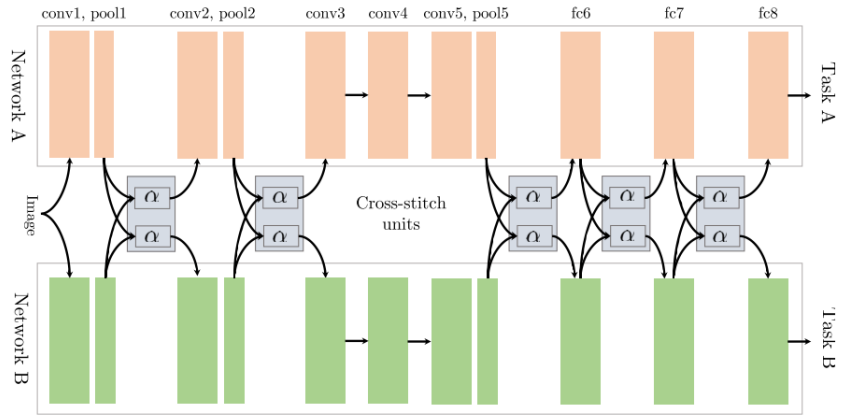

FIGURE 3.4: Cross stitch network. Cross stitch units are placed after pooling and fully connected layers. [6]

Finally, Ruder [7] proposed a model that generalizes some of the above approaches in a single framework for NLP, so that it has all benefits of the above approaches. This multitask learning networks are known as Sluice Networks. Basically, it allows the model to learn which layers and subspaces should be shared and also it learns at which layers of the network the best representations of the input sequences were learned.

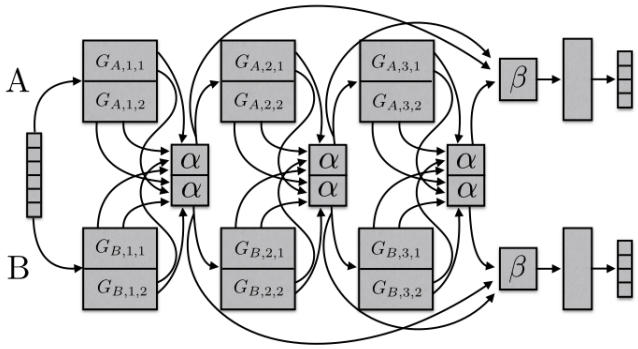

FIGURE 3.5: An example of an Sluice network. [7]

One approach for doing feature selection is to use regularization in the form of norm constraints on the parameters W. The goal of performing regularization is to enforce W to be row-sparse, which can be interpreted as a selection of the important features. Mathematically it can be expressed as:

$$\min_{W,b} L(W,b) + \lambda ||W||_{p,q}$$

where p and q can be any norm and $||W||_{p,q} = ||(||w_1||_p, ..., ||w_d||_p)||_q$. This feature selection approach can be viewed as a special case of the feature transformation approach with 0's in the transformation matrix which discard irrelevant features. Therefore, it is immediate to observe that feature transformation approaches will fit better the training data as they have more capacity which, in turn, it will yield to a better generalization if the model does not overfit . However, feature selection approaches have better interpretation since they are directly selecting the relevant

features. This trade-off should be considered when deciding between these two subcategories.

3.2.2 Parameter-based MTL

As we have seen before, the other big family of MTL methods are the parameter based. The most iconic methods are the low-rank approaches. The motivation for these methods is found in the relatedness of the multiple tasks, which may imply the low-rank of the parameter matrix W. For example, some methods assume that the model parameters of the multiple tasks belong in a low-rank subspace. More specifically, Ando and Zhang [11] assume the following form for the model parameters:

$$w^i = u^i + \Theta^T v^i,$$

where Θ is the low-rank subspace. Then, the parameter matrix can be written as $W = U + \Theta^T V$. Then, they propose this objective function:

$$min_{U,V,\Theta,b} L(U + \Theta^T V, b) + \lambda ||U||_F^2$$

$$\text{subject to } \Theta^T \Theta = \mathbf{I}.$$

The purpose for the orthonormal constraint on Θ is for the subspace to be non-redundant. There are some generalizations for this method, using more complex or transformations of this objective function. Another low-rank approach is to use a different norm as a regularization on $\mathbf{W}$. For example, the trace norm of a matrix $\mathbf{W} \in \mathbf{R}^{m \times d}$ is defined as:

$$||W||_{S(1)} = \sum_{i=1}^{min(m,d)} \mu_i(\mathbf{W}),$$

where μ is the ith singular value of $\mathbf{W}$. It is known that using the trace norm as a regularizer enforces a matrix to be low-rank, therefore it is a suitable norm for parameter based MTL. The objective function is then:

$$\min_{W,b} L(W, b) + \lambda ||W||_{S(1)}.$$

These approaches can be extended to deep learning MTL models. One example and the most iconic is known as soft parameter sharing [7]. In this setting, each task has its own model, similar to the split architectures described for the cross-stitch networks. Then, the distance between model parameters in each layer is regularized using norms such as the L2 norm or the trace norm. This method is known as soft parameter sharing since it enforces the parameters to be similar and was inspired by the previous low-rank approaches for traditional machine learning algorithms.

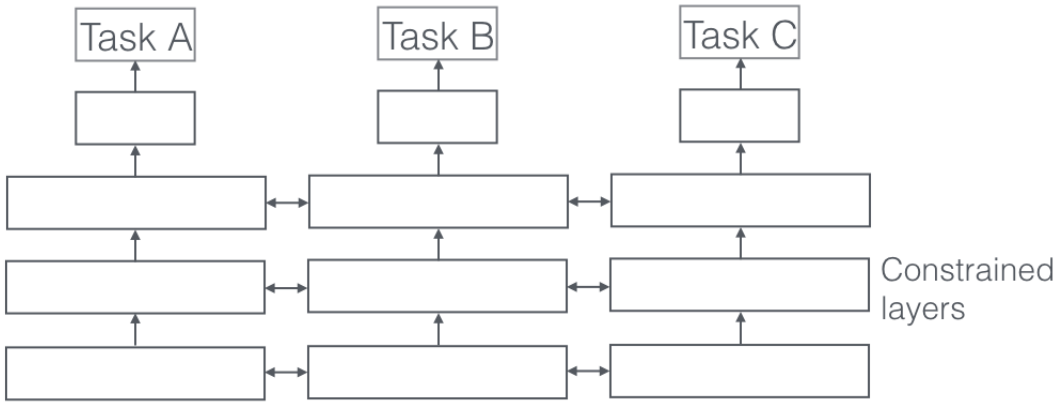

FIGURE 3.6: Soft parameter sharing network for three tasks. [7]

The other three approaches (task clustering, task relationship and decomposition) are mostly used for non-neural models, therefore only a brief description of the nature of these methods will be provided. For instance, task clustering approaches, as their name suggest, are based in clustering algorithms. Basically, they assume that the multiple tasks considered form clusters of similar tasks. Once the clusters of similar tasks have been found, these methods will exploit this information and use it for training the learning algorithm. On the other hand, task relationship approaches try to model the task relatedness by computing quantities such as task similarity, task correlation etc. Afterwards, for example, using this information on the relatedness of the tasks personalized regularizers can be designed so that the more similar two tasks are, the closer their parameters will be. Finally, decomposition approaches assume that the parameter matrix can be decomposed into two or more matrices: $\mathbf{W} = \sum_{k=1}^{h} \mathbf{W}_k$. Then, the loss function of these methods takes the form of:

$$\min_{W_k,b} \quad L(W,b) + \sum_{k=1}^{h} g_k(W_k).$$

In other words, the regularization is decomposed into each component matrix. Using different types of regularization helps catch in a more intelligent manner the different nature of the multiple tasks. Also, it can help avoid negative transfer from outlier tasks.

3.2.3 Task priorization

Another crucial aspect in MTL is the loss function and its optimization. In a multitask learning setup, the loss function of the model is composed of several loss functions, one for each task to solve, and normally, the global loss function is the sum of the individual losses. A natural question to ask ourselves is if we should prioritize some tasks over others during training and if so, under which criteria. For instance, a student in college has many subjects but he does not allocate the same resources into each one. Usually he spends more effort on complex subjects prioritizing these ones.

In MTL, the prioritization may not necessarily follow the same logic [3]. One line of research in task prioritization, known as curriculum learning, learns the tasks progressively, starting with easy and simple tasks first and thus presenting the tasks to the model in an increasing difficulty. However, curriculum learning assumes that

all tasks have the same underlying distribution, which may not be true for tasks that differ very much in nature (pose estimation vs classification). Some methods that can be used for task prioritization are:

- **Task weighting:**

  A task weight is commonly defined as the scalar coefficient of the individual loss. Then, the global loss function for N tasks takes the form: $L(W, b) = \sum_{i=1}^{N} \lambda_i L_i(W, b)$ where λ_i is the task weight for task i. Tasks weights are usually selected through extensive hyperparameter tuning and are often static, which can make this solution inefficient in terms of computational resources. To solve that, recent methods try to dynamically adjust these weights.

- **Self-Paced learning:**

  Compared to static task weighting, self-paced learning is an automated approach for task prioritization where the weights are decided by taking into account the model's abilities. One method is for example selecting automatically these task specific weights via a regularizer on all task weights, according to some predefined criteria or regularization.

- **Task hierarchy:**

  Creating task hierarchies can be an effective way of exploiting tasks relatedness. In these hierarchical multitask models, the more complex tasks are predicted at successively deeper layers of the model. This has been successfully implemented in NLP tasks, where easier tasks as Part of Speech (POS) tagging are implemented in lower layers of the network compared to more complex task such as machine translation, which is implemented later in the model. It differs from task weighting methods since it defines the task prioritization in the structure of the model.

- **Dynamic task priorization based on difficulty:**

  Guo et al. [3] propose a method similar to the student example: let the model focus on the tasks with which it is struggling instead of starting with new ones. This approach can be used on tasks without the same underlying distribution (as the pose estimation and classification example). It is a method which uses task weighting, but it updates the weights dynamically according to the difficulty of the tasks. They use key performance metrics κ, as for example accuracy, instead of losses values to assign automatically new task weights. Then, they define difficulty as $D \propto \kappa^{-1}$. Afterwards, combining this performance metric and a focusing parameter for each task denoted as λ_i, initialized properly at the beginning of the training process, we obtain the task weight at each epoch or time step.

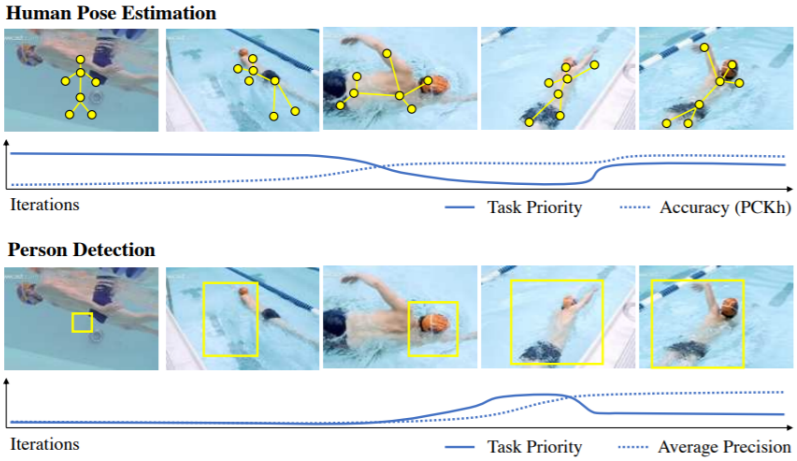

FIGURE 3.7: Example of task priorization.

In figure 3.7, we can see that once the model has reached an acceptable level on pose estimation, it starts lowering the priority on it and increasing the priority on person detection, where the model is not performing well. Then, we can see that with this dynamic adjustment the average precision on the person detection task increases and therefore the priority decreases until it stabilizes and the model stops learning.

3.2.4 Applications of MTL in Computer Vision

Multitask learning has been successfully applied to many machine learning domains such as bioinformatics, audio signal processing, natural language processing, web applications and computer vision to name a few. In this thesis, we are obviously interested in computer vision applications, as it is the domain of our dataset, and therefore, we will briefly review some examples of tasks which have been successfully solved using MTL approaches.

One example of a successful application, presented by Zhang et al. [12], uses auxiliary tasks for improving robustness in detecting facial landmarks. In particular, they use different facial attributes tasks such as smiling, gender, glasses as well as pose estimation inference. Additionally, they implement task-wise early stopping to facilitate learning convergence. Experiments show that multitask learning helped improve current state of the art models for facial landmark detection.

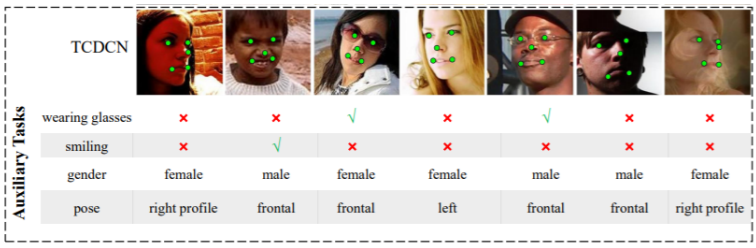

FIGURE 3.8: Tasks considered in [12] MTL setting.

Another example proposed by Luvizon et al. [5], solves action recognition and human pose estimation tasks in a single network. In this work, they estimate jointly

2D and 3D poses from images and human action recognition from video sequences. With a single architecture they can solve both problems in an efficient way and still achieving state-of-the-art results. Additionally, the proposed architecture can be trained using data from different categories simultaneously in a seamlessly way. Furthermore, they reported results on four different datasets (MPII, Human3.6M, Penn Action and NTU) which demonstrated the effectiveness of their method on the targeted tasks.

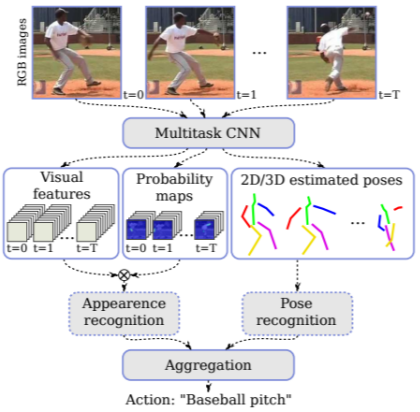

FIGURE 3.9: MTL Architecture for solving action recognition and pose estimation tasks. [5]

Chapter 4

Methods selected

In this thesis, our goal was to explore multitask learning architectures and to perform an exhaustive analysis of these models on our new dataset, by learning multiple facial attributes jointly. One major point was in selecting some multitask learning architectures of the ones described in the previous chapter. To perform an extensive analysis only a subset of them were implemented, those which had demonstrated their accuracy and are currently state of the art in MTL. In this chapter, we will describe in more detail the methods implemented.

4.1 Methods selected

In Chapter 3, we reviewed the current state of the art in multitask learning. We saw the two main families of architectures: feature and parameter based approaches. Since in this thesis we are dealing with computer vision tasks, were inputs basically consists of image data, it is necessary that the considered architectures have been somehow tested and specifically designed for Convolutional Neural Networks, the current state-of-the art of deep learning in image data processing.

Therefore, we discarded parameter based approaches, as they are more thought for non-neural models, and focused on feature-based approaches only. Considering the nature of the models described in the featured based section we thought it would be interesting to select the classical hard parameter sharing approach as the baseline MTL architecture and one of the advanced MTL methods described. This second selected approach was finally Cross-stitch Networks for many reasons, but mainly since they are somehow the natural evolution of hard parameter sharing as we will see in the following sections.

For both models we used the same baseline architecture which does not use any MTL knowledge. It is a standard VGG-style CNN, which consists of 5 convolutional blocks of a convolutional layer of an increasing number of filters plus a max pooling layer and 3 hidden fully connected layers of 256, 128 and 64 neurons plus an output layer with one neuron and no activation for age (regression task), another output layer with one neuron with a sigmoid activation for gender (binary classification) and finally an output layer of 5 neurons with a softmax activation for the ethnicity task (multiclass classification).

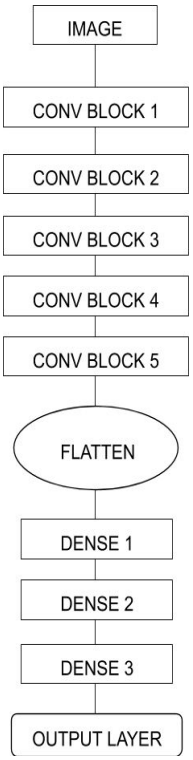

FIGURE 4.1: Baseline architecture. The number of filters for the convolutional layers are 16, 16, 32, 32 and 64 respectively.

Furthermore, the hard parameter network models and the Cross-stitch networks we tested were based in that baseline architecture. In particular, hard parameter sharing models use the same number of convolutional blocks and fully connected layers with the same value for the different hyperparameters (neurons, filters etc.). For Cross-stitch networks this baseline model is precisely the architecture used in each split model, with cross-stitch units placed after the pooling layers. Therefore, we can successfully analyse if the use of MTL techniques increases performance, as all models have a similar level of complexity.

4.1.1 Hard Parameter Sharing

As described in Chapter 3, hard parameter sharing networks are the most iconic example of a feature based MTL models. It is a suitable architecture for solving related tasks which share the same inputs, as it is the case for the UTKFace dataset [23], where the input for the three defined tasks (age, gender and ethnicity) is the same image. Hard Parameter Sharing was first introduced in [2].

Hard parameter sharing forces the network to learn useful representations in the hidden layers for all tasks, by sharing the paramaters in the hidden layers for all tasks. In other words, by using the same hidden layers. The standard hard parameter sharing network contains some shared layers in the beggining and middle of the network while keeping some task-specific layers in the output layers as shown in Figure 4.1. In particular, Convolutional Neural Networks typically share the convolutional layers while using the fully connected layers as the task specific ones.

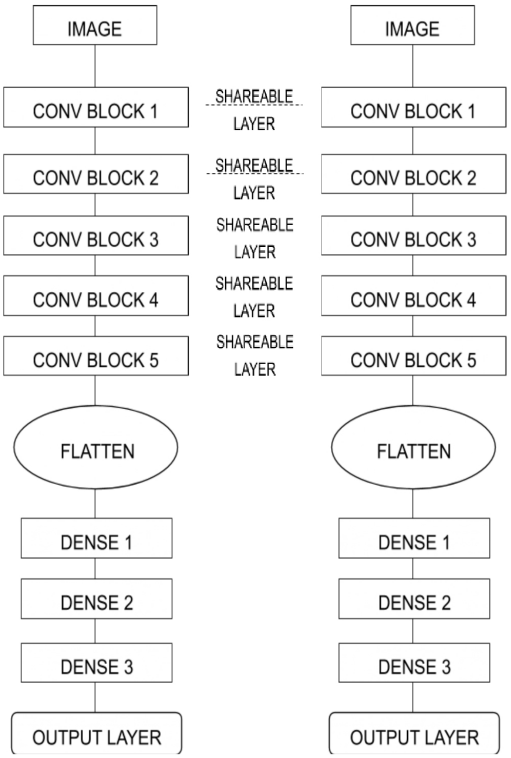

FIGURE 4.2: Hard Parameter Sharing architecture for two tasks. If a layer is shared then there is only one convolutional block.

Theoretically it can reduce the risk of overfitting since the learned representations are more general since they must be useful for all the considered tasks. Additionally, Baxter showed that the risk of overfitting is of an order N (where N is the number of tasks) smaller in the shared layers than in the task specific layers.

However, there are two main issues that must be addressed: the total number of shared layers and the global loss function. The number of shared layers is a hyperparameter, so we tested multiple hard parameter sharing models with different number of sharings. By doing this, we can also analyze how many shared layers is optimal and why. For example, learning jointly only the first convolutional layer might lead to zero significant improvement compared to the baseline model, since CNNs tend to learn the same low-level features in the first layers, independently of the task. However, sharing many layers could cause problems as well, if for example there are tasks which are not as related as they seem. In fact, hard parameter sharing is very sensitive to unrelated tasks and it can break down in those scenarios.

The global loss function is also a key aspect in hard parameter sharing models. Since all losses finally backpropagate in the same shared parameters the scales of the losses should be similar. If that condition was not satisfied, it could be that the network would focus more on some tasks than others, those with higher values in the global loss function. Scaling losses to similar terms is achieved through task weighting, as discussed in Chapter 3.

4.1.2 Cross-stitch Networks

Cross-stitch networks were introduced in 2016 by Misra et al. [6] as a way of addressing automatically some of the issues that we have seen in hard parameter sharing. More specifically, in hard parameter sharing we do not know which is the optimal number of shared layers we should consider, and therefore current approaches simply find this number by brute force. Cross-stitch networks assume that a baseline architecture is replicated for each task separately. Then, they propose cross-stitch units, which try to find the best shared representations by learning linear combinations of the features learned in the different layers of the multiple models. Furthermore, this cross-stitch units can be placed into a CNN, providing an end-to-end learning framework.

In more mathematical detail, let's assume we have two learning tasks A and B on the same input image. Let's also assume we have two separate networks for each task. The cross-stitch unit then combines these two networks in a way such that the tasks supervise how much sharing is needed. At each layer, given the two activation maps F_A and F_B we can learn linear combinations $\tilde{F}_A$ and $\tilde{F}_B$ of both the activations and use these combinations as input to the next layers. Specifically, at each location (i, j) of the activation map we perform:

$$\begin{bmatrix} \tilde{F}_A^{ij} \\ \tilde{F}_B^{ij} \end{bmatrix} = \begin{bmatrix} \alpha_{AA} & \alpha_{AB} \\ \alpha_{BA} & \alpha_{BB} \end{bmatrix} \begin{bmatrix} F_A^{ij} \\ F_B^{ij} \end{bmatrix}$$

where α are the parameters to learn. An immediate observation is that the network can make any layer task-specific by setting α_{AB} and α_{BA} to zero or creating more shared representations by setting large values to these parameters. Partial derivatives can be easily found in terms of F and $\tilde{F}$ since this is a linear combination:

$$\frac{\partial L}{\partial \alpha_{AA}} = \frac{\partial L}{\partial \tilde{F}_A^{ij}} F_A^{ij}, \quad \frac{\partial L}{\partial \alpha_{AB}} = \frac{\partial L}{\partial \tilde{F}_B^{ij}} F_A^{ij}$$

$$\frac{\partial L}{\partial \alpha_{BA}} = \frac{\partial L}{\partial \tilde{F}_A^{ij}} F_B^{ij}, \quad \frac{\partial L}{\partial \alpha_{BB}} = \frac{\partial L}{\partial \tilde{F}_B^{ij}} F_B^{ij}$$

These α values should be initialized randomly in the range [0,1] using a uniform distribution for stable learning purposes, as it ensures that values in the output activation map are of the same order of magnitude as the input values before the cross-stitch operation. Another question is how to initialize the separate networks. The authors propose two methods, either first train on these tasks separately or use the same initialization and train them jointly with the cross-stitch units. Learning rates are also initialized in the range [0.1, 0.9], to lead to a better and faster convergence which is a magnitude larger than usual.

The authors [6] tested the cross-stitch units for learning jointly two computer vision tasks: semantic segmentation and surface normal prediction. They compared the Cross-stitch unit network with hard parameter sharing architectures and non multitask learning baselines on the NYU-v2 dataset. Two AlexNets were used, placing cross-stitch units after the pooling and fully connected layers. Cross-stitch units proved to be useful in picking the optimal shared representation as desired. Therefore, these units eliminate the need to search through several architectures by brute

force. As future work, it should be studied where in the network they should be used and how their weights should be constrained.

	Surface Normal					Segmentation		
	Angle Distance (Lower Better)		Within t° (Higher Better)			(Higher Better)		
Method	Mean	Med.	11.25	22.5	30	pixacc	mIU	fwIU
One-task	34.8	19.0	38.3	53.5	59.2	-	-	-
	-	-	-	-	-	46.6	18.4	33.1
Ensemble	34.4	18.5	38.7	54.2	59.7	-	-	-
	-	-	-	-	-	**48.2**	18.9	33.8
Split `conv4`	34.7	19.1	38.2	53.4	59.2	47.8	19.2	33.8
MTL-shared	34.7	18.9	37.7	53.5	58.8	45.9	16.6	30.1
Cross-stitch **[ours]**	**34.1**	**18.2**	**39.0**	**54.4**	**60.2**	47.2	**19.3**	**34.0**

FIGURE 4.3: Results for the experiments on the Surface normal prediction and semantic segmentation tasks. Cross-stitch Networks are only beaten in one aspect in segmentation. [6]

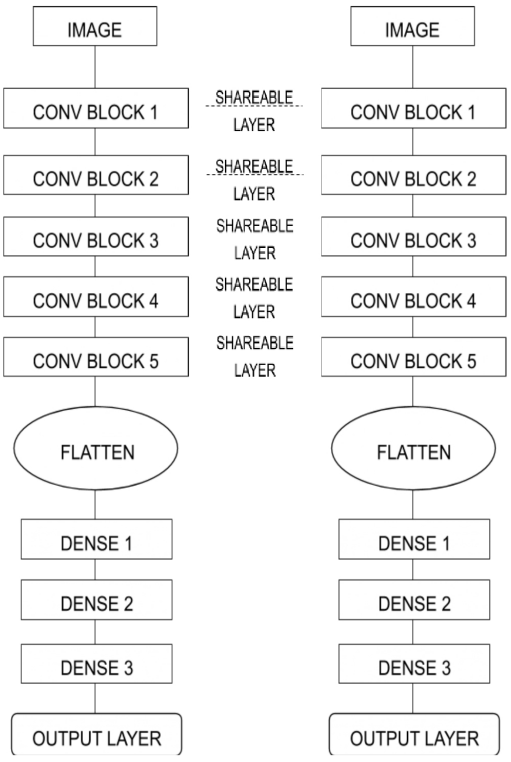

FIGURE 4.4: Cross-stitch Network architecture for two tasks. Cross-stitch units can be placed after each pooling layer in each convolutional block.

Chapter 5

Experiments

In this section, experiments using the UTKFace dataset and the methods explained previously and an evaluation of the trained models on our dyadic dataset will be described and discussed. Then, results obtained with the non-MTL baseline model will be presented. After, a discussion with the results from the hard parameter sharing models will follow. Afterwards, results obtained using Cross-stitch architectures will be analyzed. Finally, a comparison between these two MTL approaches will conclude the section.

5.1 UTKFace

Since the recordings and annotations of this dataset was done in parallel with the multitask learning research, another similar dataset was used to start experimenting and therefore, prepare baselines for when our dataset was completely or partially labelled. As proof of concept of a cross-database application, all models trained on this similar dataset were evaluated on our Dyadic dataset. There are some datasets containing multiple facial attributes for faces in the wild. As an example: the IMDB-Wiki dataset which contains more than 500k images of people along with their labels on age and gender, the MSU-LFW which extends the LFW dataset to study age, gender and ethnicity or the Adience dataset which provides 26k pictures of real-world imaging conditions to estimate gender and age.

However, we choose the UTKFace dataset [23] for three reasons: number of images, number of tasks and also because the faces are cropped and aligned. The UTK dataset is large-scale face dataset for non-commercial reasearch purposes only, with a long age range (from 0 to 116 years old). It consists of more than 20,000 images with annotations of age, gender and ethnicity. Furthermore, it covers large variation in terms of pose, facial expression, illumination, occlusion etc. Faces are cropped and aligned and also are provided with the corresponding facial landmarks.

As mentioned above, available labels for this dataset are:

- **Age:** An integer number ranging from 0 to 116.
- **Gender:** Either 0 (male) or 1 (female).
- **Ethnicity:** An integer from 0 to 4, indicating White, Black, Asian, Indian and Others (such as Hispanic, Latino or Middle Eastern).

The availability of at least three facial attribute estimation tasks, plus the fact that it contains 5,000 images more than the MSU-LFW dataset was crucial in our

decision. Also, the facial landmarks provided could be also used in a future in new applications of our dataset.

FIGURE 5.1: Samples from the UTKFace dataset.

5.2 Implementation details

Different configurations of the methods described in Chapter 4 were trained on the UTKFace dataset in different groupings of the task. All trainings were done using the same train-test split of the UTKFace dataset, and each model configuration was trained three times so that results were statistically significant. All code and implementations needed to reproduce these experiments can be found in the github of the repository [24].

For the MTL architectures, to perform a full analysis of the tasks each configuration of the architectures was trained on each possible pair of tasks as well as for the three tasks at once. Therefore, 4 models were trained for each configuration: one for age and gender, one for age and ethicity, one for gender and ethnicity and finally one for age, gender and ethnicity. Obviously, for the baseline model each task was trained separately. This pairing was useful for the hard parameter sharing models to know which tasks were better trained jointly.

5.3 Baseline Model

In the previous section a description of the baseline model was provided. For the three tasks, the model is the same except for the output layer, which is different due to the different nature of the tasks. The three models were trained for 20 epochs using Early Stopping to prevent overfitting and with a batch size of 32 images. The following table 5.1 shows the results for each task in terms of loss and accuracy. Since each model was trained three different times with three different initializations, these numbers for the performance metrics are the average of the 3 trainings.

Model	Age	Gender	Ethnicity
Baseline Model	88.3889	0.8886	0.7844

TABLE 5.1: Results for the Baseline Model.

Gender and Ethnicity results are reported in terms of accuracy, as they are classification tasks. However, age is solved as a regression problem and therefore its results are expressed in terms of the MSE.

5.4 Hard Parameter Sharing

As described in the previous section, the hard parameter sharing models considered use the same structure than the baseline model, except for the fact that some layers are shared for the tasks. One crucial question stated in the previous section was how many layers should be shared in a hard parameter approach. In computer vision, the standard is to share only the convolutional layers. Therefore, for the following experiments, only the convolutional blocks were considered for sharing. More precisely, different sharings were tried, incrementally. This means that we started sharing the first convolutional block starting from the top, then the first two until all blocks were shared, which results in 5 different configurations. Backwards configurations were also tried, in which we start sharing the last convolutional layers instead. This adds 5 more configurations. These configurations also let us decide where it is more useful to share features, either in the beginning of the network, where low-level features are learned, or in the last layers of the CNN, where high-level features are learned instead. Weights for the networks were initialized randomly.

The following table 5.4 shows the results for each task in terms of loss and accuracy, it is indeed the average of the 3 trainings. The table is split in two ways: each row represents a configuration in terms of number of shared layers, backwards or normal, and the grouping of tasks and each column (or two columns) represents a different task. There are exactly 40 different configurations. The best results for each task are marked in bold.

From this table we can answer many of the already presented questions, and also some new ones. First of all, does Hard Parameter Sharing improve the performance in any of the three tasks? Figure 5.2 shows which percentage of configuration showed improvement in each task. Note that in this figure, each task is considered separately, as we are not interested in a multitask model for predicting all tasks at once, but rather if the training signals of other tasks increase the performance of the considered ones.

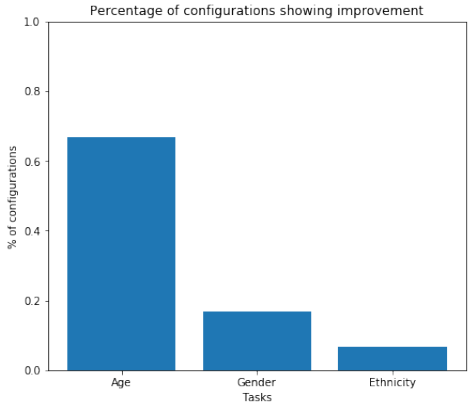

FIGURE 5.2: Percentage of configurations showing improvements in each task.

Model	Age	Gender	Ethnicity
Baseline Model	88.3889	0.8886	0.7844
HPS 1 shared layers age gender	90.183	0.8846	-
HPS 2 shared layers age gender	92.5701	0.8873	-
HPS 3 shared layers age gender	90.0453	0.8892	-
HPS 4 shared layers age gender	91.6989	0.8729	-
HPS 5 shared layers age gender	89.937	0.8562	-
HPS 1 shared layers age ethnicity	91.5793	-	0.7638
HPS 2 shared layers age ethnicity	91.4345	-	0.7702
HPS 3 shared layers age ethnicity	88.2931	-	0.7646
HPS 4 shared layers age ethnicity	86.8621	-	0.7493
HPS 5 shared layers age ethnicity	86.7622	-	0.7162
HPS 1 shared layers gender ethnicity	-	0.8834	0.7638
HPS 2 shared layers gender ethnicity	-	0.8859	0.7571
HPS 3 shared layers gender ethnicity	-	0.8833	0.7617
HPS 4 shared layers gender ethnicity	-	0.8876	0.7832
HPS 5 shared layers gender ethnicity	-	0.8866	0.7754
HPS 1 shared layers 3 tasks	90.1221	0.8891	0.7808
HPS 2 shared layers 3 tasks	87.2329	0.8885	0.772
HPS 3 shared layers 3 tasks	87.8337	0.8849	0.7652
HPS 4 shared layers 3 tasks	86.4123	0.8683	0.7462
HPS 5 shared layers 3 tasks	87.8481	0.8518	0.7112
HPS B 1 shared layers age gender	91.6593	**0.8943**	-
HPS B 2 shared layers age gender	82.9839	0.8772	-
HPS B 3 shared layers age gender	84.9338	0.8687	-
HPS B 4 shared layers age gender	83.4413	0.8708	-
HPS B 5 shared layers age gender	86.6709	0.8551	-
HPS B 1 shared layers age ethnicity	93.0082	-	0.7723
HPS B 2 shared layers age ethnicity	82.8938	-	0.744
HPS B 3 shared layers age ethnicity	87.1743	-	0.7382
HPS B 4 shared layers age ethnicity	82.3652	-	0.7354
HPS B 5 shared layers age ethnicity	87.7482	-	0.7169
HPS B 1 shared layers gender ethnicity	-	0.8796	0.7641
HPS B 2 shared layers gender ethnicity	-	0.8898	0.7857
HPS B 3 shared layers gender ethnicity	-	0.8919	**0.7861**
HPS B 4 shared layers gender ethnicity	-	0.8881	0.7787
HPS B 5 shared layers gender ethnicity	-	0.8857	0.7771
HPS B 1 shared layers 3 tasks	**80.5638**	0.868	0.7411
HPS B 2 shared layers 3 tasks	84.6329	0.8666	0.7373
HPS B 3 shared layers 3 tasks	81.4108	0.8719	0.738
HPS B 4 shared layers 3 tasks	81.965	0.8638	0.7426
HPS B 5 shared layers 3 tasks	87.4871	0.8565	0.709

TABLE 5.2: Results for Hard Parameter Sharing Models.

From 5.2 we can observe that all tasks show improvements with respect to the baseline results. In particular, age is the most benefited task of the three, with almost 70% of configurations showing improvements. However, if we look at the table we see that in terms of improvement of the model (accuracy and loss improvements)

these are slight improvements on the baseline results, specially for gender and ethnicity. In particular, age improves 8 units in the MSE score, gender increases 0.6 points in accuracy and ethnicity increases 0.2 points in accuracy in the best scenarios, if we consider that accuracy ranges from 0 to 100 (in all tables accuracy is shown from 0 to 1). Another interesting question is whether it is more advisable to share layers starting from the top or from the bottom (backwards approach) of the convolutional layers of the network. The following three figures show boxplots of the distribution of the performance metrics in each task for the normal and the backwards approach using the combination of the results for all configurations of shared layers.

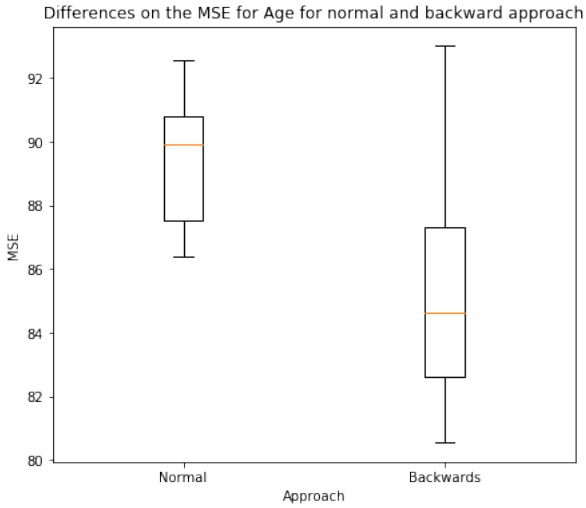

FIGURE 5.3: Difference between normal and backwards in age.

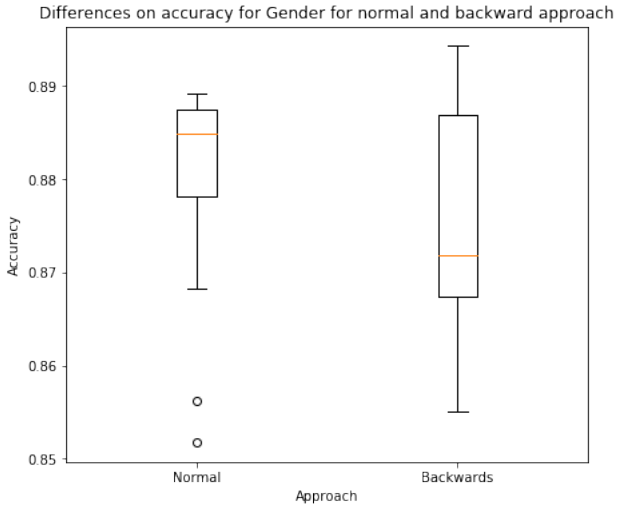

FIGURE 5.4: Difference between normal and backwards in gender.

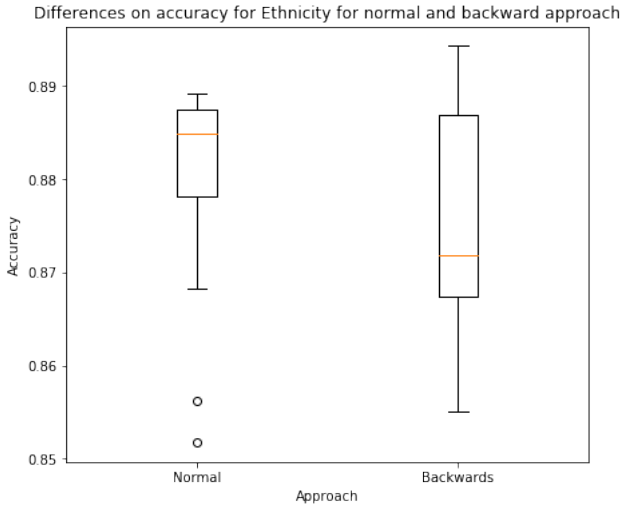

FIGURE 5.5: Difference between normal and backwards in gender.

From these boxplots we can observe that the backwards approach seems to be slightly better than the normal approach. This is clearly shown in the age boxplot, where not only the best result is obtained with the backward approach but the medians are very far from each other. In gender and ethnicity, the differences are not so clear. In fact, in these two classifications tasks the median is better in the normal approach, however, the best results are attained with a backward model. This preference over the backwards approach could indicate that high-level features are more important to be shared than the low-level ones learned in the beginning of any CNN. Also, notice that the box for the backwards approach is constantly larger than for the normal approach. This could be due to the fact that it less stable and it depends a lot on the number of layers shared.

Finally, our last question for the Hard Parameter Sharing architecture was how many layers should be shared within the network. The following three figures represent the evolution in performance in each of the three tasks as we share more layers, also in terms of the normal and backwards approach. Notice that when the maximum number of layers are shared, then both approaches become the same.

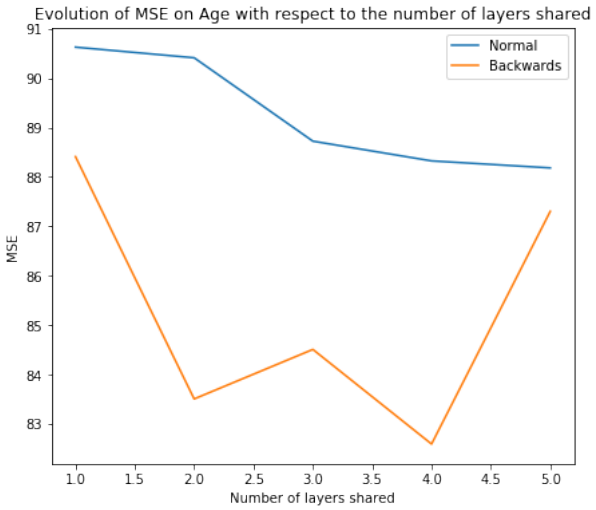

FIGURE 5.6: Evolution of performance for normal and backwards in age when sharing more layers.

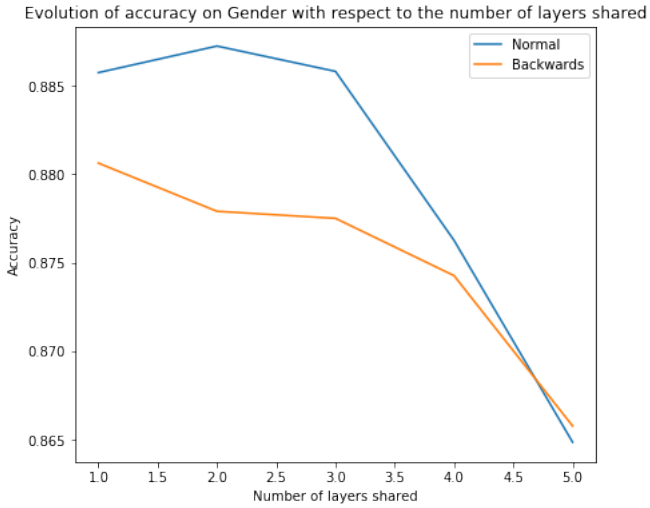

FIGURE 5.7: Evolution of performance for normal and backwards in gender when sharing more layers.

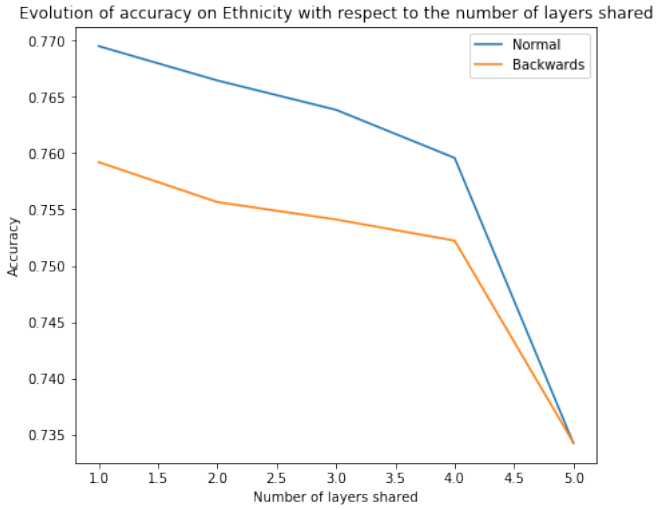

FIGURE 5.8: Evolution of performance for normal and backwards in ethnicity when sharing more layers.

In these plots we observe a difference in the nature of sharing more layers in the age task, compared to the classification tasks. From the first line plot, we observe that we obtain the best performance in age when sharing 2 or 4 layers. However, for the classification tasks, sharing more layers implies a reduction in accuracy. This could indicate that a negative transfer from age to these two other tasks is taking place. However, this is discussed in the Task Grouping section of this Chapter. Therefore, there is not an evident conclusion of how many layers should be shared, as it may depend on the grouping of the tasks as well.

5.5 Cross-stitch Networks

Similar to the hard parameter sharing models trained, the Cross-stitch networks implemented fully use the same structure than the baseline model, except for the fact that some layers are connected through the so-called cross-stitch units. One crucial question stated in the previous section was where and how many cross-stitch units should be used in the network. To keep similarity and to perform a logical comparison with the hard parameter sharing models considered, cross-stitch units were only placed after each pooling layer (in the original paper they placed them in the pooling and fully connected layers). Additionally, as in the configuration of the hard parameter models, cross-stitch units were also placed incrementally starting from both the top and the bottom of the convolutional blocks. In this case, unlike in the Hard Parameter Sharing approach, we refer to backwards as starting from the top, and normal to start form the bottom layers. Although it can cause confusion, this is not casual. In hard parameter sharing approaches the standard is to start sharing layers from the top, however the authors of the Cross-stitch unit started placing the units in the bottom layers. Thus, there is an equivalence between the normal HPS approach and the backwards cross-stitch approach and viceversa. Therefore, we tried 10 different configurations of the Cross-stitch network, each one with an additional unit. The implementation of this layer was done using the custom mode layer creation in Keras. More detail can be found in the repository, in particular in the file *cross_stitch.py*.

Furthermore, for the first configuration, weights were initialized using the baseline weights (except for the cross-stitch unit which were randomly initialized using an uniform distribution so that they ranged between 0 and 1). Then, each configuration used the weights learned in the previous configuration as initialization, fine-tuning it with the new added cross-stitch unit. The following table 5.3 shows the results for each task in terms of loss and accuracy, it is indeed the average of the 3 trainings. The table is split in two ways: each row represents a configuration and each column (or two columns) represent a different grouping of the tasks. In bold, the best results are shown.

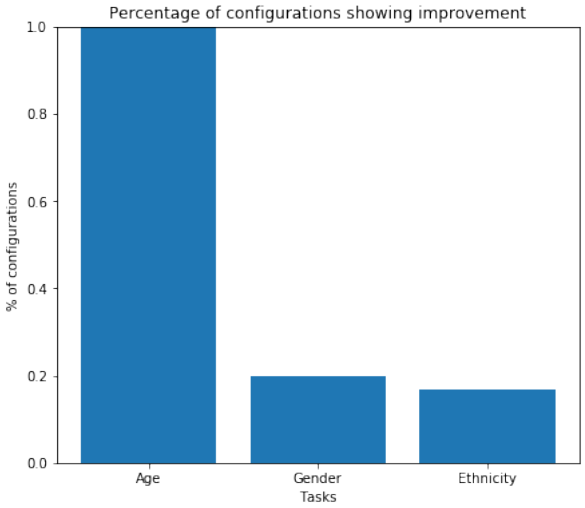

FIGURE 5.9: Percentage of configurations showing improvements in each task.

Model	Age	Gender	Ethnicity
Baseline Model	88.3889	0.8886	0.7844
CS 1 CS units age gender	79.8603	0.8838	-
CS 2 CS units age gender	80.0733	0.8698	-
CS 3 CS units age gender	77.2623	0.8777	-
CS 4 CS units age gender	77.2904	0.8735	-
CS 5 CS units age gender	80.7178	0.8718	-
CS 1 CS units age ethnicity	79.9294	-	0.7585
CS 2 CS units age ethnicity	76.8115	-	0.7538
CS 3 CS units age ethnicity	77.2167	-	0.7509
CS 4 CS units age ethnicity	77.1819	-	0.7611
CS 5 CS units age ethnicity	82.8597	-	0.7525
CS 1 CS units gender ethnicity	-	0.8917	0.788
CS 2 CS units gender ethnicity	-	0.8941	**0.7932**
CS 3 CS units gender ethnicity	-	0.891	0.7884
CS 4 CS units gender ethnicity	-	0.8888	0.7906
CS 5 CS units gender ethnicity	-	0.885	0.784
CS 1 CS units 3 tasks	81.5089	0.8791	0.7657
CS 2 CS units 3 tasks	76.3846	0.873	0.7582
CS 3 CS units 3 tasks	75.3369	0.8778	0.747
CS 4 CS units 3 tasks	78.5396	0.8772	0.756
CS 5 CS units 3 tasks	78.6268	0.8758	0.7522
CS B 1 CS units age gender	85.2453	0.8859	-
CS B 2 CS units age gender	84.888	0.8862	-
CS B 3 CS units age gender	86.5238	0.8814	-
CS B 4 CS units age gender	83.2252	0.8865	-
CS B 5 CS units age gender	79.9315	0.8733	-
CS B 1 CS units age ethnicity	85.7692	-	0.7712
CS B 2 CS units age ethnicity	82.2241	-	0.7593
CS B 3 CS units age ethnicity	77.7165	-	0.7698
CS B 4 CS units age ethnicity	79.4951	-	0.7588
CS B 5 CS units age ethnicity	81.623	-	0.7486
CS B 1 CS units gender ethnicity	-	0.8873	0.7789
CS B 2 CS units gender ethnicity	-	0.8794	0.7736
CS B 3 CS units gender ethnicity	-	0.886	0.774
CS B 4 CS units gender ethnicity	-	0.8891	0.767
CS B 5 CS units gender ethnicity	-	**0.8952**	0.7851
CS B 1 CS units 3 tasks	80.5679	0.882	0.7789
CS B 2 CS units 3 tasks	82.7939	0.8867	0.7749
CS B 3 CS units 3 tasks	83.9139	0.8884	0.7675
CS B 4 CS units 3 tasks	79.7182	0.8853	0.7677
CS B 5 CS units 3 tasks	**74.7696**	0.8725	0.7468

TABLE 5.3: Results for the Cross-stitch unit Networks.

As in the Hard Parameter Sharing section, from this table we can answer many of the already presented questions, and also some new ones. The same analysis is performed. First of all, do Cross-stitch Networks improve the performance in any of the three tasks? Figure 5.9 shows which percentage of configuration showed improvement in each task. As before, in this figure, each task is considered separately, as we

are not interested in a multitask model for predicting all tasks at once, but rather if the training signals of other tasks increase the performance of the considered ones.

From 5.9 we can observe that all tasks show improvements with respect to the baseline results. In particular, age is the most benefited task of the three, with a total improvement since a 100% of configurations showed improvements. However, similar to the results from Hard Parameter Sharing, if we look at the table we see that these are slight improvements on the baseline results, specially for gender and ethnicity. In particular, age improves 8 units in the MSE score, gender increases 0.7 points in accuracy while ethnicity improves almost one point in accuracy in the best scenarios. Another interesting question is whether is it more advisable to place the cross-stitch units starting from the top (backwards) or from the bottom (normal approach) of the convolutional layers of the network. Three boxplots of the distribution of the performance metrics in each task for both approaches.

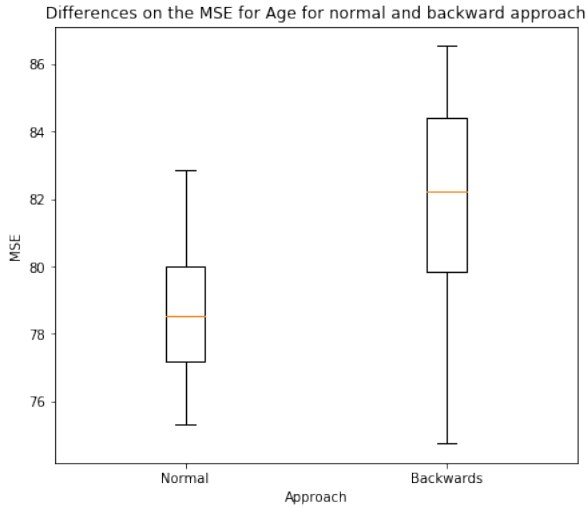

FIGURE 5.10: Difference between normal and backwards in age.

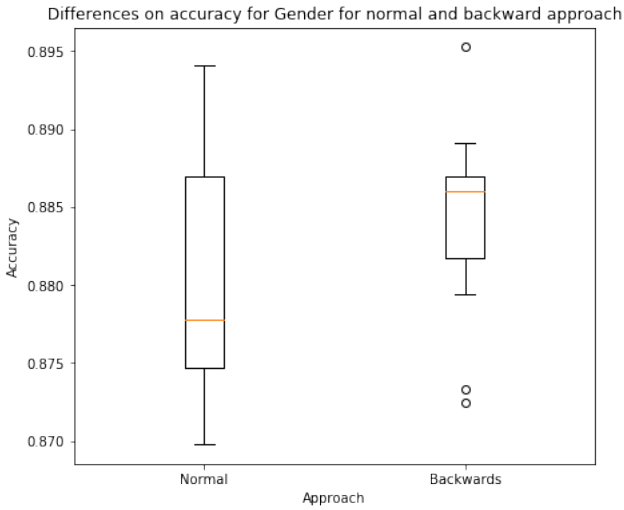

FIGURE 5.11: Difference between normal and backwards in gender.

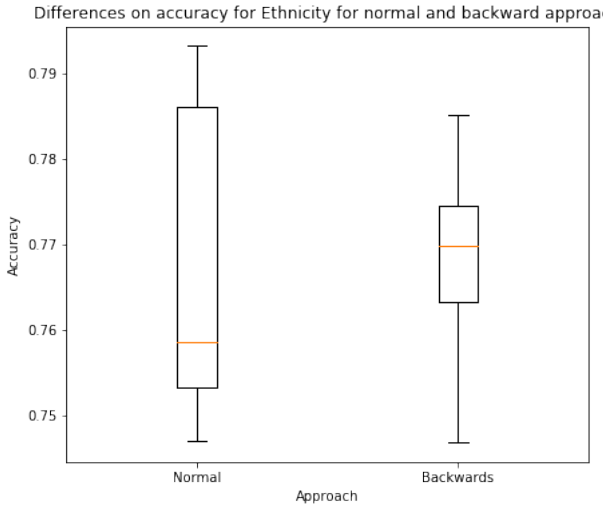

FIGURE 5.12: Difference between normal and backwards in ethnicity.

From these boxplots we can observe that the normal approach seems to be slightly better than the backwards approach. In fact, the results are almost identical as in the Hard Parameter Sharing section (if we consider the equivalence between normal in CS and backwards in HPS). This is clearly shown in the age boxplot, where not only the best result is obtained with the normal approach but the medians are very far from each other. In gender and ethnicity, the differences are once again, not so clear. In fact, in these two classifications tasks the median is better in the backwards approach, however, it seems that there is a higher variance for the normal approach, as the best and worst results are obtained through this approach. This preference over the normal approach could indicate once again that high-level features are more important to be shared than the low-level ones learned in the beginning of any CNN.

Finally, our last question for Cross-stitch Networks was how many cross-stitch units should be placed within the network. The following three figures represent the evolution in performance in each of the three tasks as we place more units, also in terms of the normal and backwards approach. Notice that when the maximum number of units are placed, this time both approaches are not exactly the same, since weights may differ due to the different nature in their initialization.

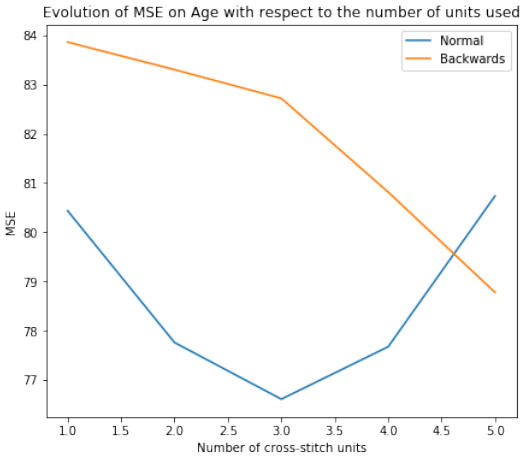

FIGURE 5.13: Evolution of performance for normal and backwards in age when placing more units.

FIGURE 5.14: Evolution of performance for normal and backwards in gender when placing more units.

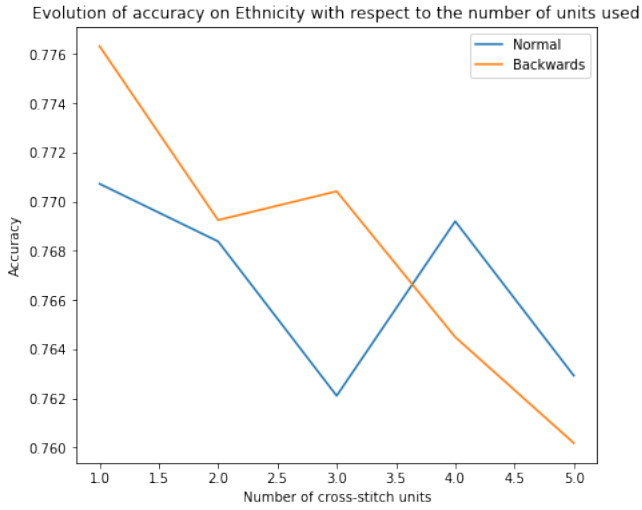

FIGURE 5.15: Evolution of performance for normal and backwards in ethnicity when placing more units.

In these plots we observe a difference in the nature of placing more units in the age task, compared to the classification tasks, but it is not as significant as it was in the Hard Parameter Sharing models. From the first line plot, we observe that we obtain the best performance in age when placing 3 cross-stitch units (there is a minimum on the MSE in that point). However, for the classification tasks, the number of units that should be placed is not as clear. In both tasks, adding new units can cause an increase or a decrease in performance. This could be due to the fact that Cross-stitch units can mitigate the negative transfer from age to these two other tasks that we observed in the last section. And performance may depend on the convergence of the α values, if they can successfully stop the transfer by setting some values to zero. However, this is discussed in the next section of this Chapter.

Therefore, there is not an evident conclusion of how many units should be placed, because even though that theoretically the maximum number of units should be placed, a bad convergence of the α values could lead to a reduction in performance, and the risk is increased as we add more and more units.

5.6 MTL Model Comparison

Another matter of interest was which MTL approach would perform better on the UTKFace dataset, if the simple and standard MTL approach of Hard Parameter Sharing, or a more advanced technique such as the Cross-stitch Networks. So far, we have seen that both approaches slightly outperform the single-task baseline. A first comparison between models can be made by determining which approach outperforms better the baseline, in terms of percentage of configurations and also in terms of performance metrics. We can combine figures 5.2 and 5.9 to obtain the following figure.

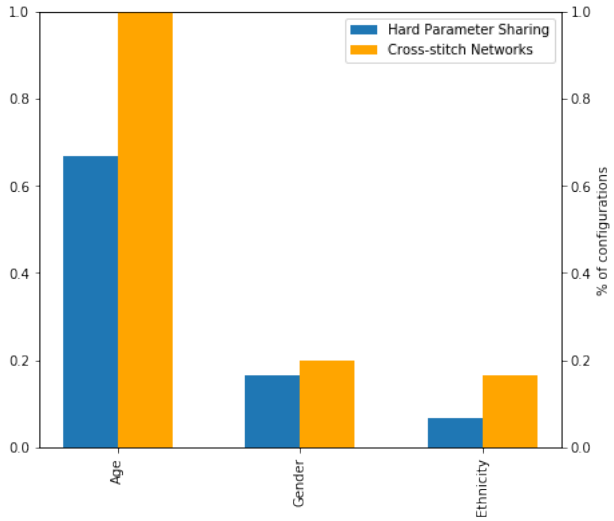

FIGURE 5.16: Percentage of configurations showing improvements in each task.

Clearly, in terms of percentage of configurations outperforming the baseline, Cross-stitch Networks represent a better choice. This does not necessarily mean that the best performance is achieved through this approach, but it indicates that it is more stable and consistent than Hard Parameter Sharing, probably due to the fact that cross-stitch units decide the amount of sharing required in each layer through backpropagation, unlike what happens in Hard Parameter Sharing where the amount of sharing is fixed. In terms of performance, we can analyze through boxplots the comparison for each task, similar to the comparison of the backwards and normal approach done in the last sections.

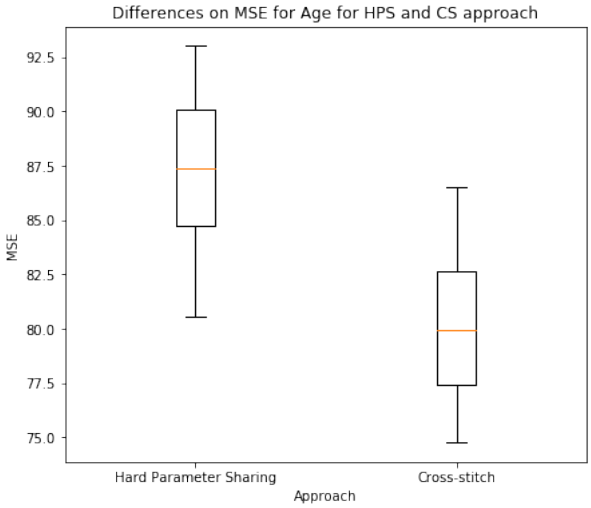

FIGURE 5.17: Difference between HPS and CS Networks in age.

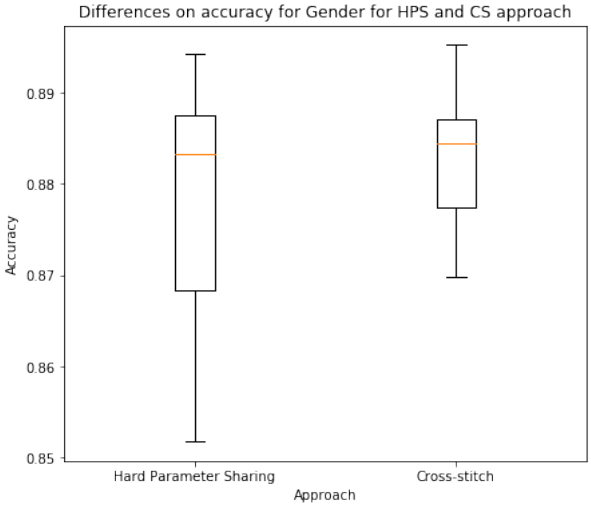

FIGURE 5.18: Difference between HPS and CS Networks in gender.

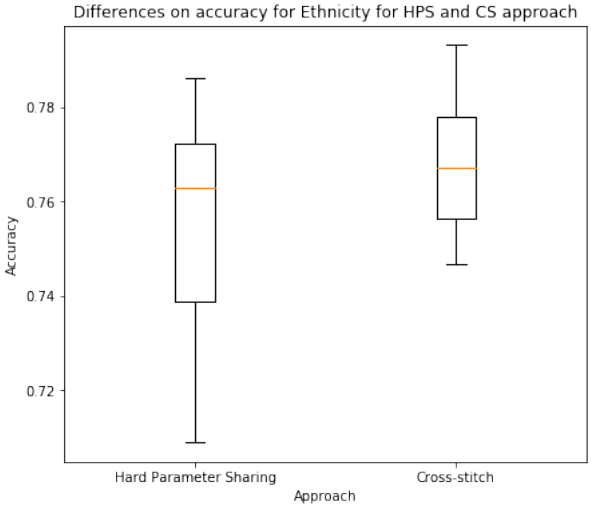

FIGURE 5.19: Difference between HPS and CS Networks in ethnicity.

In terms of performance, from these boxplots, we can observe that Cross-stitch Networks are also a better choice. In the three tasks, not only the median of the results is better (in gender and ethnicity the difference is small) but also the best results are achieved in this approach, specially in age and ethnicity where the difference is important (6 points in MSE in age and 0.7 points in accuracy in ethnicity), while the worst results are attained through Hard Parameter Sharing configurations. Also, we see that the boxes are smaller for Cross-stitch Networks, meaning that there is less variance, implying once again that this a more stable method than Hard Parameter Sharing. Finally, it is also interesting to compare the evolution of performance when adding more cross-stitch units or sharing more layers.

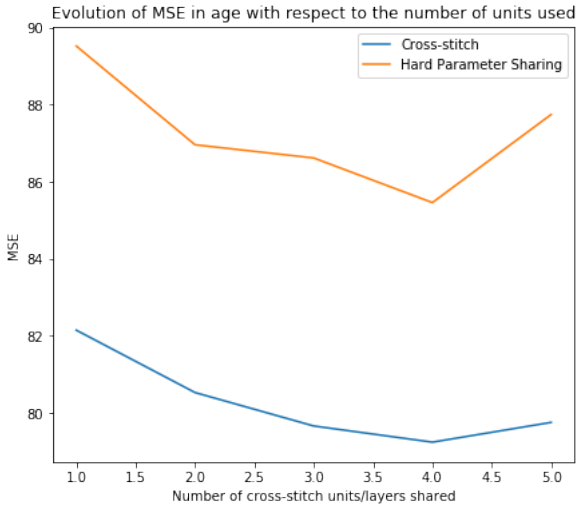

FIGURE 5.20: Evolution of performance for CS Networks and HPS in age when placing more units/sharing more layers.

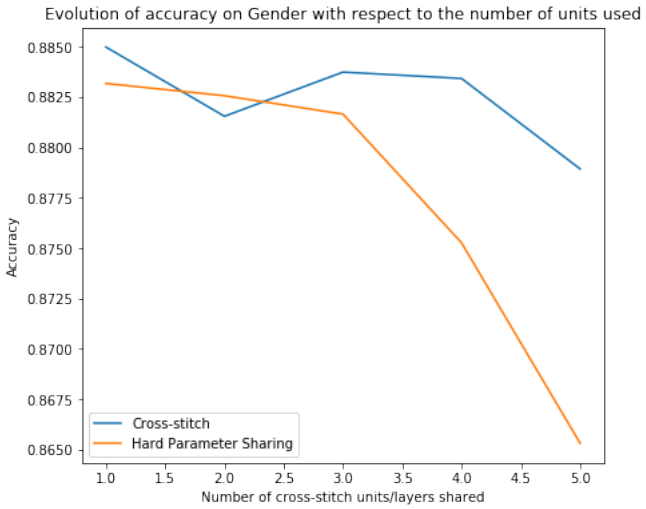

FIGURE 5.21: Evolution of performance for CS Networks and HPS in gender when placing more units/sharing more layers.

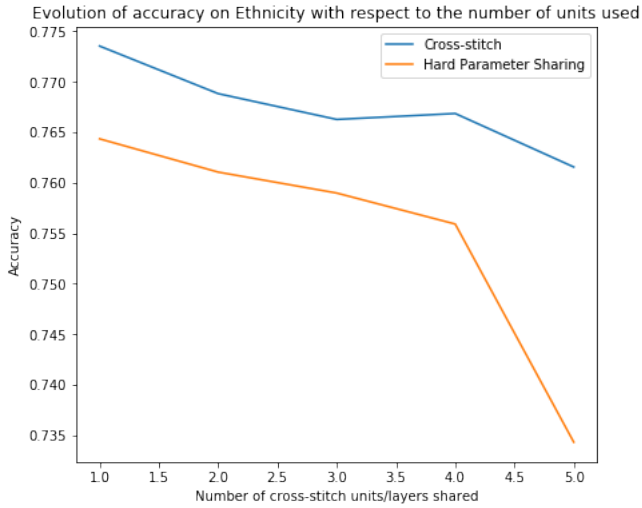

FIGURE 5.22: Evolution of performance for CS Networks and HPS in ethnicity when placing more units/sharing more layers.

These line plots provide very interesting and supporting evidence for the fact that Cross-stitch Networks are a natural evolution of Hard Parameter Sharing. We see a similar tendency in the three plots in both approaches, but the decaying of performance when adding a fifth unit in ethnicity and gender is stopped in Cross-stitch Networks while it is clearly visible in Hard Parameter Sharing. This represents more evidence to the fact that since CS Networks can control the amount of sharing, sharing too much does not have such a strong impact than in HPS. In some way, we can consider HPS as a subset of the hypothesis space of the CS Networks. From this plot and this last observation we can also see, that since CS Networks control the amount of sharing it can lead to these little differences in performance. The line plots have similar tendencies but are displaced in the vertical axis. For all of the reasons shown above, we can see that as expected, Cross-stitch units provide a better alternative for Multi-task Learning than the standard Hard Parameter Sharing approach.

5.7 Task Grouping

One of the crucial tasks in MTL, as stated in Chapter 3, is when to share. From tables 5.4 and 5.3 we can analyse the relatedness of each task. This is an interesting analysis, since some inter-relations between tasks could lead to a negative transfer, and therefore to a bad exploitation of the training signals of other tasks, and finally to a bad implementation of a multitask model. The following figures show for each task performance depending on which auxiliary tasks are used in training. Therefore, each figure contains three boxplots: one when the task considered A is trained with task B (one of the two missing), one for A and C and finally one when all tasks are trained jointly. Also, baseline performance is marked with a red horizontal line.

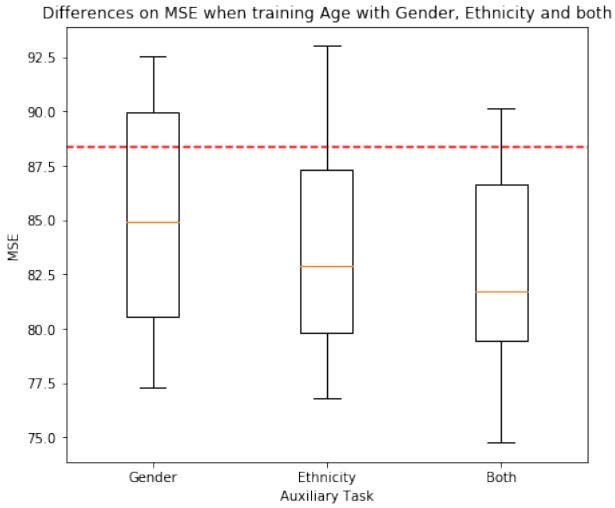

FIGURE 5.23: Distribution of performance in age depending on the task grouping.

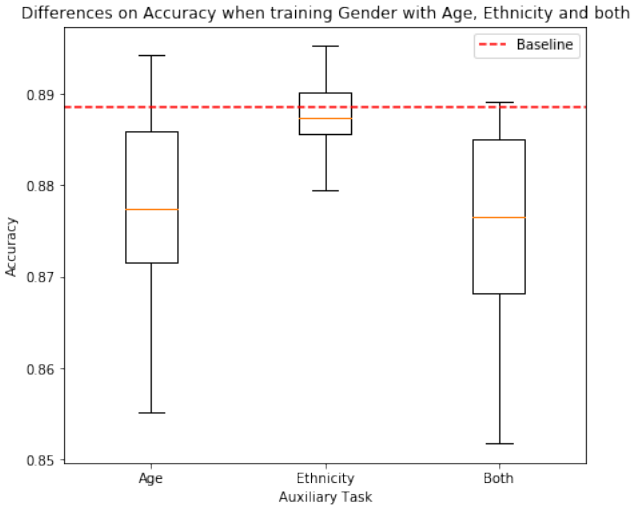

FIGURE 5.24: Distribution of performance in gender depending on the task grouping.

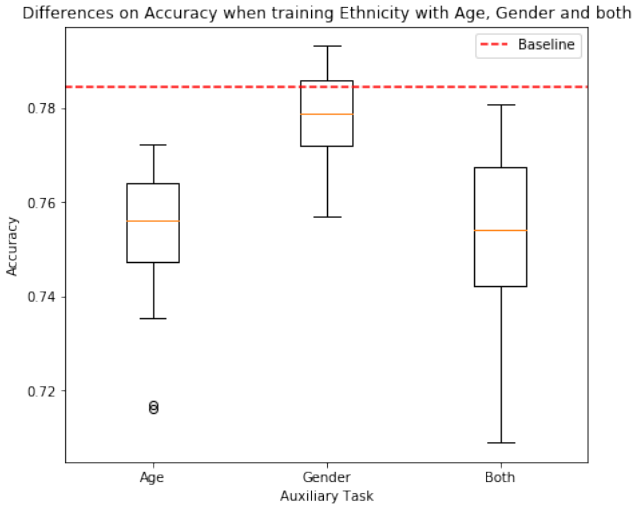

FIGURE 5.25: Distribution of performance in ethnicity depending on the task grouping.

In figure 5.23 we can clearly observe that age is clearly benefited by using the training signals from gender and ethnicity. From the plot, we can see that the three boxes are far below the baseline line. Furthermore, in the three task training setting is where we obtain the best performance when age can receive features from both gender and ethnicity. This represents a clear example of how exploiting MTL can help increase performance in one task. Somehow, we can conclude that the age of a person estimated from its face is highly correlated with his gender and ethnicity.

However, for gender and ethnicity we have a different scenario. Clearly, age represents a negative transfer of information for both tasks, specially for ethnicity. We can clearly see how the boxplots are below the baseline line in the examples where age is trained together with gender and ethnicity. However, gender and ethnicity have a good transfer of information. In fact, the best results in gender and ethnicity are achieved in this task grouping setting, where age is set aside. It may seem surprising that for age the transfer of information only works in one direction, it is unclear why the features learned for age are not useful at all for classifying gender and ethnicity.

However, this negative transfer represents a key example of why Cross-stitch Networks are superior. Their ability to control the amount of sharing between tasks, by possibly blocking transfer between tasks, could prevent some of this negative transfer to take place.

5.8 Class Activation Maps

Class Activation Maps [14] is a technique used to help understanding which regions of an image have more influence in a prediction made by a Convolutional Neural Network. This technique produces a heatmap highlighting the more important areas of the image. In our case, we can take advantage of these heatmaps to observe how Multitask Learning changes the focus of the network. Following, there are some examples of heatmaps obtained using Class Activation Maps and some of the MTL models trained. These maps can be created using the Jupyter Notebook *ClassActivationMap.ipynb* in the repository using the *keras-vis* package.

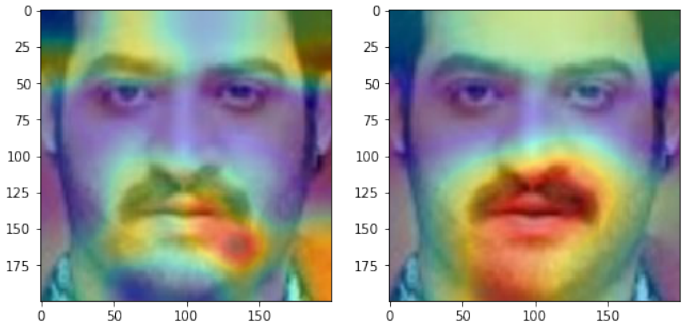

FIGURE 5.26: CAM Generated with the Baseline Model (left) and for a Cross-stitch Network (right) for gender prediction.

Figure 5.26 we see an example of an activation map for gender prediction. In the left, we have the CAM generated by the Baseline model while the right CAM was generated with a Cross-stitch Network which contained 3 units since 3 units resulted to be the optimal number of units for age, and was trained on Gender, Age and Ethnicity. From these heatmaps we can observe that both models focus on similar parts of the Network. However, notice that for the Multitask Model the moustache of the individual is more highlighted than in the baseline model. This moustache is definitive for classifying the individual as male, but it also crucial for determining age and even it can be beneficial for predicting the ethnicity of the individual.

This is an example of the focusing attention principle discussed in Chapter 3. Auxiliary tasks increased the attention in relevant features such as the moustache, which the baseline was ignoring, or at least was not focusing enough attention on it.

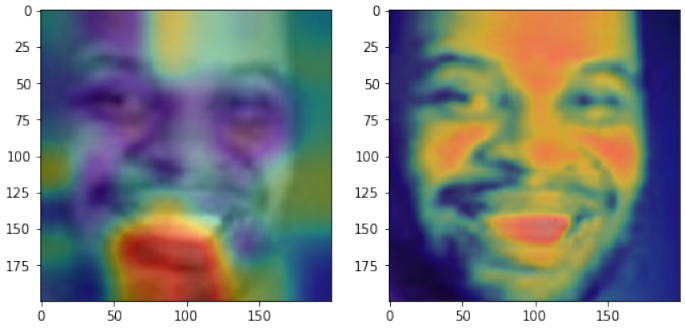

FIGURE 5.27: CAM Generated with the Baseline Model (left) and for a Hard Parameter Sharing (right) for ethnicity prediction.

Figure 5.27 we see an example of an activation map for ethnicity prediction. In the left, we have the CAM generated by the Baseline model while the right CAM was generated with a Hard Parameter Sharing Model where 3 convolutional layers were shared and was trained on Gender, Age Ethnicity. Here we can clearly see how the Hard Parameter Sharing approach enforces the network to extract features useful for the three tasks. In contrast, the Baseline Model extracts task-specific features such as the size of a person lips to determine only ethnicity. This is another effect of MTL discussed in Chapter 3.

5.9 Dyadic Dataset

Finally, as a proof of concept of a cross-database application we performed an evaluation of all the models trained and discussed in the last section in a sample of our Dyadic Dataset. In detail, this sample of the dataset consists in 5 videos of 5 different participants. Each video represents one of the 5 activities recorded in a session. The following table provides more information on the nature of this dataset.

Activity	**Video Length**	**Age**	**Gender**	**Ethnicity**
Animals	6min 23s	26	Female	White
Lego	6min 45s	42	Female	White
Talking	5min 22s	75	Male	White
Ghost	1min 54s	19	Female	White
Looking	1min 26s	25	Male	Indian

TABLE 5.4: Details of this sample dataset.

Each video was segmented frame by frame and each frame was labelled with its corresponding label for age, gender and ethnicity. Each video was recorded with 25fps. Furthermore, as in the UTKFace, using the Haar Cascade Face Detector from OpenCV [16], in each frame the face of the individual was cropped. This resulted in a total of 31,229 images of faces. Then, each model trained was assessed in this new dataset. However, since the image conditions may be different in terms of illumination, face alignment, pose etc. only a subsample of frames satisfying similar conditions to those of the UTKFace was manually selected. From each video an approximate amount of 100 pictures was selected, which amounts to a total of 529 different images. For privacy and confidentiality reasons, no images from this dataset can be displayed in the memory of this thesis. Results are shown in the following tables 5.5 and 5.6.

From these tables we can observe similar results than those obtained in the UTKFace dataset. For example, in all 3 tasks the best results are obtained with a multitask learning configuration. Furthermore, in this dyadic dataset, the improvement with respect to the baseline model is much bigger than it was for the UTKFace dataset. In particular, age improves 62 points in MSE, gender improves 15 points in accuracy and ethnicity improves up to 16 points in accuracy. However, these results are obviously below the ones obtained in the UTKFace dataset, since, as it was mentioned, images from both datasets have different conditions. In particular, ethnicity results should not be taken very seriously as this new dataset only contains two of the five classes. Age and gender results are more representative.

Also, from these results we can obtain the same figures as we obtained in the UTKFace analysis, however, due to the nature of the dataset (different conditions than those in the training images, class imbalance in ethnicity, etc.) some figures can be misleading. All of these figures can be visualized in the notebook *Analysis Dyadic.ipynb* in the repository [24]. Anyway, some explanations extracted from those figures can also be extracted here if we take a closer look. For example, Cross stitch units seem to perform better once again, as we can see in figures 5.28 and 5.29.

Furthermore, in figure 5.30 we can also observe the negative transfer from age to gender.

However, other results obtained in the UTKFace Dataset are not so clear here, for example, in the backwards and normal comparison. For the dyadic dataset, there is

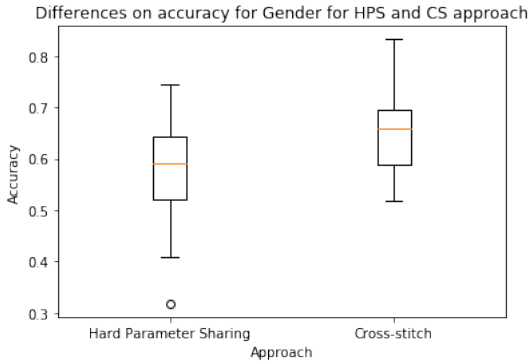

FIGURE 5.28: Distribution of accuracies in gender prediction for HPS models and CS models.

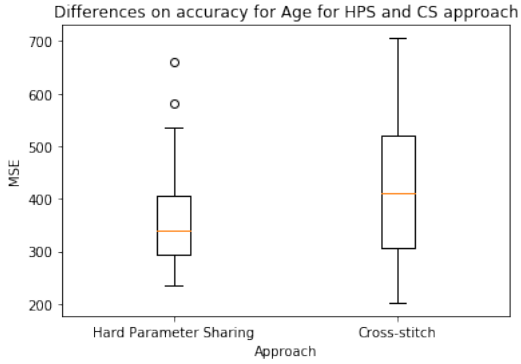

FIGURE 5.29: Distribution of MSE scores in age prediction for HPS models and CS models.

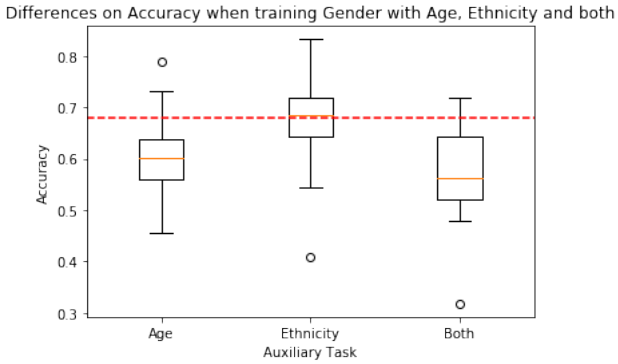

FIGURE 5.30: Gender prediction is worse when trained jointly with age.

not a clear option between these two configurations, although normal seems slightly better. This also happens for determining the optimal number of shared layers or cross-stitch units. This may be due to the noisy nature of this dataset. Despite this, we have seen that similar results occur in this new dataset which helps confirm our hyphotesis that multitask learning does indeed help increase performance for facial attributes tasks.

Model	Age	Gender	Ethnicity
Baseline Model	268.435	0.679	0.561
HPS 1 shared layers age gender	305.834	0.732	-
HPS 2 shared layers age gender	246.918	0.599	-
HPS 3 shared layers age gender	234.871	0.641	-
HPS 4 shared layers age gender	489.928	0.573	-
HPS 5 shared layers age gender	284.552	0.455	-
HPS 1 shared layers age ethnicity	235.982	-	0.48
HPS 2 shared layers age ethnicity	299.023	-	0.536
HPS 3 shared layers age ethnicity	406.232	-	0.517
HPS 4 shared layers age ethnicity	524.272	-	0.37
HPS 5 shared layers age ethnicity	317.104	-	0.438
HPS 1 shared layers gender ethnicity	-	0.701	0.484
HPS 2 shared layers gender ethnicity	-	**0.745**	0.545
HPS 3 shared layers gender ethnicity	-	0.646	0.475
HPS 4 shared layers gender ethnicity	-	0.719	0.46
HPS 5 shared layers gender ethnicity	-	0.544	0.399
HPS 1 shared layers 3 tasks	355.555	0.528	0.684
HPS 2 shared layers 3 tasks	448.229	0.64	0.451
HPS 3 shared layers 3 tasks	383.485	0.676	0.303
HPS 4 shared layers 3 tasks	389.74	0.318	0.471
HPS 5 shared layers 3 tasks	581.548	0.525	0.592
HPS 1 shared layers age gender Backwards	249.559	0.602	-
HPS 2 shared layers age gender Backwards	401.684	0.631	-
HPS 3 shared layers age gender Backwards	331.271	0.595	-
HPS 4 shared layers age gender Backwards	298.847	0.522	-
HPS 5 shared layers age gender Backwards	291.526	0.569	-
HPS 1 shared layers age ethnicity Backwards	369.404	-	0.325
HPS 2 shared layers age ethnicity Backwards	466.957	-	**0.725**
HPS 3 shared layers age ethnicity Backwards	342.736	-	0.16
HPS 4 shared layers age ethnicity Backwards	660.189	-	0.43
HPS 5 shared layers age ethnicity Backwards	386.976	-	0.395
HPS 1 shared layers gender ethnicity Backwards	-	0.68	0.498
HPS 2 shared layers gender ethnicity Backwards	-	0.659	0.478
HPS 3 shared layers gender ethnicity Backwards	-	0.586	0.563
HPS 4 shared layers gender ethnicity Backwards	-	0.41	0.537
HPS 5 shared layers gender ethnicity Backwards	-	0.614	0.366
HPS 1 shared layers 3 tasks Backwards	536.355	0.521	0.461
HPS 2 shared layers 3 tasks Backwards	334.769	0.479	0.431
HPS 3 shared layers 3 tasks Backwards	291.401	0.519	0.627
HPS 4 shared layers 3 tasks Backwards	336.065	0.492	0.147
HPS 5 shared layers 3 tasks Backwards	260.642	0.487	0.493

TABLE 5.5: Results for Hard Parameter Sharing Models

Model	Age	Gender	Ethnicity
Baseline Model	268.435	0.679	0.561
CS Network 1 CS units age gender	380.75	0.567	-
CS Network 2 CS units age gender	284.119	0.687	-
CS Network 3 CS units age gender	509.759	0.688	-
CS Network 4 CS units age gender	459.378	0.536	-
CS Network 5 CS units age gender	**202.556**	0.518	-
CS Network 1 CS units age ethnicity	302.261	-	0.52
CS Network 2 CS units age ethnicity	567.503	-	0.561
CS Network 3 CS units age ethnicity	326.494	-	0.591
CS Network 4 CS units age ethnicity	437.371	-	0.509
CS Network 5 CS units age ethnicity	366.665	-	0.404
CS Network 1 CS units gender ethnicity	-	0.721	0.435
CS Network 2 CS units gender ethnicity	-	0.689	0.51
CS Network 3 CS units gender ethnicity	-	0.683	0.515
CS Network 4 CS units gender ethnicity	-	0.67	0.389
CS Network 5 CS units gender ethnicity	-	0.636	0.422
CS Network 1 CS units 3 tasks	440.323	0.543	0.283
CS Network 2 CS units 3 tasks	520.281	0.547	**0.644**
CS Network 3 CS units 3 tasks	384.737	0.654	0.227
CS Network 4 CS units 3 tasks	634.309	0.584	0.573
CS Network 5 CS units 3 tasks	360.011	0.579	0.596
CS Network 1 CS units age gender Backwards	533.168	0.788	-
CS Network 2 CS units age gender Backwards	471.356	0.63	-
CS Network 3 CS units age gender Backwards	705.452	0.538	-
CS Network 4 CS units age gender Backwards	316.372	0.618	-
CS Network 5 CS units age gender Backwards	516.155	0.637	-
CS Network 1 CS units age ethnicity Backwards	292.595	-	0.456
CS Network 2 CS units age ethnicity Backwards	292.642	-	0.302
CS Network 3 CS units age ethnicity Backwards	286.834	-	0.484
CS Network 4 CS units age ethnicity Backwards	244.518	-	0.38
CS Network 5 CS units age ethnicity Backwards	358.565	-	0.509
CS Network 1 CS units gender ethnicity Backwards	-	0.763	0.305
CS Network 2 CS units gender ethnicity Backwards	-	0.728	0.446
CS Network 3 CS units gender ethnicity Backwards	-	**0.832**	0.488
CS Network 4 CS units gender ethnicity Backwards	-	0.703	0.564
CS Network 5 CS units gender ethnicity Backwards	-	0.696	0.432
CS Network 1 CS units 3 tasks Backwards	480.425	0.688	0.456
CS Network 2 CS units 3 tasks Backwards	611.534	0.72	0.446
CS Network 3 CS units 3 tasks Backwards	560.812	0.604	0.477
CS Network 4 CS units 3 tasks Backwards	564.556	0.619	0.399
CS Network 5 CS units 3 tasks Backwards	288.404	0.661	0.455

TABLE 5.6: Results for Cross Stitch Networks

Chapter 6

Conclusions and Future Work

6.1 Conclusions

As it was stated in the introductory chapter of this thesis, the main objectives were the creation, recording and annotation of a dyadic interaction dataset and the implementation of Multitask Learning architectures to test its efficiency and prepare baseline models for learning tasks jointly in this new dyadic interaction dataset. To that end, some multitask learning have been implemented and tested and also the database is now fully recorded but pending to annotate. Therefore, the following conclusions have been reached:

- **Database Creation:** Creating a face-to-face dyadic database is important to research and implement new paradigms and technologies of interpersonal behavior understanding. Currently, databases to recognize emotions do not usually contain dyadic conversations between individuals, in addition of having worse image quality and much less diversity and amount of data.
- **MTL as a method to improve performance:** Sometimes in machine learning we try many things to improve performance of our models, for example: adding regularization, trying different optimizers or selecting more complex architectures. From the results in this thesis we can conclude that multitask learning can be another similar tool to improve performance over one or many tasks by exploiting the training signals of other related tasks. Even if the improvements are slight, they are statistically significant and constant over many MTL configurations chosen.
- **Information Transfer:** Model performances indicate that there are transfers of information between the different tasks considered. For example, performance in age significantly increases when the networks is learning features for solving gender and ethnicity simultaneously. Furthermore, we could also notice the negative transfer for gender and ethnicity when these tasks are trained along with age. From these two observations we can directly observe the flow of information due to Multitask Learning.
- **Learning to Multitask:** Another conclusion derived from the results of the experiments is that the future of multitask learning is in those architectures that adapt the amount of sharing through backpropagation, and therefore automatize the when and how to share decisions. This was very clear in the Hard Parameter Sharing and Cross-stitch Networks comparison, where the automatization of the when to share decision helped the latter architecture to prevent the negative transfer from age, compared to the Hard Paramater Sharing approach.

- **Drawbacks of MTL:** However, implementing Multitask Learning has some drawbacks and it is not an easy task. For example, there are many hyperparameters to select and many decisions to make. Some of these decisions, as mentioned above, can be automatized with advanced architectures, however, for some other architectures this is not possible and then the selection becomes computationally very expensive (for instance in Hard Parameter Sharing). Another problem is related with the size and the amount of parameters of the model. For example, Cross-stitch units replicate for each task the same baseline network. If this baseline network is a huge one such as the VGG, and as the number of tasks considered increase, this can cause memory problems during training. These issues may overcome the benefits that MTL provides. Finally, as MTL learning is not a common topic in the Deep Learning community, the main frameworks such as Keras or TensorFlow do not have specific MTL packages and implementations can become a little bit tricky.

6.2 Future Work

The Experiments Chapter shows that there are some experiments that could be further studied. For example, specific task weights for the loss function could be an interesting experiment. Specially, since age, gender and ethnicity have different scales. However, this task weighting was not included in the thesis since it can be very demanding computationally if the weights have to be found manually or if they have to be updated dynamically Keras cannot be used for training the models.

Other experiments that could be run in the future are related with the MTL architectures considered in this thesis. Testing new MTL architectures with task weighting could result in obtaining more relevant results than the ones obtained in this thesis, where MTL only improves single-task learning by a small margin. However, there is little literature in Multitask Learning

Finally, once the final dataset is completely annotated there will be many tasks that could be performed jointly, specially emotion recognition along with the three tasks considered in this dataset. Current multitask databases have a few number of tasks available, furthermore the existing ones do not have properly visual or auditory quality, or the participants are actors, or there is only one recorded view for each speaker, etc. Here is the opportunity of achieving a great improvement in the state-of-the-art with the Dyadic Interaction Dataset, which is going to provide more visual quality, nearly a 360^{o} view of the recordings, natural behaviour of the participants, a wide range of ages and ethnicities and 81 hours of recordings of dyadic conversations. Exploiting this database is one of the most important future works.

Bibliography

References

[1] Buechel, S., & Hahn, U. (2016). *Emotion Analysis as a Regression Problem — Dimensional Models and Their Implications on Emotion Representation and Metrical Evaluation.*

[2] Caruana, R. (1997). Multitask Learning. *Machine Learning, 28(1)*, 41-75. doi:10.1023/a:1007379606734

[3] Guo, M., Haque, A., Huang, D.-A., Yeung, S., & Fei-Fei, L. (2018, 2018//). *Dynamic Task Prioritization for Multitask Learning*. Paper presented at the Computer Vision – ECCV 2018, Cham.

[4] Lee, H. B., Yang, E., & Hwang, S. J. (2018). *Deep Asymmetric Multi-task Feature Learning*. Paper presented at the Proceedings of the 35 th International Conference on Machine Learning, Stockholm, Sweden,.

[5] Luvizon, D. C., Picard, D., & Tabia, H. (2018). 2D/3D Pose Estimation and Action Recognition using Multitask Deep Learning.

[6] Misra, I., Shrivastava, A., Gupta, A., & Hebert, M. (2016). Cross-stitch Networks for Multi-task Learning.

[7] Ruder, S. (29 May 2017). An Overview of Multi-Task Learning in Deep Neural Network.

[8] Russell, J., & Mehrabian, A. (1977). Evidence for a Three-Factor Theory of Emotions. *Journal of Research in Personality*, 11, 273-294. doi:10.1016/0092-6566(77)90037-X

[9] Xiao, L., Zhang, H., Chen, W., Wang, Y., & Jin, Y. (2018). *Learning What to Share: Leaky Multi-Task Network for Text Classification.* Paper presented at the Proceedings of the 27th International Conference on Computational Linguistics, Santa Fe, New Mexico, USA.

[10] Zamir, A., Sax, A., Shen, W., Guibas, L., Malik, J., & Savarese, S. (2018). Taskonomy: Disentangling Task Transfer Learning.

[11] Zhang, Y., & Yang, Q. (2018). A Survey on Multi-Task Learning. Cornell University. doi:arXiv:1707.08114

[12] Zhang, Z., Luo, P., Loy, C. C., & Tang, X. (2014, 2014//). *Facial Landmark Detection by Deep Multi-task Learning.* Paper presented at the Computer Vision – ECCV 2014, Cham.

[13] Zhang, Z., Song, Yang, and Qi, Hairong. (2017). *Age Progression/Regression by Conditional Adversarial Autoencoder.*

[14] Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2015). Learning Deep Features for Discriminative Localization.

[15] Language Development Problem, Utterance boundaries, http://ldp-uchicago.github.io/docs/guides/transcription/sect_4.html#tg-4-8-1

[16] OpenCV, Face detection using haar cascades, https://docs.opencv.org/3.4.1/d7/d8b/tutorial_py_face_detection.html.

[17] O. M. Parkhi, A. Vedaldi and A. Zisserman. 'Deep Face Recognition', British Machine Vision Conference, Swansea, UK, 2015.

[18] I. Goodfellow, Y. Bengio and A. Courville. Deep learning, MIT Press, 2016.

[19] C. Busso, M. Bulut, C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J. Chang, S. Lee, and S. Narayanan, "IEMOCAP: Interactive emotional dyadic motion capture database," Journal of Language Resources and Evaluation, vol. 42, no. 4, pp. 335-359, December 2008.

[20] Putnam SP, Rothbart MK (2006). Development of short and very short forms of the Children's Behavior Questionnaire.

[21] Lesa K Ellis, Mary K Rothbart (2001). Revision of the Early Adolescent Temperament Questionnaire.

[22] Russell J.A, Mehrabian A. (1997). Evidence for a Three-Factor Theory of Emotions. Journal of Research in Personality, 11, 273-294.

Datasets

[23] Zhang, Zhifei, Song, Yang, and Qi, Hairong. *Age Progression/Regression by Conditional Adversarial Autoencoder*. UTKFace. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2017

6.3 Github Repository

[24] Repository of the Master Thesis, https://github.com/andreu15/Non-acted-Multi-view-Audio-Visual-Dyadic-Interactions-Project