POLITECNICO DI TORINO

Corso di Laurea Magistrale in Computer Engineering

Tesi di Laurea Magistrale

Emotion Recognition in ambito Neuromarketing e Neuroriabilitazione: sviluppo di un applicazione desktop per esperimenti neuroscientifici e di un algoritmo per il Facial Coding

Relatori

Prof. Gabriella Olmo

Prof. Vito De Feo

Candidato

Mattia Tarquinio

Student ID: 265802

Anno Accademico 2020-2021

This work is subject to the Creative Commons Licence

Abstract

La scarsa diffusione di training neuroriabilitativi riguardo deficit nel riconoscimento delle proprie emozioni in pazienti vittime di lesioni al sistema nervoso centrale e il crescente interesse all'ambito del neuromarketing hanno ispirato questo lavoro di ricerca. Lo studio è basato su un tentativo di produzione di un'applicazione desktop per l'effettuazione di esperimenti neuroscientifici di varia natura che hanno come comune denominatore il riconoscimento delle emozioni del partecipante al fine di permettere un training riabilitativo adeguato, o per meglio comprendere i meccanismi sottostanti che guidano il processo decisionale di un potenziale acquirente. Entrambi gli ambiti di ricerca pongono le loro fondamenta sulla raccolta e analisi di alcuni biomarcatori utilizzando tecniche come EEG, Eye-tracker e GSR. I dati raccolti negli esperimenti sono poi processati da un algoritmo di machine learning di cui si segue lo studio e lo sviluppo, che tenta di predire in maniera corretta l'emozione provata dal partecipante. Particolare enfasi in questo studio viene data al Facial coding che, tramite la registrazione del volto, fornisce una predizione sull'emozione osservata. Il risultato migliore viene cercato dapprima utilizzando come output della rete neurale convoluzionale le label dell'emozione stessa, per poi passare ad un sistema più complesso che utilizza l'attivazione e la valenza emotiva come indicatori del risultato.

Contents

List of Figures

Chapter 1

Introduzione

Questo lavoro di tesi affronta il tentativo di produzione di un'applicazione per l'effettuazione di esperimenti neuroscientifici e i primi passi nell'utilizzo di tecnologie specifiche per la raccolta e il trattamento di dati significativi per dei gruppi di ricerca del settore. Viene mostrato il percorso che si segue quando si collabora ad un progetto di modeste dimensioni, ma che unisce persone appartenenti a vari campi di ricerca che vengono uniti ad uno scopo comune. Altro aspetto che risulterà evidente dalla trattazione dei prossimi capitoli è l'orizzonte temporale del progetto. Ho collaborato con il gruppo strutturato dal professor Vito De Feo per 6 mesi, conscio di prendere parte a qualcosa di cui non avrei visto il completamento con la fine del mio lavoro di tesi, ma che sarebbe stato utile per gettarne le fondamenta. Questa è una prospettiva che da un lato elettrizza perché fa sentire parte di qualcosa di importante e con grandi aspirazioni, ma può risultare frustrante poiché non sarai tu a poter toccare con mano i risultati nel mondo reale del progetto, frutto anche del tuo lavoro. Questo progetto è nato della collaborazione di tre università, i cui studenti hanno collaborato, ognuno secondo il proprio percorso di studio, specializzazione e interesse. Le tre università sono: il **Politecnico di Torino**, l'**Università degli Studi di Torino** e **la University of Essex**. La presenza di colleghi provenienti da questa università estera, che è caratterizzata da una grande multiculturalità, oltre a darmi la possibilità di interfacciarmi con un progetto internazionale, mi ha permesso

di confrontarmi con diverse mentalità e approcci anche molto lontani ad argomenti in comune. Ritengo queste possibilità di grande aiuto per la crescita personale di un professionista di un ambito come il nostro, quello informatico, sempre meno legato a confini nazionali e che sottende l'interfacciarsi costantemente con l'estero, che diventa ogni giorno meno distante.

Non solo, oltre alla diversità di culture la grande varietà tematica degli studenti e professionisti che erano presenti all'interno dei vari sottogruppi di ricerca mi hanno permesso non solo di approcciarmi ad argomenti che altrimenti non avrei mai avuto l'occasione di affrontare, ma di pormi domande che, non essendo strettamente legate al mio percorso di studi, non avrei avuto modo di immaginare. La caratteristica fondamentale di questo progetto che mi ha portato a sceglierlo è il non essere stato pensato come un mero esercizio teorico, ma, anzi, con un'impronta prettamente pratica; le scelte in ogni fase non sono votate solamente al miglior risultato teorico possibile, ma alla sua migliore attuazione possibile nelle condizioni in cui poi dovrà essere utilizzato nel mondo reale. Insieme alle tre università, infatti, per questo progetto hanno collaborato e stanno collaborando il **Centro Puzzle** e la **Fondazione Neurone Onlus**.

Il Centro Puzzle è un centro dell'omonima Cooperativa Puzzle, che nata nel 1998 si pone come obiettivo di donare al paziente e di conseguenza alla sua famiglia, un ritorno alla vita nella maniera più dignitosa possibile, cercando di restituirgli la normalità alla quale era abituato prima dei disagi da cui è affetto. Il centro si occupa di traumi cranio-encefalici e gravi cerebrolesioni acquisite, cura il percorso del paziente sia nella fase acuta e post-acuta, cercando di permetterli un rientro tra le mura domestiche nella maniera migliore possibile, viste le conseguenze neuro comportamentali che questa tipologia di traumi provocano. Il Centro Puzzle è stato il primo centro in Italia, essendo stato inaugurato nel 2001, ad occuparsi dei traumatizzati cranici e delle gravi cerebro lesioni acquisite.

La Fondazione Neurone Onlus nasce invece, nel 2006, con lo scopo ultimo, tramite studi di ricerca all'avanguardia, di aiutare a prevenire e curare malattie sia neurologiche che psichiatriche, come ad esempio l'Alzheimer, il morbo

di Parkinson, l'epilessia e la schizofrenia. Possiede quindi un team formato da numerosi specialisti in ambiti legati principalmente alla neurologia (neurologi, neurochirurghi, neuropsicologi), ma anche genetisti, psichiatri, bio-ingegneri e numerose altre figure professionali. In particolare, la Fondazione gestisce il Caserta Hub Lab, una struttura dedicata alla neuro-cibernetica, nonché la prima del suo genere nel Sud Italia. Questa struttura ha a disposizione numerose tecnologie, anche molto all'avanguardia, che permettono di analizzare e studiare i segnali cerebrali, nonché altre importanti funzioni del corpo umano, evidenziate prettamente tramite biomarcatori. Queste due strutture collaborano al progetto in quanto saranno i due luoghi dove, una volta finiti di sviluppare, gli esperimenti neuroscientifici prodotti saranno utilizzati attivamente, inizialmente per testarne l'efficacia e si spera successivamente per migliorare quelli che sono i processi che già vengono utilizzati, come per esempio nel Centro Puzzle per quanto riguarda la neuroriabilitazione dei pazienti.

Tornando a questa tesi, nello specifico è stata strutturata in 5 capitoli, oltre questa breve introduzione, ognuno trattante un aspetto o una fase di quello che è stato il mio lavoro all'interno di questo progetto.

Nel secondo capitolo ho parlato dei due ambiti applicativi che hanno interessato il progetto, infatti la produzione software nella quale sono stato coinvolto tratta l'effettuazione di esperimenti neuroscientifici, ma in due diversi campi applicativi. Il primo campo è la già citata neuroriabilitazione, con il tentativo di trovare un percorso più rapido che permetta ai pazienti del Centro Puzzle di tornare alla normalità nella maniera più rapida possibile. Il secondo campo è il neuromarketing, una disciplina che verrà analizzata e spiegata nel capitolo, insieme ai numerosi esperimenti proposti nell'applicazione. Gli esperimenti di neuromarketing verranno condotti nel Caserta Hub Lab grazie alle loro tecnologie. Infine, vengono brevemente analizzate le varie tecnologie che avranno un ruolo, anche futuro, nella fase sperimentale trattandone gli aspetti principali e il loro funzionamento ad alto livello.

Il terzo capitolo tratta la produzione dell'applicazione desktop che ho

dovuto sviluppare insieme alla creazione degli esperimenti stessi ideati dai gruppi di ricerca. Affronta le scelte tecniche effettuate, dal linguaggio di programmazione scelto con annesse decisioni legate alla GUI, fino a parlare delle varie librerie di terze parti utilizzate. Si cerca di spiegare perché si è operata una determinata scelta, quali sono i vantaggi e le situazioni che hanno reso necessario un dato modulo. Questi discorsi sono fatti in particolare riguardo le librerie di terze parti che mi hanno permesso di adattare l'applicazione a quelle che erano le necessità specifiche dei singoli esperimenti.

Il quarto capitolo affronta la seconda parte del progetto, quella relativa allo sviluppo dell'algoritmo per la face emotion recognition. In particolare, viene fatta una trattazione teorica degli argomenti che saranno poi applicati teoricamente, partendo dal Machine Learning in generale, per andare poi ad approfondire le reti neurali convoluzionali, tecnologia principale scelta per la nostra ricerca. Infine, si affrontano alcune task di interesse legate alla Computer Vision e che risultano molto vicine allo scopo del progetto. Con questo capitolo si pone come scopo quelli di fornire in maniera molto coincisa delle basi teoriche per poter comprendere le scelte pratiche operate successivamente.

Nel quinto capitolo viene affrontato l'effettivo sviluppo dell'algoritmo in collaborazione con i ragazzi della University of Essex. Viene mostrato il punto di partenza dell'applicazione Vibe, con il relativo modello e metodo di classificazione. Dopo viene evidenziato come e perché l'output delle immagini viene cambiato da delle etichette indicanti le emozioni, ad un doppio valore di attivazione e valenza emozionale come indicatori dell'emozione stessa. Infine, si parla dello sviluppo vero e proprio diviso in tre parti: Face Detection, Feature Extraction e Classificazione. Per ogni fase vengono evidenziate le varie alternative analizzate e loro caratteristiche.

Il sesto capitolo rappresenta una breve conclusione, in cui vengono descritti i temporanei risultati ottenuti al momento del termine del mio lavoro all'interno del progetto e le conclusioni che ho tratto dal lavoro. Vengono

inoltre analizzati anche possibili sviluppi futuri del progetto, con miglioramenti che ho discusso con i miei colleghi e idee che ho avuto personalmente per poter forse migliorare ulteriormente i risultati ottenuti.

Chapter 2

Ambiti di Applicazione

Il progetto, come detto, mira a creare un ambiente entro il quale poter effettuare numerosi esperimenti che attraverso una diversa metodologia e alcuni strumenti in comune, mirano però ad analizzare in maniera profonda e inconscia le emozioni e le sensazioni che il partecipante prova quando viene sottoposto a determinati stimoli di vario genere. Questo ambiente è pensato per unire due progetti con scopi molto lontani, ma che hanno in comune il tipo di informazioni che intendono analizzare. Il gruppo di ricerca era infatti diviso in due sottogruppi di cui io e i miei colleghi eravamo il collante e la parte in comune. Da un lato vi era un gruppo che si occupava di alcune ricerche riguardanti il neuromarketing e che voleva approfondire come gli stimoli da loro prodotti agivano in maniera inconscia su quello che poteva essere un potenziale acquirente. Da questo gruppo sono stati prodotti un numero maggiore di esperimenti, alcuni che si basavano sulla visione di alcuni video creati ad hoc per provare ad influenzare il partecipante per quanto riguarda le sue scelte; altri che invece andavano ad analizzare l'efficacia di alcuni siti web sull'utilizzatore, così da testarne l'efficacia sul partecipante che doveva navigarci effettuando alcuni semplici task. Questi esperimenti servivano ad analizzare in maniera più approfondita come viene inconsciamente influenzato il comportamento di un potenziale acquirente che si trova davanti a degli stimoli pensati per spingerlo a comprare o semplicemente per fargli vivere un'esperienza più piacevole durante un qualunque tipo di acquisto.

Delle basi simili, ma con finalità totalmente diverse, ha il secondo progetto; il gruppo che se ne occupa ha come scopo la neuroriabilitazione di pazienti che in seguito a delle problematiche, principalmente si parla di trauma cranico, perdono la capacità di riconoscere le emozioni altrui o proprie, e proprio su questo secondo aspetto lo studio si va a indirizzare. Il paziente viene quindi sottoposto a degli stimoli multimediali che dovrebbero provocare in lui delle risposte emotive. In questo modo tramite una serie di strumenti che tratteremo nello specifico in seguito, si dovrà stabilire quanto sia accurata e veritiera la sensazione del paziente. Prima di passare all'analisi più dettagliata dei due ambiti applicativi, mi sembra importante sottolineare come una disciplina tecnica come l'informatica, e più nello specifico il machine learning, può essere applicato in un vasto ventaglio di argomenti permettendo di poter approcciare a problemi grandi e trasversali che affliggono ogni giorno le persone. Per questo è importante per una figura che si approccia a questo mondo essere competente nel proprio ambito, ma il quanto più aperto possibile ad interessarsi così da acquisire competenze, anche minime, ma orizzontali, che gli permettano di svolgere al meglio il proprio lavoro all'interno di un team variegato come quello che ho descritto. Per questo prima di passare alla parte puramente tecnica ho dovuto approcciarmi a questi due mondi, capendo almeno a grandi linee la teoria presente dietro a i loro studi e i loro progetti. Presento in seguito brevemente questi due ambiti e gli esperimenti ad essi associati, che ho dovuto implementare per gettare le basi di un futuro utilizzo su partecipanti in carne ed ossa.

2.1 Neuromarketing

2.1.1 Cenni sul Neuromarketing

Il neuromarketing è una sotto-branca della neuroeconomia e, come si può intuire dal nome, è una disciplina che attraversa varie materie tra loro molto lontane: esso va infatti a studiare come vengono raccolti ed elaborati dal corpo umano e in particolare dal cervello alcuni stimoli a cui una persona è sottoposta (medicina/neurologia), così da capire come questi stimoli possano influenzare il processo decisionale umano (psicologia), il tutto con il fine di creare campagne promozionali o progettare strategie d'acquisto che siano il più efficaci possibili sul consumatore (economia). Come è semplice immaginare è una scienza di recente formazione che ha visto la sua importanza crescere con l'aumento dei mass media, ma soprattutto con la loro presenza, diventata sempre più preponderante all'interno della vita quotidiana delle persone. La sua teorizzazione è datata 2002 per mano di un Professore di analisi di mercato della Rotterdam School of Management di nome Ale Smidts che coniò il termine e diventò uno dei volti più famosi di questa nuova branca. Quello che però rese questa materia molto famosa, sono gli esperimenti compiuti dai grandi brand di qualsiasi settore, che continuano ad investire ogni anno milioni e milioni in questo settore per migliorare a volte anche solo la percezione del consumatore riguardo la propria immagine. Famosa al mondo è diventata la cosiddetta "Pepsi Challenge" dapprima sperimentata su larga scala nei centri commerciali e dopo ripresa e analizzata scientificamente da Read Montague, che si avvalse di strumenti come la risonanza magnetica per analizzare la risposta inconscia dei partecipanti allo studio [1]. Questo esperimento servì ad evidenziare un bias cognitivo nella valutazione dei consumatori, essi infatti reagivano in maniera differente se alla bevanda che veniva loro proposta era associato oppure no il brand corrispondente; questa tendenza era confermata anche dalle analisi scientifiche attuate nello studio che evidenziavano una diversa reazione cerebrale alla stessa bevanda nel caso le informazioni in mano al consumatore fossero diverse. Questo dimostrava come nella scelta entrassero in gioco anche le emozioni e di come

quindi la Coca-Cola avesse sempre lavorato in maniera ottimale sulla propria immagine, tanto che effettivamente il consumatore era portato a preferirla, scelta che spesso non avveniva in caso il brand non fosse stato esplicitamente dichiarato. Proprio a questo mira il neuromarketing, andare ad analizzare i processi decisionali del consumatore, che hanno quindi non solo una componente conscia e razionale, dovuta a dei semplici calcoli di beneficio e convenienza, ma anzi sono fortemente influenzati dalle emozioni della persona e quindi risulta di primaria importanza costruire delle campagne di marketing che vadano a saziare anche questi bisogni inconsci dell'animo umano, così da risultare più attrattive e appetibili. Finora si è trattato l'argomento sottolineando come finalità unicamente la possibilità di migliorare l'efficacia di campagne pubblicitarie, in realtà com'è facilmente deducibile il concetto di prodotto nel mondo moderno è molto inclusivo, per questo si è cominciato ad applicare questo argomento di studio non solo alla pubblicità ma a diversi ambiti e in maniera sempre più importante. Le principali motivazioni sono da ricavarsi in un grande calo del costo della tecnologia e, nello specifico, dei dispositivi di misurazione biometrica, che permettono quindi di raccogliere i dati necessari a questi studi senza per forza dover avere i capitali di una multinazionale. Parimenti al progresso tecnologico è di grande importanza anche il grande avanzamento nel campo delle neuroscienze con un ampio numero di studi sempre più approfonditi sul funzionamento del sistema nervoso sia centrale che periferico. Questi progressi uniti alla crescente consapevolezza che ha cominciato ad espandersi, rispetto alla iniziale incredulità, riguardo la fondamentale importanza che le emozioni e l'inconscio ricoprono all'interno del processo decisionale dell'essere umano e considerata la crescente competitività di un mondo globalizzato che richiede uno sforzo in più per affermarsi sul mercato, hanno contribuito in maniera definitiva allo sviluppo di questa disciplina, così da poter risultare più appetibili sul mercato e ridurre gli sprechi comprendendo eventuali punti deboli del prodotto prima della fase di produzione. Per questo il Neuromarketing viene ormai applicato nei campi più disparati oltre a quello pubblicitario, come per esempio nella produzione di prodotti multimediali per predirne il successo studiando i punti in cui il

fruitore è più o meno trasportato, così da produrre un prodotto equilibrato e che mantenga piacevolmente l'attenzione dello spettatore, si parla sia di film che di serie tv o programmi televisivi; basti pensare agli investimenti che un'azienda importante come Disney effettua per valutare i propri prodotti prima di lanciarli sul mercato per prevederne il successo e comprendere se dover cambiare qualcosa prima dell'uscita [2]. Due ambiti che intendo nominare in quanto collegati agli esperimenti che abbiamo implementato e che spiegherò in seguito; il primo è il product placement, uno dei cui significati è quello di studiare qual è il modo migliore di posizionare i prodotti che si intende vendere così da mettere in risalto la merce, non solo singolarmente ma anche creando delle sinergie tra prodotti che aumentano l'attrattività generale degli oggetti nella vetrina o sullo scaffale di un negozio. L'altro ambito entro cui andremo ad operare con gli esperimenti all'interno dell'applicazione è il web design, si può infatti capire grazie ad alcuni strumenti, di cui parleremo in un paragrafo successivo, quali siano le aree del sito che attirano di più l'attenzione del fruitore del servizio, oppure capire se alcune task che una persona deve eseguire su quel dato sito provocano frustrazione poiché non sono così immediate o semplici. Questo tipo di studio può essere utilizzato per rendere più facilmente navigabili dei siti web, oppure per inserire della pubblicità all'interno di essi nelle aree che l'occhio umano va a prediligere all'interno della pagina, così da ottimizzare il loro inserimento, controllando però che il processo non vada ad influire troppo sulle sensazioni dell'utente che utilizza il sito per non comprometterne l'esperienza. Andiamo allora a vedere gli esperimenti che abbiamo implementato dividendoli in due categorie, quelli che si concentrano sul product placement e la pubblicità, che hanno come filo di unione la birra, e quelli che si concentrano sul web design.

2.1.2 Esperimenti sulle marche di birra

Come detto il product placement è un ambito in cui il neuromarketing viene utilizzato per capire come il posizionamento dei vari prodotti all'interno di una vetrina o di uno scaffale, possa incrementare le possibilità di vendita degli oggetti al suo interno, aumentandone l'appetibilità. Il primo esperimento va

infatti ad investigare i principi dello shelf marketing, che va specificatamente ad analizzare varie caratteristiche dei prodotti per quanto riguarda il posizionamento sullo scaffale, dal contrasto dei colori alla forma del package entro cui il prodotto è confezionato, fino ad arrivare alla varia importanza di alcuni specifici brand che possono attirare una maggiore attenzione. Basti pensare che molte marche di bevande, come ad esempio la già citata Coca-Cola, forniscono ad esercizi di dimensioni medio-grandi come supermercati o centro commerciali dei frigoriferi da esposizioni dentro cui esporre i propri prodotti. Questi frigoriferi sono a volte riforniti e riempiti secondo un ordine ben specifico da alcuni addetti del brand stesso che sono istruiti su quale sia il posizionamento migliore per permettere una vendita ottimale dei prodotti. Già abbastanza è stato detto invece sul produrre delle pubblicità che risultino efficienti, andando ad agire su vari fattori come video, musica, colori e posizione di messaggi che si vuole abbiano presa sul potenziale acquirente, si vuole quindi vedere come questi fattori vengano elaborati dai partecipanti all'esperimento e quali sembrano avere un impatto maggiore riguardo alla scelta di comprare o meno una data marca di birra, sia in ambito conscio, quindi con dei questionari di gradimento, sia in ambito inconscio utilizzando quindi una serie di strumenti che vanno a registrare i dati associati ad una serie di biomarcatori da noi scelti. Su questa linea si basano i primi due esperimenti che ho dovuto inserire all'interno dell'applicazione, entrambi scelti dei partecipanti sia abituali fruitori di birra che no, cercano di analizzare come e se il loro percorso decisionale venga influenzato da ciò a cui vengono sottoposti. Entrambi i test, infatti, si basano su video che mi sono stati forniti dalle colleghe che hanno focalizzato la loro tesi in ambito neuromarketing. Nel primo è unicamente mostrato uno a caso tra due planigrammi che sono stati costruiti (uno per gruppo di partecipanti), queste immagini rappresentano uno scaffale contenente 60 bottiglie di birra rappresentanti 20 marche diverse. Le due immagini, che sono mostrate per 20 secondi, raffigurano una diversa disposizione delle bottiglie sullo scaffale.

Figure 2.1. I due planogrammi con gli scaffali contenenti le birre mostrati separatamente ai partecipanti

I partecipanti dopo quindi aver effettuato alcuni passaggi utili alla riuscita dell'esperimento per la rilevazione dei biomarcatori, come ad esempio l'aver indossato la strumentazione o aver effettuato la corretta calibrazione, quando saranno pronti faranno partire il video. Il video come detto è molto breve, ho quindi scelto di riprodurlo tramite una libreria per esperimenti clinici chiamato Psychopy, che approfondirò in seguito, che permette una visione molto immersiva riproducendo il video a schermo intero e lasciando abilitato solo il mouse per una eventuale interruzione prematura per qualsiasi problema. Durante la visione del video verranno rilevati e registrati tutti i dati dei biomarcatori sui quali si andranno poi a trarre le conclusioni. Terminato il video l'utente verrà portato nel browser predefinito, nei vari test è stato utilizzato Google Chrome, in una pagina di Google Moduli sulla quale dovrà rispondere a dei questionari mirati a capire se e come l'utente si sente influenzato da ciò che ha visto. Ovviamente le sue sensazioni saranno messe a confronto con ciò che viene rilevato dalla strumentazione, così da capire anche la distanza tra l'influenza effettiva e quella percepita dal consumatore. Per il secondo esperimento mi è stato invece fornito un video molto più lungo, si tratta infatti di una puntata di una nota serie TV americana chiamata Modern Family, che come ovviamente un qualunque show trasmesso sulle reti

televisive è intervallata da alcune pubblicità. Tra le normali pubblicità si è scelto di inserire gli spot di tre diverse birre: Corona, Peroni e Ceres. Sono state scelte, come mi ha spiegato chi ha ideato l'esperimento, perché anche pubblicizzando lo stesso prodotto, hanno deciso di adottare scelte narrative diverse: chi adottando lo humor e la satira politica (Ceres), chi scegliendo di immergere il telespettatore in situazioni del quotidiano (Peroni), chi ambientando lo spot in un luogo lontano dalla vita comune ma collegato al brand della birra (un luogo tropicale per la Corona).

Anche per questo esperimento è stato progettato in primis un passaggio di preparazione e calibrazione degli strumenti, su tutti l'Eye-Tracker, e alla fine del video un questionario sempre fornito su Google Moduli. Le domande saranno in questo caso legate al tipo di birra che si vuole acquistare dopo la visione e un giudizio generale sugli spot. Come si può immaginare siamo di fronte ad un video sicuramente più lungo e impegnativo, 15-20 minuti di tempo richiesto, per questo ed alcune difficoltà tecniche dovute alla piattaforma scelta che sarà meglio descritta successivamente, ho deciso di creare un lettore multimediale ad hoc per l'esperimento. Non ho inserito la possibilità di caricare diversi video o altre funzionalità per permettere all'utente di concentrarsi unicamente sull'esperimento. Ho deciso di lasciare solamente i pulsanti di play, pausa e lo slider del volume, oltre alla possibilità di chiudere il player multimediale. La scelta è stata pensata mettendo in conto che durante l'esperimento potrebbe presentarsi una necessità di qualsiasi tipo e visto che il partecipante non avrà idea dello scopo dello studio inizialmente, potrebbe voler bloccare la visione temporaneamente (per esempio per non perdere il filo dell'episodio), potrà così farlo senza però dover ricominciare dall'inizio ogni volta che accade. Questi due esperimenti erano già stati elaborati prima del mio arrivo e quindi mi è stata fornita una struttura decisa precedentemente, che ho dovuto applicare con dei piccoli accorgimenti tecnologici ove necessario. I prossimi esperimenti presentati, invece, sono stati pensati e creati dal momento in cui ero già nel team, ne ho potuto quindi osservare la creazione e lo sviluppo passo dopo passo.

2.1.3 Esperimenti sul design di siti web

Come detto in precedenza il web design è un altro dei campi di applicazione del neuromarketing, viene infatti utilizzato per produrre siti web più accattivanti e facilmente utilizzabili da un qualunque utente, creando per lui un'esperienza positiva e piacevole che lo invogli non solo a comprare un prodotto, ma magari ad utilizzare nuovamente la stessa pagina per usufruire di quello specifico servizio. Entrambi gli esperimenti si basano sul far navigare l'utente in due specifici siti, facendo eseguire task analoghe a due gruppi diversi di partecipanti, ognuno su un sito differente, così da poter confrontare in maniera oggettiva i due esempi scelti; gli esperimenti però si concentrano su aspetti differenti interni ai siti. Nello specifico il primo esperimento si concentra sull'importanza dello storytelling attraverso il web e del diverso modo in cui può essere fatto su una piattaforma multimediale come un sito web. In particolare, la differenza può essere schematizzata in "emotional" e "non emotional" storytelling come trattato e approfondito nello studio [3]. Si tratta di un diverso approccio al raccontare la storia, rimanendo più asettici e legati ai fatti oggettivi, oppure provando ad approfondire altri aspetti raccontando una storia che vada a toccare le emozioni del lettore, magari a discapito di una completa oggettività.

È ovvio che questo effetto con l'utilizzo di un sito web può essere ottenuto utilizzando i contenuti multimediali oltre che con le semplici parole, permettendo di avere un effetto sicuramente amplificato sul lettore utilizzando anche immagini, video, musica o anche solo colori predominanti sul sito, permettendo un maggiore impatto di quello che può essere chiamato "persuasive storytelling". In questo esperimento si vanno a contrapporre due siti per quanto riguarda l'argomento della Sostenibilità che entrambi propongono e che dovrebbe essere un motivo in più per scegliere il loro prodotto. Stiamo parlando di caffè e i due produttori sono Lavazza, che opta per un approccio al racconto più breve e schematico, contro Nespresso, che invece decide di scrivere un racconto più lungo e coinvolgente che cerca di spingere più sulla parte emotiva. Lo studio vuole quindi, una volta collegati gli strumenti del caso,

andare ad analizzare come il lettore reagisca consciamente, tramite un questionario a cui viene sottoposto al partecipante al termine dell'esperimento, e inconsciamente, tramite le stesse tecniche utilizzate anche dagli esperimenti precedenti, vedendo come varia la risposta del lettore a seconda del tipo di stimolo, emotivo o non emotivo, al quale viene sottoposto. Il secondo esperimento pone sempre a confronto due siti web, ma questa volta si tratta di pagine web dedicate al turismo in due diversi paesi: la Spagna e la Norvegia. Questa volta lo studio si concentra sulla funzionalità e l'intuitività dei due siti web, in entrambi l'utente dovrà infatti navigare nella pagina e cercare di prenotare un'esperienza a contatto con la natura. Ovviamente i due siti web sono sviluppati in maniera diversa ed hanno quindi un diverso layout della pagina nonché aspetto estetico e funzionalità diverse, questo provocherà sicuramente una differenza nel come il partecipante si approccerà al sito. Lo studio andrà quindi ad analizzare sempre con lo stesso metodo, conscio tramite questionario e inconscio tramite rilevazione di biomarcatori, l'intuitività e la facilità con il quale l'utente compirà il task, nonché le emozioni provocate dall'azione, così da poter comprendere se una pagina rispetto all'altra abbia lasciato un'impronta migliore sul potenziale cliente, che sarà quindi più invogliato in futuro ad utilizzarla. Per entrambi gli esperimenti, aiutandomi con delle librerie esterne ho ricreato un'interfaccia simile a quella di un normale browser ma con molte meno funzioni. Questo permette una maggiore facilità di utilizzo (l'utente può infatti accedere alla pagina richiesta tramite un bottone e non deve navigare sul web alla sua ricerca) permettendo al partecipante di concentrarsi solo sull'esperimento. Inoltre, con questo browser custom all'interno dell'applicazione ho potuto inserire un pannello laterale in cui l'utente può leggere le istruzioni delle azioni che dovrà compiere. Questa scelta è stata esplicitata per permettere al partecipante di non essere in difficoltà a dover chiedere nuovamente indicazioni su cosa fare e quindi per metterlo totalmente a suo agio e non creare stress o tensione che avrebbero potuto influire sulle misurazioni. Mentre gli esperimenti riguardanti la birra erano già stati progettati ed erano stati testati su un'altra piattaforma per ottenere dei risultati puramente indicativi, questi due protocolli sono stati

pensati e definiti durante il mio lavoro di tesi, quindi ho potuto seguirne la nascita e la definizione, con altrettanto interesse seguirò la fase di testing che avverà invece in laboratorio una volta che la parte riguardante il machine learning sarà ultimata e l'emergenza Covid permetterà una maggiore elasticità riguardo i protocolli, poiché la salute dei partecipanti ha ovviamente la precedenza.

2.2 Neuroriabilitazione

2.2.1 Neuroriabilitazione e scopo dello studio

Il secondo gruppo che collaborava al progetto e che cercava di applicare gli stessi principi di studio delle emozioni umane era il team che si occupava della neuroriabilitazione. Una collega laureanda in psicologia presso l'Università degli Studi di Torino ha infatti elaborato un esperimento neuroriabilitativo che potrà aiutare persone che in seguito a trauma cranico hanno perso o ridotto la capacità di riconoscere le proprie emozioni. Andando ad osservare in maniera più generale l'argomento, possiamo definire la neuroriabilitazione come "una branca della medicina specializzata nel curare e riabilitare il corpo in seguito a una lesione del sistema nervoso e a minimizzare o compensare eventuali alterazioni funzionali che ne derivano" [4]. È una disciplina relativamente nuova, ma che sta trovando grande successo e importanza visto il gran numero di disagi, e la loro varietà, che un problema al sistema nervoso può comportare. Questi danni possono essere causati da un gran numero di motivazioni, nel nostro studio abbiamo raggruppato indicativamente la causa del disagio a tre categorie: danno vascolare, danno oncologico e trauma cranico. Le problematiche dovute all'insorgenza di malattie che provocano una lesione al sistema nervoso o più semplicemente di un trauma cranico possono appartenere a più di un dominio: possono essere di natura linguistica, visiva, motoria, percettiva, oppure peggiorare anche pesantemente la capacità dell'individuo di ragionare lucidamente e, appunto, di riconoscere in maniera chiara le emozioni proprie o altrui. Questi deficit possono avere conseguenze molto importanti a livello di vita quotidiana soprattutto per quanto riguarda

il rapporto con chi si ha intorno, portando il paziente a non avere una vita normale in particolare dal punto di vista interpersonale, tutto questo, come si può immaginare, può essere fonte di un notevole disagio nella vita al di fuori della struttura clinica. In particolare, è stato importante osservare come nel campo della riabilitazione neurologica delle emozioni, i training più sviluppati sono quelli relativi alla riabilitazione del riconoscimento delle emozioni altrui, quindi di pazienti che faticano ad empatizzare con chi gli sta intorno perché osservandoli o sentendo il loro tono di voce non riescono più a comprendere quale sia lo stato d'animo di chi si trovano davanti. Al contrario si è evidenziata una carenza di training destinati alla riabilitazione del riconoscimento delle emozioni che è il paziente stesso, vittima del trauma o della malattia, a provare. Lo scopo di questo studio sarebbe quindi proprio quello di colmare questa carenza cercando di aiutare e facilitare la riabilitazione di pazienti che dimostrano deficit in questo senso; poter velocizzare il percorso di persone che hanno tali disagi diventa di fondamentale importanza, poiché per problemi neurologici legati ad esempio a malattie come il Parkinson, i pazienti possono dover partecipare a sedute neuroriabilitative anche per tutta la vita. Concentrandosi sul nostro specifico task si è osservato che è presente una maggior difficoltà nel riconoscere alcune emozioni rispetto ad altre, in particolare quelle che sono universalmente classificate come negative, quali tristezza, paura e disgusto, sono inficiate maggiormente rispetto alle emozioni classificate come positive, quali ad esempio la felicità, l'eccitazione o anche solo la rilassatezza. Altro fattore importante che è stato identificato è la presenza di una diminuzione dell'intensità emotiva riguardo alle emozioni definite negative, in maniera maggiore rispetto a quelle positive. Queste alterazioni dell'intensità emotiva saranno identificate e analizzate grazie alla rilevazione di biomarcatori, quali la conduttanza cutanea, la frequenza del battito cardiaco e la dimensione della pupilla che fungono da feedback biologici. Si vuole quindi riabilitare nel paziente la percezione delle proprie emozioni, con particolare interesse nel capire se alcune emozioni provochino più problemi ad essere individuate rispetto ad altre, aspettandoci

come detto una maggiore difficoltà nell'individuazione delle emozioni etichettate come negative.

2.2.2 L'esperimento

L'obiettivo dell'esperimento sarà quindi quello di identificare l'emozione provata dall'utente e l'intensità di quest'ultima, basandosi principalmente sull'analisi delle espressioni facciali del volto dell'utente che ne fa uso (facial coding), in unione con gli altri biomarcatori citati precedentemente (conduttanza cutanea, battito cardiaco e dimensione pupillare) che faranno da supporto per le scelte prese dall'algoritmo alla base della classificazione. L'obiettivo finale del progetto, quindi, è quello di stabilire se, attraverso un programma di training basato sull'uso dell'applicazione, il paziente può migliorare il riconoscimento delle sue stesse emozioni. Il paziente potrà infatti utilizzare una versione dell'applicazione in clinica, lavorando con tutti gli strumenti sopracitati, mentre potrà lavorare quotidianamente anche a casa con una versione web dell'applicazione che però utilizzerà solamente il facial coding, ma permetterà comunque di continuare il percorso. L'esperimento è strutturato in maniera semplice, dopo aver preparato la strumentazione al paziente viene mostrata una foto presa casualmente all'interno del database IAPS (International Affective Picture System), sviluppato da Lang, Bradley Cuthbert. Si tratta di un database composto appositamente per lo studio di emozioni e attenzione, poiché al suo interno contiene da immagini riferite ad oggetti e situazioni quotidiane, fino a bellissimi paesaggi per arrivare a scene molto crude o a sfondo sessuale molto esplicito. Risulta essere un dataset perfetto per scatenare la reazione emotiva del paziente e poterla analizzare più chiaramente. Dopo aver mostrato l'immagine, l'utente dovrà scegliere una delle label che gli verranno mostrate sulle quali è scritta una delle emozioni che il paziente può aver provato. Una volta scelta l'emozione, si passa alla foto successiva, fermandosi per questo specifico protocollo a 20 fotografie. Si potrà così confrontare la risposta dell'utente con quello che è il software a rilevare, così da avere un'indicazione di come il paziente reagisce e crede di reagire ad una data emozione. I pazienti saranno divisi in due gruppi,

uno sperimentale e uno di controllo. Quello sperimentale sarà sottoposto a questo protocollo riabilitativo, mentre quello di controllo riceverà il trattamento classico. Il risultato atteso è il verificarsi di un miglioramento statisticamente significativo della capacità di riconoscimento delle emozioni provate nel gruppo sperimentale rispetto al gruppo di controllo. I gruppi verranno confrontati sia in fase preliminare, che alla fine del percorso riabilitativo. Come detto sarà inoltre posta una particolare attenzione a rilevare se nel riconoscimento delle proprie emozioni si verifica una difficoltà a riconoscere alcune emozioni rispetto ad altre siccome, allo stato dell'arte, è riscontrata una maggiore difficoltà a riconoscere le emozioni etichettate come negative. Questo verrà verificato analizzando le risposte dei pazienti al questionario, che gli verrà sottoposto al termine dell'esperimento, che unito ai dati rilevati sui biomarcatori potranno fornire un quadro migliore sulla sua percezione delle emozioni a livello conscio e inconscio. In aggiunta, lo studio si propone di valutare se il punteggio ai test Toronto Alexithymia Scale e GERT-S effettuati prima del training, dopo il training e al momento del follow up migliora in modo statisticamente significativo. In caso positivo, questo rappresenterebbe un miglioramento a livello della sfera emozionale del paziente, indicando un successo del percorso neuroriabilitativo.

2.3 Gli strumenti e le tecniche

Ciò che unisce questi due ambiti è senza dubbio lo studio dell'effetto sull'uomo degli stimoli proposti nei vari esperimenti, per andare a ricercare come utilizzare le reazioni che la persona non percepisce e quindi raggiungere un obiettivo, che sia questo la guarigione di un paziente o il fornire un prodotto nella maniera migliore possibile. Come ampiamente detto per fare tutto ciò si fa uso di vari strumenti che vanno a monitorare: • l'attività cerebrale, permettendo quindi di capire che parte del cervello viene attivata durante azioni specifiche o quando la persona riceve determinati stimoli, queste tecniche possono essere associate alla branca denominata brain imaging; • i

segnali psicofisiologici, che non sono altro che le risposte psichiche e fisiologiche del corpo alle emozioni, che come abbiamo detto risultano avere un ruolo centrale in tutto lo studio, queste risposte sono state denominate negli anni Settanta biofeedback e il loro studio si basa sulla concezione che l'organismo interagendo con l'ambiente riceva da questo un gran numero di informazioni, che innescano inconsciamente nell'individuo dei meccanismi di autoregolazione provocando reazioni che possono essere analizzate. Andremo quindi a vedere brevemente i vari biomarcatori che si è deciso di prendere in considerazione, per quanto riguarda il brain imaging esistono una grande varietà di tecniche come la fMRI (Risonanza magnetica funzionale) o la PET (Tomografia ad emissione di positroni), il nostro studio però concentrandosi sulle emozioni ha dato la precedenza alle emozioni e alle risposte psicofisiologiche dell'organismo, per questo l'unica tecnica che va ad osservare direttamente l'attività cerebrale che si è deciso sarà utilizzata in una fase successiva dell'esperimento sarà l'Elettroencefalografia o EEG. Il Facial coding non sarà trattato in questa sezione poiché se ne parlerà largamente e in maniera più tecnica nei capitoli successivi.

2.3.1 Elettroencefalografia o EEG

L'EEG è uno strumento di monitoraggio che permette di registrare l'attività elettrica del cervello, questa funzione è possibile grazie al posizionamento di una serie di elettrodi, da 10 a 20, in alcune specifiche posizioni della testa che sono definite da uno standard chiamato "sistema internazionale 10-20", con l'utilizzo di gel conduttivo. Questi elettrodi sono collegati ad un convertitore analogico-digitale, che è a sua volta collegato ad un dispositivo che permette la registrazione delle onde rilevate. I vari dispositivi devono essere accuratamente schermati per evitare interferenze che potrebbero falsare le misurazioni dello studio. Le posizioni in cui vengono applicati gli elettrodi sono associate ad una particolare zona del cervello, in questo modo se durante la presentazione di uno stimolo, prettamente visivo e uditivo come nel nostro caso, si rileva una particolare attività cerebrale in una data zona, questo dato può essere associato ad un diverso processo cognitivo a seconda

della parte del cervello attivata. Queste attività possono essere associate ad esempio all'utilizzo della memoria o al tentativo di prendere una decisione, molto importante nei nostri esperimenti legati al neuromarketing, ma anche all'insorgenza di emozioni. Per questo motivo questa tecnica è molto utilizzata per la visione di spot o contenuti multimediali in generale; osservando in tempo reale quali zone del cervello vengono attivate, gli studiosi possono infatti comprendere quando il partecipante attiva i processi di memorizzazione. Stimolando questi processi si può permettere l'associazione, ossia permettere all'oggetto in questione di essere richiamato alla mente, anche molto velocemente se la relazione nella memoria è molto forte. Stimolando questi processi mnemonici posso quindi favorire il partecipante a pensare al prodotto, spesso utilizzando spot che richiedono una grande attività cerebrale da parte dello spettatore. Per capire che tipo di processi cognitivi si stanno attivando nel partecipante dobbiamo allora definire i vari tipi di risposta che possono essere rilevati dall'EEG, sto parlando delle onde cerebrali, che possono avere frequenza diversa e di conseguenza si dividono in vari tipi: 1. Onde Alfa, con frequenza dagli 8 ai 13 hertz, tipiche di stati di rilassamento e calma. Quando troviamo un picco di tali onde vuol dire che non si è in presenza di grandi sforzi cognitivi e quindi la concentrazione è ostacolata, al contrario in situazioni dove l'individuo è in ansia o sotto stress, se ne rileva un livello molto basso. 2. Onde Beta, con frequenza dai 13 ai 30 hertz, sono presenti in caso di una forte attività neuronale in cui l'attenzione è massima. Un livello di onde Beta troppo basso fa riferimento a stati di calma eccessiva, al contrario una loro presenza evidente indica come l'individuo in quel momento possegga una giusta dose di concentrazione. 3. Onde Gamma, con frequenza che dai 30 ai 42 hertz, sono quelle di studio più recente e fanno la loro comparsa durante processi di forte elaborazione di tipo cognitivo. I picchi di queste onde possono essere correlati sia a stati di felicità che alla fase del sonno REM, al contrario una bassa attività di onde Gamma si registra in persone con problemi mentali o di apprendimento. 4. Onde Delta, con frequenza da 0.1 a 4 hertz, sono presenti nella fase di sonno profondo e non REM, senza quindi la presenza di sogni. Alti picchi di questo tipo di

onde possono rivelare delle lesioni cerebrali o problemi di apprendimento, al contrario la presenza di picchi bassi indicherebbe una probabile carenza di sonno. 5. Onde Theta, con frequenza che 4 ai 8 hertz, sono legate alla capacità di immaginazione e riflessione, oltre che al sonno nell'individuo adulto, mentre sono predominanti nei neonati.

Possiamo quindi capire perché questa tecnica stia trovando spazio in vari ambiti applicativi, in quanto permette di comprendere l'efficacia comunicativa che il messaggio proposto possiede o anche l'emozione che lo stimolo sottoposto scatena nel paziente in maniera molto profonda.

2.3.2 Eye-tracking

Si tratta di una tecnica molto utilizzata nel campo soprattutto del neuromarketing, permette di comprendere in quali zone dello stimolo e per quanto tempo l'occhio del partecipante si sofferma durante l'osservazione. I dispositivi utilizzati per registrare i movimenti dell'occhio si chiamano Eye-tracker e ne esistono vari tipi, alcuni sono dei semplici dispositivi da applicare sopra allo schermo e sono raccomandati per studi compiuti in laboratorio (come il Tobi Eye-tracker da noi utilizzato), altri sono comparabili ad occhiali dove sono le lenti a compiere il lavoro e permettono un utilizzo più semplice nel mondo reale. La tecnologia non è assolutamente invasiva, consiste infatti nella proiezione di raggi infrarossi verso l'occhio dell'individuo mentre gli viene sottoposto lo stimolo di interesse. Sono scelti i raggi infrarossi per la loro precisione e per il fatto che non essendo percepiti dall'occhio umano, non disturbano il partecipante nel corso dello studio. La luce viene riflessa sia sulla pupilla che sulla cornea, la telecamera a infrarossi riesce a catturare questi riflessi costruendo un vettore che arriva fino allo schermo così da poter ricavare il movimento della pupilla e il punto dello schermo che viene osservato, basta quindi registrare le zone istante per istante che vengono osservate e creare una mappa di queste [5].

Nel processo di misurazione attraverso l'eye-tracker, si vanno a misurare i punti di osservazione, campioni grezzi corrispondenti a circa a 1/60 di secondo (lo strumento registra circa 60 segnali al secondo), e i punti di fissazione, che

sono identificati da una serie di punti di osservazione molto vicini nello spazio e mediamente durano sui 200 millisecondi. Queste due sono le unità base di misurazione di un eye-tracker, ma grazie ad esse si può arrivare a visualizzare delle metriche più utili per la comprensione dei risultati. Le heat map, ad esempio, sono delle rappresentazioni visive dei punti di osservazione, proprio come le classiche mappe di calore i punti in cui si raggruppano un maggior numero di punti vengono caratterizzate da un colore sempre più tendente al rosso, al contrario le zone su cui l'occhio staziona in maniera minormente passano dal giallo al verde, fino a zone prive di colore che non sono state raggiunte dall'occhio. Le heat map possono essere statiche, comprendendo i punti di osservazione all'interno di un intervallo di tempo ben definito, o dinamiche se cambiano real-time con il progredire dello stimolo. Sono molto utili per individuare le zone di maggiore interesse per l'osservatore e comprendere sia se ci siano particolari elementi che attirano l'utilizzatore dello strumento, sia se ci siano delle zone specifiche che si è più propensi ad osservare e che quindi devono avere la precedenza se si vuole trasmettere un messaggio. Altre metriche di interesse sono ad esempio le aree di interesse (aree dello stimolo alle quali vengono associate informazioni, come ad esempio l'ordine con cui vengono notate o per quanto tempo), il tempo per la prima fissazione (quanto impiega il soggetto a fissare una data aerea per la prima volta), le sequenze di fissazione (che individuano il percorso dell'occhio nelle varie fissazioni dalla presentazione dello stimolo), e altre di minore importanza.

2.3.3 Risposta galvanica della pelle o GSR

La risposta galvanica della pelle o GSR (Galvanic Skin Response) è anche detta attività elettrodermica, è una metrica che permette di indicare variazioni nella conduttanza elettrica della pelle dell'individuo in seguito ad un aumento o diminuzione della sudorazione. In risposta ad uno stimolo, per necessità di termoregolazione o semplicemente per l'insorgere di emozioni principalmente negative (soprattutto ansia o stress), il sistema nervoso simpatico o parasimpatico attiva le ghiandole sudoripare. L'attività di queste

ghiandole provoca un aumento di salinità della pelle e di conseguenza una variazione di conduttanza del derma.

Queste osservazioni possono essere molto importanti poiché si è visto come il sistema nervoso simpatico svolga un ruolo molto importante nel processo decisionale dell'individuo. Allo stesso tempo può essere un importante indicatore da affiancare ad altre misurazioni, come controprova, per l'individuazione di una data emozione o comunque evidenziare uno stato di alto coinvolgimento emozionale.

Per la misurazione abbiamo utilizzato (mindfield E-sense skin response) uno strumento composto da due elettrodi che vengono applicati alle dita indice e medio della mano non dominante, non quella che verrà quindi utilizzata per gli esperimenti, questi elettrodi collegati ad un sensore rilevano i dati che possono essere rappresentati o classicamente su di un piano cartesiano, oppure direttamente inseriti negli algoritmi predittivi utilizzati. Si è infine pensato di passare a strumenti ancora meno invasivi, come orologi o bracciali che potrebbero essere utilizzati quindi anche per registrare il battito cardiaco essendo questo un altro fattore che può essere facilmente inserito nel modello predittivo, potrebbero perciò rivelarsi molto utili in caso di future task che richiedano l'uso di entrambe le mani.

Chapter 3

La creazione dell'applicazione

In questo capitolo affronteremo le varie fasi che si sono succedute per andare a costruire un ambiente applicativo in cui poter permettere lo svolgersi in maniera ottimale degli esperimenti descritti nel capitolo precedente. Anche qui i due gruppi di ricerca precedentemente descritti hanno evidenziato necessità e preferenze diverse che insieme abbiamo cercato di accontentare nei limiti di tempo e capacità a disposizione. In particolare, per quanto riguarda la Neuroriabilitazione si è evidenziata la necessità per il paziente di effettuare le sedute in una sede ben definita, in cui poter utilizzare tutti gli strumenti precedentemente descritti, per registrare i vari biomarcatori, ma per un processo riabilitativo più efficace ed efficiente si è ritenuto utile fornire al paziente anche la possibilità di continuare il percorso comodamente da casa. Si è quindi deciso di creare due diverse versioni dell'applicazione, una desktop application e una web application. In questo modo il paziente potrà utilizzare la desktop application nel centro riabilitativo, nel nostro caso il Centro Puzzle (centro per traumi cranio-encefalici e gravi cerebrolesioni acquisite) di Torino, dove sarà installata e disposta per poter utilizzare tutte le attrezzature legate ai biomarcatori. Una versione con meno funzionalità, la

web-application, potrà essere invece utilizzata ovunque, basterà avere un dispositivo con collegamento a internet ed una fotocamera, quindi un qualunque smartphone. Su questa versione dell'applicazione, è implementato lo stesso esperimento della prima, con la differenza che l'unica metrica valutata per andare a evidenziare l'emozione provata dal paziente è l'algoritmo di facial coding che fa uso del machine learning per effettuare emotion recognition. La parte di facial coding sarà quindi la stessa in entrambi i progetti, il suo sviluppo è stato effettuato in parallelo con il collega che si occupava della web application così da ottenere risultati consistenti in entrambe le applicazioni e non avere discrepanze nei risultati nella fase di sperimentazione, che avrebbero potuto creare problemi al personale. Al contrario, per quanto riguarda gli esperimenti di Neuromarketing, si è scelto in accordo con il gruppo di ricerca, di sviluppare solamente un applicazione desktop, vi è infatti la necessità di raccogliere unicamente i dati nel laboratorio (Nome?) situato a Caserta, nel quale è presente tutta l'attrezzatura necessaria per poter registrare al meglio tutti i dati necessari. Personalmente mi sono dovuto occupare dell'applicazione desktop, che si era pensato inizialmente essere una per entrambi i progetti, ma con il crescere degli esperimenti e la possibilità che in futuro due team diversi lavorino in parallelo sui due progetti senza forzatamente avere un punto di contatto, si è scelto di separare le due applicazioni. Questa separazione per il momento è in realtà solo teorica, in quanto la struttura delle applicazioni è esattamente la stessa se non per la differenziazione negli esperimenti che esse presentano, che possono però essere guardati come dei moduli indipendenti che vengono avviati all'interno di questo ambiente. La scelta ha già però cominciato a mostrare la sua utilità, poiché una volta passati a dettagli minori, ma che comunque richiedono tutta l'attenzione e la precisione possibile, ci si è resi conto come le informazioni registrate per pazienti clinici rispetto a quelle salvate per dei partecipanti ad un esperimento sono soggette a regolamentazioni diverse riguardo la privacy e questa divisione permetterà una gestione differente dei dati (utilizzo di un codice ID invece che nome e cognome, aggiunta di una patologia per quanto riguarda i pazienti, etc.) a seconda della destinazione di utilizzo. Era proprio questo il

mio scopo, trovandomi infatti all'inizio di un progetto che continuerà anche alla fine del mio lavoro di tesi, dovevo gettare le basi per creare un ambiente flessibile che potesse dare la possibilità di inserire in maniera semplice e senza grandi sforzi nuovi esperimenti che nei prossimi mesi o anni vorranno essere sottoposti ai partecipanti, per investigare magari nuovi biomarcatori o anche solo provare la stessa tipologia di test, ma da punti di vista diversi e poter confrontare i risultati. Per questo tutto lo sviluppo dell'applicazione e le scelte che ho compiuto in merito ad essa sono state guidate da questo spirito, non solo cercare di colmare nella maniera più rapida possibile le esigenze che venivano sollevate da entrambi i team di ricerca, ma anzi cercare di anticipare i problemi che potranno presentarsi e scegliere sempre una soluzione che non tarpi le ali ai futuri colleghi e che gli permetta di cambiare rotta nel modo più indolore e veloce possibile. Non tutte le scelte sono quindi ottimali con ciò che si voleva produrre, ma si è cercato di raggiungere un buon compromesso tra efficienza, flessibilità e rapidità di sviluppo così da poter passare all'ultima fase del mio lavoro, ossia la produzione di un algoritmo efficace di facial coding. Questa fase mi ha quindi aiutato a capire la difficoltà di lavorare in gruppi eterogenei e con necessità diverse, nei quali la comunicazione continua, ma soprattutto la capacità di anticipare le esigenze dei colleghi e di metterli nella situazione di poter esprimere al meglio il proprio lavoro, permettono di risparmiare molto tempo in fasi successive del lavoro. Tutto ciò è stato evidente nella fase di transizione, in cui vi è il passaggio di testimone ai laureandi successivi che dovranno portare ulteriormente avanti il lavoro, che avendo visioni diverse e idee diverse, stanno cambiando alcune caratteristiche del progetto e degli esperimenti associati. In tutto questo non si sono trovati però legati in alcun modo dalla struttura in cui sono entrati, ma hanno potuto muoversi liberamente e con minimo sforzo. Posso quindi evidenziare l'importanza per un programmatore, quando si tratta di sviluppo di progetti a lungo termine, di saper capire quali strade potranno essere percorse in futuro, anche se al momento non sembrano percorribili, rinunciando anche ad una piccola quantità di tempo ed efficienza nell'immediato, per poterne far

risparmiare molto al team in caso di un futuro potenziale cambio alle caratteristiche o necessità strutturali del progetto. Andiamo quindi nei paragrafi successivi ad analizzare le varie scelte effettuate e le motivazioni dietro di esse, nonché le difficoltà che si sono presentate nel corso dello sviluppo.

3.1 La scelta del linguaggio

Dopo aver deciso la struttura del software, avendo optato nel mio caso per un applicazione desktop standalone, senza quindi la necessità di un server che elabori i dati forniti dagli esperimenti, si è dovuto passare alla scelta successiva, il linguaggio di programmazione. La scelta del linguaggio è un passaggio molto importante all'interno di un progetto, poiché da esso dipenderanno molte delle scelte riguardo la compatibilità e le librerie esterne (o i moduli) di cui si potrà usufruire per semplificare enormemente il lavoro. Come suggerito da molti esperti del settore [6] quando si vanno ad operare delle scelte è importante porsi le domande corrette, così da capire le necessità che il progetto richiede e cercare di andare a coprire tutti i bisogni al costo (sia esso tempo, competenze o soldi) minore possibile, senza che però la qualità non ne risenta in maniera evidente. Non essendo in ambito aziendale (dove spesso vi sono degli standard di compatibilità da rispettare tra i vari progetti) e dovendo creare qualcosa di nuovo e completamente indipendente, vi erano infinite possibilità per la creazione di questa applicazione, la sfida è stata quindi andare ad individuare tutti i possibili bisogni che sono emersi e in futuro continueranno ad emergere e mettersi nella situazione migliore possibile rispetto ad essi, perseguendo quella logica di sviluppo orientato al futuro di cui si è parlato precedentemente. Solitamente si parte dal vedere i vincoli del progetto noti a priori, nel nostro caso ciò che era chiaro fin da subito è che i vari esperimenti dovessero poter comunicare con la strumentazione elencata nel paragrafo 2.3 che era già in nostro possesso. Ogni strumento ha infatti un proprio Software Development Kit o SDK (o perlomeno un API) tramite il quale il produttore dell'oggetto facilita e spesso promuove lo sviluppo di applicazioni e soluzioni che coinvolgano l'utilizzo del proprio prodotto. Questo

modo di agire aumenta l'appetibilità dello strumento, poiché più l'SDK è semplice e ben fornito di strumenti e funzioni utili con cui comunicare con il dispositivo maggiore è la sua facilità di utilizzo e di conseguenza l'interesse di chi intende avvalersi delle sue funzioni poiché solitamente permette di risparmiare una grande quantità di tempo. Per questo il primo passaggio è stato ricavare tutte le informazioni sugli SDK collegati ai dispositivi che si intendevano utilizzare all'interno della ricerca e comprendere i loro vincoli e la loro compatibilità, non solo per quanto riguarda il linguaggio, ma anche per esempio le versioni dei linguaggi con cui questi SDK erano implementati, cosa che spesso può creare una serie di problemi in caso di mancati aggiornamenti del software che ne minano in maniera critica la funzionalità. Un altro aspetto molto importante da considerare era senza dubbio l'ambiente di deploy, quindi le risorse delle workstation di destinazione su cui il software verrà installato o fatto girare, nel nostro caso a seconda dello studio il software deve poter funzionare senza intoppi o cali evidenti di prestazioni anche su portatili non troppo potenti, quindi le risorse necessarie non dovranno essere eccessive. Per quanto riguarda il sistema operativo, si è deciso di dare priorità al funzionamento in ambiente Windows 10, senza escludere l'interoperabilità su di altri sistemi, ma lasciando un eventuale studio di compatibilità ad una fase successiva, non avendo postazioni specifiche che montano altri sistemi operativi. Finite le limitazioni dovute alle necessità hardware, non vi erano all'inizio grandi indizi su quali dovessero essere le problematiche lato software che sarebbero potute sorgere, si è dovuto ipotizzare lo sviluppo del progetto e lasciare aperte più strade percorribili possibili. La più grande fonte di varietà era senz'altro la struttura degli esperimenti che poteva cambiare nel corso del tempo per la modifica di questi ultimi o per l'inserimento di nuovi test teorizzati dai futuri membri del gruppo di ricerca. Questo implicava che servisse un ecosistema molto vario e ben fornito, che permettesse di trovare una soluzione pratica e abbastanza immediata ai problemi più disparati, in quanto lo scopo fin dall'inizio è stato di non perdere troppo tempo sui dettagli implementativi nelle fasi successive del progetto per concentrarsi sui risultati e sul trattamento dei dati ottenuti una volta che gli esperimenti sarebbero stati

avviati. È quindi evidente la necessità di un ecosistema con queste caratteristiche, in continuo sviluppo e dinamico, nonché di facile reperibilità, poiché l'integrazione di moduli di terze parti avrebbe semplificato notevolmente il lavoro permettendo un risparmio di tempo notevole, nonché probabilmente il raggiungimento di una qualità maggiore. Un aspetto secondario che si è dovuto prendere in considerazione è quello dell'esperienza dei programmatori per quanto riguarda i vari linguaggi di programmazione tra i quali si poteva optare. Non vi erano infatti delle necessità che richiedessero una conoscenza approfondita del linguaggio, in quanto non si puntava a livelli di performance e ottimizzazione ottimali, ma di sicuro serviva una certa familiarità del team con il linguaggio per poter partire in fretta a costruire il sistema, magari approfondendone la conoscenza con il progredire dell'applicazione.

3.1.1 Python

Analizzati i quesiti posti precedentemente la risposta più logica da dare ci è sembrata l'utilizzo di Python. Python [7] [8] è un linguaggio di programmazione di alto livello introdotto a inizio anni 90 dal programmatore Guido Van Rossum, è un linguaggio pseudointerpretato (un interprete gira su di una macchina virtuale ed esegue il codice una linea alla volta, convertendolo poi in linguaggio macchina, è quindi indipendente dalla piattaforma di utilizzo) e general-purpose, destinato perciò alla programmazione in un largo ambito di domini di applicazione, anche se negli ultimi anni ha trovato un larghissimo impiego soprattutto in ambiti scientifici (come ad esempio il Machine Learning nello specifico). Nell'idea del suo programmatore vi era quello di renderlo molto semplice e di facile approccio, viene infatti spesso insegnato come primo linguaggio per la sua somiglianza allo pseudocodice e la sua struttura molto snella, che non lascia spazio ad ambiguità, come ad esempio la scelta di raggruppare i vari blocchi logici grazie all'indentazione delle varie righe di codice. Python supporta diversi paradigmi di programmazione come la programmazione procedurale, imperativa, funzionale e orientata agli oggetti, permettendo un utilizzo quanto mai variegato del linguaggio, che può essere utilizzato in maniera differente a seconda delle necessità e delle

capacità del programmatore. Si può infatti utilizzare per semplici comandi che condividono solo l'ordine di esecuzione (scripting), mentre invece si possono creare sistemi complessi e strettamente in collaborazione grazie alla programmazione a oggetti. Un'altra caratteristica centrale del linguaggio è la tipizzazione dinamica forte, che permette l'assegnazione di un valore di qualsiasi tipo alle variabili, il cui tipo verrà però poi controllato unicamente a runtime e non in fase di compilazione, permettendo una maggiore elasticità nella scrittura dei programmi, ma con il rischio dell'identificazione di un errore solo in fase di esecuzione, mentre prima sembrava funzionare tutto correttamente; si parla di forte invece perché il sistema dei tipi ha comunque una sua regolamentazione ben definita che non è sovvertibile. Infine, Python fa uso di un garbage collector che si basa sui concetti di generazione, ai quali gli oggetti appartengono, e di soglia o "threshold", che indica il numero massimo di oggetti in una data generazione, al raggiungimento del quale verrà attivato un "collection process" che si occuperà di liberare la memoria occupata da elementi che non sono più in uso da parte del programma. Questo sistema, che può essere modificato dal programmatore, permette di evitare errori nel caso di memoria non liberata o memoria liberata prematuramente, a volte sacrificando qualcosa per quanto riguarda le prestazioni. Python è completamente open-source, è gratis sia l'interprete che l'utilizzo del linguaggio stesso per i propri progetti, inoltre basa molto il proprio ecosistema sulla possibilità di modificare e ridistribuire il codice; parte della sua forza è infatti la sua grandissima standard library che va a coprire una grandissima quantità di strumenti per i task più disparati ai quali è affiancata una libreria di terze parti immensa chiamata Python Package Index o PyPi, utilizzata come sorgente standard di pacchetti e dipendenze da molti packet manager come pip, è in continua espansione e aggiornamento grazie a un ecosistema tra i più attivi sull'intero panorama dei linguaggi di sviluppo. Oltre alle caratteristiche elencate del linguaggio, la nostra scelta si è basata come detto sulle domande poste al capitolo precedente a molte delle quali Python è sembrata la risposta più evidente. Le SDK per tutti gli strumenti che mano mano verranno inseriti nel progetto in maniera incrementale, partendo dal facial

coding passando per l'Eye-tracker in coppia con il GSR, per arrivare infine alle EEG, sono tutte compatibili con Python, anzi per alcune di queste è addirittura il linguaggio in cui si trovano la maggior parte delle risorse. Per quanto riguarda il sistema operativo, essendo Python platform-indipendent, non c'è alcuna problematica legata all'ambiente in cui viene utilizzato, dandoci quindi la possibilità di lavorare inizialmente su Windows senza preoccuparci troppo della portabilità. Già molte parole si sono spese sull'ambiente di Python e ciò è sicuramente in accordo con ciò che cercavamo, in particolare come è sottolineato in "Python for Scientists and Engineers" [9] l'utilità del suo ecosistema per l'ambiente scientifico è molto evidente, come sottolineato dall'esistenza della community Scipy, una comunità di scenziati, ingegneri e ricercatori che utilizza, estende e favorisce l'utilizzo del linguaggio stesso nell'ambito della ricerca. Riguardo l'esperienza del team, è stata necessario un primo breve periodo di studio da parte mia vista la limitata conoscenza del linguaggio, utilizzato fino a quel momento principalmente come linguaggio di scripting e per qualche breve task legata al machine learning; molto utile è stato quindi il libro "Python Cookbook" [10] che mi ha permesso di comprendere al meglio gli altri paradigmi di programmazione, in particolare quella ad oggetti, e di approcciarmi ad argomenti utili per lo sviluppo di una applicazione come la creazione della User Interface.

3.1.2 Gli altri vantaggi di scegliere Python

La scelta del linguaggio è stata confermata non solo perché ha risposto ai nostri vincoli di base del progetto, essenziali anche solo per poter partire, ma anche perché ci ha convinto su alcune questioni secondarie, che però sono sempre parte di quell'approccio di sviluppo guardando al futuro del progetto, che si spera potrà facilitare i prossimi collaboratori che ne prenderanno parte. In primis arrivati a questo punto si era già deciso di sviluppare la parte di machine learning in Python, vista la sua totale predominanza in questo ambito e il suo largo utilizzo in tutta la letteratura del settore. Per quanto riguarda la web-app avendo una struttura client e server si poteva tranquillamente scegliere di avere un linguaggio sul client che avrebbe ospitato l'applicazione

web e uno totalmente diverso sul server che avrebbe avuto il compito di elaborare le risposte e comunicarle, non era questa quindi una grande limitazione. Per quanto riguarda la desktop application, come si è detto si è deciso per una struttura standalone, risultava quindi sicuramente più immediato utilizzare lo stesso linguaggio per tutte le sue parti, anche se come detto far comunicare due moduli in linguaggi differenti, non sarebbe sicuramente stato un vincolo limitante. Come detto la flessibilità era una delle caratteristiche che questo progetto poneva al centro delle sue intenzioni; trovandoci in un ambito tecnico-scientifico sarebbe stato possibile pensare all'utilizzo di tools più specifici come Matlab o Mathematica che permettono di gestire task matematiche in maniera compatta e immediata, ma come sottolinea [11] il grande vantaggio di Python è essere stato pensato e creato come un linguaggio general-purpose, che ha poi preso il sopravvento in molti ambiti scientifici grazie a moduli specifici che si sono evoluti nel corso degli anni. In particolare, si può citare la libreria NumPy che fornisce il NumPy array, una struttura dati elementare che può essere compresa virtualmente da tutte le librerie e può quindi essere utilizzata come oggetto comune per scambio di dati. Questo è un grande vantaggio perché gli array possono essere acceduti anche da moduli in linguaggi come C, C++ e Fortran, permettendo quindi di ottimizzare alcuni processi che in Python risultano essere colli di bottiglia di una data task, utilizzando linguaggi che lavorano a livello più basso, oppure semplicemente utilizzando librerie specifiche che non sono state implementate in Python. Si pone infatti molta attenzione al permettere il wrapping di moduli o librerie nei linguaggi sopra citati per aumentare le performance di Python, utilizzando sue estensioni come ad esempio Cython, un estensione con annotazioni sui tipi che permette la produzione tramite il suo compilatore di codice in puro C che viene poi compilato in moduli Python; oppure tramite l'utilizzo di sistemi come Simplified Wrapper Interface Generator o SWIG, che consente, una volta prodotta la descrizione dell'interfaccia per la comunicazione, il wrapping di grandi librerie C e C++, permettendo idealmente quindi di chiamare librerie a runtime senza però avere la disponibilità del codice sorgente; questa funzione è molto utile in particolare nel caso ci si

debba interfacciare con dell'hardware per il quale il produttore ha, ad esempio, fornito solo una libreria dinamica, DLL in Windows, senza però fornire il codice sorgente allo sviluppatore.

In generale un'ottima panoramica con i vantaggi di questo linguaggio è descritta nel paper "Python for Data Analytics, Scientific and Technical Applications" [12], dove si evidenziano in maniera chiara e coincisa i vari aspetti che ne hanno portato la grande espansione in ambito tecnico scientifico nel corso degli ultimi anni. Ad esempio, è indicativa oltre alla grande leggibilità del codice di cui si è già parlato, anche l'utilizzo di pochissime linee di codice a parità di funzioni rispetto ai linguaggi concorrenti, come si può evincere dal grafico sottostante, che gli ha permesso di ricoprire una grande importanza per quanto riguarda applicazioni in ambito Machine Learning e in generale Artificial Intelligence.

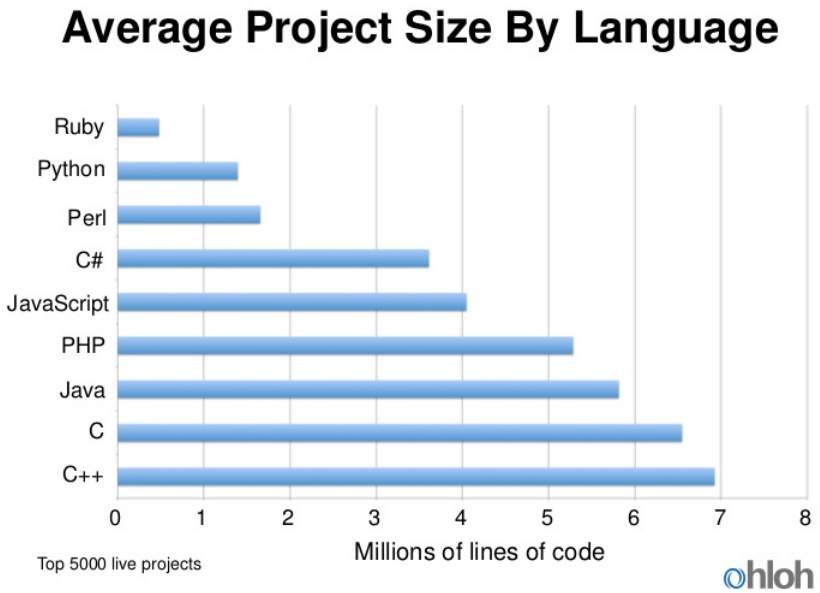

Figure 3.1. Rappresentazione grafica della media di grandezza per progetto a seconda del linguaggio. Credit to https://bit.ly/38LgaHQ

Gli altri vantaggi descritti nel paper sono stati tutti largamente affrontati in questo e nel precedente paragrafo, per questo vorrei soffermarmi sulla

possibilità di rappresentare in maniera immediata e facilmente comprensibile grandi quantità di dati, grazie alle numerose librerie utilizzate per data visualization. Questi moduli, fondamentali in ambito AI, non sono un requisito necessario per il nostro scopo, ma una volta cominciati a raccogliere i dati dei vari esperimenti potrà essere molto utile visualizzare in maniera compatta e funzionale ciò che si è raccolto, aggregando le informazioni di maggiore interesse e mostrandole in modo che anche gli esperti del dominio di utilizzo che però non hanno competenze informatiche di alcun tipo, possano comprenderne l'importanza e le caratteristiche. A questo ultimo punto si collega l'ultimo paper che vorrei citare in questa sezione [13] che analizza in maniera oggettiva come in ambito scientifico avere un minimo di skill di programmazione stia diventando molto importante poiché molte piccole task tecniche di facile risoluzione, non hanno ancora una standardizzazione informatica e il problema potrebbe essere risolto da degli esperti del dominio con un minimo di competenze informatiche. Nel nostro progetto, inoltre, finita la fase sperimentale, non è detto che gli esperimenti continuino a venir seguiti da professionisti dotati di skill di programmazione e quindi, anche se la struttura rimarrà la stessa di quella consegnata al termine dello studio, potranno essere necessarie delle piccole modifiche che potrebbero essere fatte anche da professionisti legati all'ambito di applicazione che sono in possesso di una piccola quantità di skill di programmazione. Per questo scopo Python grazie alla sua semplicità è perfetto, lo dimostra il suo utilizzo anche in classi di scuola superiore per l'introduzione alla programmazione, la sua scelta permetterà potenzialmente una maggiore manutenibilità e facilità di modifica anche a professionisti di altri settori scientifici.

3.1.3 La scelta per la Graphic User Interface: Tkinter

Scelto il linguaggio bisognava a questo punto decidere gli altri componenti che avrebbero formato questo ambiente applicativo all'interno del quale sarebbero state effettuate le varie esperienze. Compresa la necessità di un'interfaccia dalla quale permettere l'avvio degli esperimenti e permettere un minimo di gestione dei dati del partecipante è stata chiara la necessità di scegliere una

libreria per la gestione dell'interfaccia utente. Come si è detto la desktop application sarà installata unicamente nei laboratori di ricerca all'interno del quale sarà effettuato lo studio, per questo si è deciso di non concentrarsi sulla User Experience dell'utilizzatore al di fuori dell'esperimento in sé e quindi di non investire tempo nell'ottenere una soluzione particolarmente elaborata o gradevole esteticamente. Si è scelto di produrre un software funzionale e minimale, con pochi bottoni e funzionalità non necessarie, almeno in questa prima parte del progetto, sul modello delle tipiche applicazioni di laboratorio di qualche anno fa: colori quasi assenti, tutto sul grigio; presenza di pochi bottoni e menù a tendina; struttura e aspetto minimale senza troppa cura per la parte estetica. Queste poche necessità, con la voglia di non appesantire troppo il codice e il progetto in sé, oltre alla grande quantità di esempi nella letteratura, nonché tutorial e qualunque tipo di documentazione esistente, hanno fatto ricadere la nostra scelta su Tkinter. Tkinter è un binding per Python del toolikit per la GUI Tk, che viene utilizzato in accoppiata con il linguaggio di alto livello Tcl (le chiamate di Tkinter vengono tradotte in comandi Tcl a dopo sono passati all'interprete integrato), questa sinergia ha larga diffusione per quanto riguarda la produzione GUI su vari linguaggi e piattaforme. Nello specifico Tkinter è lo standard de facto per quanto riguarda l'interfaccia grafica in Python, ad esso sono state prodotte numerosissime alternative, alcune anche utilizzate all'interno di librerie inserite nel progetto (come Pyglet o PyQt) che permettono un maggior numero di funzionalità, ma richiedono sicuramente uno studio maggiore e un numero elevato di righe di codice per ottenere risultati più complessi e apprezzabili. Possiamo continuare la trattazione stabilendo aspetti positivi e negativi di Tkinter, elenchiamo le sue caratteristiche partendo dai pro: una grande maturità e stabilità della libreria, poiché è lo standard per Python da circa 30 anni; ha un porting ottimo su tutte le piattaforme, formendo widgets nativi sia in ambiente Mac che Windows; il suo binding è sicuramente più flessibile e consistente rispetto a quello di altre librerie spesso anche più quotate come ad esempio wxPython; ha una sintassi estremamente semplice e delle API di facile memorizzazione ; alcuni moduli della libreria sono davvero di ottima qualità, i

text e canvas widgets sono molto semplici e allo stesso tempo estremamente potenti, stesso discorso per i geometry managers (pack, place e grid) che permettono di gestire in maniera ottimale il layout della pagina senza grosse difficoltà. D'altra parte, troviamo alcuni contro da dover valutare: si basa su un modello di semplice utilizzo, ma in caso di interfacce complesse può risultare macchinoso da utilizzare; non ha un bell'aspetto e quindi non è indicato per un utilizzo a contatto con il pubblico (marketing ad esempio); i widgets sono ovviamente un wrapper attorno ai widgets di Tk, quindi il loro sorgente non è codice Python e questo può portare ad errori non ben identificati in caso di malfunzionamenti; è estendibile, che potrebbe essere un pro, ma spesso indica la necessità di dover obbligatoriamente scaricare dei toolkit addizionali per ottenere delle funzioni meno comuni; è una tecnologia destinata a morire e che sta venendo sostituita ovunque. Come si può vedere è sicuramente una scelta che non consiglierei in ambito commerciale, ma per il nostro utilizzo, in un ambito puramente tecnico in cui necessitavamo di un semplice entry point da cui lanciare gli esperimenti e inserire i dati dei partecipanti, è sicuramente un'opzione validissima che coniuga semplicità di utilizzo, velocità di apprendimento e minimalismo nell'interfaccia, permettendoci di creare in breve tempo un ambiente adatto alle nostre necessità e passare a costruire la parte più impegnativa della nostra applicazione.[14]

3.2 Il cuore dell'applicazione: gli esperimenti

Una volta composta la struttura dell'applicazione e prese le scelte relativa ad essa, bisogna decidere come costruire quello che è il vero nucleo di questo progetto e da cui verranno poi ricavati i dati che saranno analizzati in futuro: gli esperimenti. Come visto nel capitolo precedente gli esperimenti appartengono a due ambiti, neuromarketing e neuroriabilitazione, che però hanno spesso necessità simili e con i quali ho condiviso lo stesso metodo per concordare la struttura degli esperimenti. Per iniziare si sono dovuti tenere in considerazione alcuni aspetti tipici che caratterizzano la sperimentazione in laboratorio. Rispetto alla controparte online, la sperimentazione

in laboratorio deve tenere in conto il rischio di influenzare il partecipante in qualche modo, che si presenta non tanto nella visione dei video, quanto negli esperimenti legati al web design. In particolare, si è cercato di ridurre al minimo l'influenza dell'operatore che guiderà il partecipante all'interno dell'esperimento, questo dovrà interagire ampiamente con il partecipante solo nella fase iniziale. Prima dell'esperimento vero e proprio il partecipante verrà introdotto a cosa sarebbe andato a fare, senza entrare nello specifico dei task dell'esperimento per non influenzarne la prestazione. Successivamente viene fatto firmare il consenso informato e i vari moduli relativi alla privacy e al trattamento dei dati, questo passaggio è molto importante perché oltre alle varie liberatorie collegate all'utilizzo di strumenti per la registrazione dei biomarcatori, alcuni anche abbastanza invasivi, bisogna sottolineare nella modulistica come i dati raccolti non saranno utilizzati solo per la cura del paziente (nel caso della neuroriabilitazione), ma anche per lo studio teorico su cui si sta lavorando. Questa fase è molto delicata e rispetto alla controparte online, può essere di aiuto avere un operatore che possa chiarire i dubbi del partecipante evitando problemi di ambiguità e comprensione. Finita la fase di modulistica e soprattutto la preparazione della strumentazione sul partecipante, l'esperimento vero e proprio può cominciare e in questa fase l'operatore deve cercare di essere il più possibile esterno dal momento del lancio del software dello specifico esperimento. Questa necessità di allontanare l'operatore è dettata dalla volontà di evitare quel fenomeno chiamato "Demand characteristics", ossia un artefatto sperimentale per cui il partecipante di uno studio tende ad avere una sua interpretazione dello scopo dell'esperimento. Maturata una sua interpretazione, l'uomo ha sempre l'impressione o l'aspettativa di essere valutato per il modo in cui la società ci ha abituato, per questo cercherà di adattare il suo comportamento per adattarsi nella maniera migliore allo scopo dello studio. Risulta quindi estremamente importante non evidenziare lo scopo dello studio nei suoi particolari e sottolineare chiaramente, sia in maniera scritta nella modulistica che a voce da parte dell'operatore, che il partecipante non verrà in nessun modo valutato o giudicato e di comportarsi quindi nella maniera più naturale

possibile. Inoltre, una volta iniziato il processo sperimentale vero e proprio, l'operatore deve cercare di essere il più possibile esterno al processo, rimanendo però nella stanza per poter supportare il partecipante in caso di problemi imprevisti riguardo principalmente la strumentazione. Si può quindi capire come sia fondamentale che la struttura e il flusso dell'esperimento debbano essere chiare e non necessitare mediamente di ulteriori spiegazioni, così da permettere l'estraneità dell'operatore. Altro fattore importante, oltre alla chiarezza, è la riduzione di stimoli esterni che possano distrarre il partecipante dallo studio in sé. Questa è la motivazione per la scelta, ad esempio, di costruire un browser estremamente scarno e senza una barra di ricerca per aprire i siti richiesti, oppure la decisione di impostare gli esperimenti più brevi a schermo intero, in maniera che fossero più immersivi possibile e non permettessero distrazioni e quindi perdite di informazione o peggio reazioni emotive dovute ad altri elementi nella stanza che falserebbero lo studio. Per gli studi riguardanti la neuroriabilitazione, si è deciso anche di disabilitare più input gestiti dall'utente, come ad esempio la tastiera, per far sì che il partecipante non sviluppasse stress o tensione dovuta a doversi impegnare nell'eseguire una task o dover ragionare su come proseguire nell'esperimento. Una volta concordati questi punti ai quali porre attenzione con i team di ricerca, i gruppi mi hanno fornito il protocollo dell'esperimento, che analizza in maniera precisa scandendo i vari passaggi, come dovrà essere effettuato l'esperimento nelle sue varie fasi. Presi in mano questi documenti ho dovuto immaginare un prodotto il più simile possibile a quello strutturato nella trattazione teorica e in diversi incontri abbiamo concordato dei compromessi per permettere una produzione rapida e funzionale del software. Si sono utilizzati degli use case, sia grafici che non, per evidenziare le varie fasi del singolo esperimento, evitando ambiguità e verificando che la mia idea fosse completamente allineata con il gruppo di ricerca, e una volta terminata la gestione del flusso si è passati ad analizzare la parte grafica. Per mostrare le schermate a chi si è occupato di ideare l'esperimento si sono utilizzati dapprima degli schizzi molto approssimativi delle schermate dell'applicazione e delle varie fasi dei test, per passare poi alla breve costruzione di alcuni wireframe così

da mostrare in maniera chiara l'aspetto e la presenza di alcune funzionalità al resto del team. (Inserisci screen wireframe)

Dopo aver analizzato i vari wireframe ed aver concordato eventuali cambiamenti, si è passato all'implementazione vera e propria di applicazione ed esperimenti. Mentre per il software applicativo le tecnologie sono già state presentate, andremo ora a fare una breve panoramica su quali siano le librerie esterne utilizzate per semplificare e migliorare la produzione di esperimenti neuroscientifici efficaci e che rispettassero i vincoli che ci eravamo preposti.

3.2.1 Psychopy

Per creare gli esperimenti e quindi presentare gli stimoli in maniera professionale e funzionale ai partecipanti, esistono molti software specifici per esperimenti neuroscientifici, per questo per il core dei nostri test è stato utile scegliere una singola libreria, alla quale poi aggiungere (o sostituire in alcuni casi come vedremo successivamente) altri moduli a seconda delle nostre necessità, per poter riprodurre in maniera quanto più fedele i protocolli descritti teoricamente dal gruppo di ricerca. La nostra scelta tra i tanti concorrenti è ricaduta su Psychopy, questa è una libreria open-source scritta in Python che unisce alla semplicità del linguaggio, i vantaggi grafici di OpenGL (una specifica che definisce API per la scrittura di applicazioni che producono computer grafica 3D, è inoltre standard de facto di questo ambito in ambiente Unix). Psychopy permette la presentazione e la gestione di un vario numero di stimoli dei generi più disparati, si va da scritte a semplici forme, fino ad arrivare a contenuti multimediali di vario genere. Gli stimoli sono il cuore dell'esperimento, in psicologia uno stimolo è un qualunque oggetto, ma anche evento, che provoca una reazione in un organismo, questa reazione può essere sia sensoriale che comportamentale; lo stimolo è un concetto cardine della psicologia comportamentale che ha nel condizionamento classico di Pavlov una delle sue teorie chiave, dove viene ad esempio esplicitata la differenza tra stimolo incondizionato e condizionato. Famoso è l'esperimento di Pavlov in cui viene mostrato come lo stimolo incondizionato, nello specifico il cibo, produca delle reazioni nell'organismo in maniera indipendente; queste

reazioni possono essere riprodotte da uno stimolo condizionato, permettendo di collegarlo all'iniziale stimolo incondizionato, per esempio il suono di una campanella per avvertire dell'arrivo del cibo, che altrimenti sarebbe completamente estraneo a questa reazione; per questo nel cane veniva provocata la salivazione anche solo con il suono della campanella poiché questa era collegata all'arrivo del cibo nel cervello dell'animale. Psychopy permette quindi la presentazione di stimoli al partecipante all'esperimento e vari modi per controllare e registrare le sue reazioni a questi. Non è solo un modulo esterno da poter importare, ma il motivo per cui è stato scelto è il fatto che viene fornita anche un'applicazione formata da due principali oggetti il Builder e il Coder, che si adattano a seconda della necessità di flessibilità, tempo e competenze richieste. Il Builder permette di costruire un esperimento tramite una comoda interfaccia grafica seguendo la logica "drag and drop", si va infatti prima a decidere la struttura dell'esperimento composta principalmente da routine, i macro-blocchi del flusso sperimentale, e loop. Le routine sono le varie fasi di un esperimento, ogni routine identifica una specifica situazione a cui il soggetto è sottoposto e alla quale deve reagire, dentro la routine si inseriscono infatti dal menu di destra i component che identificano le varie tipologie di stimoli o di azioni che devono essere permesse o registrate (come, ad esempio, l'abilitare o disabilitare specifiche azioni di mouse e tastiera). Ogni component ha diversi parametri che possono essere settati a piacimento del creatore dell'esperimento tramite comodi menù, che facilitano notevolmente la costruzione della schermata che verrà mostrata al partecipante. Infine le varie routine devono essere poste nell'ordine voluto oppure ripetute tramite dei loop per creare il flow dell'esperimento desiderato dal ricercatore. Una volta buildato l'esperimento l'applicazione produce il codice sorgente in puro Python che può essere modificato a più basso livello nel Coder, un semplice editor di testo che permette una maggiore flessibilità rispetto alla seppur comoda interfaccia grafica del Builder. Questo dualismo permette l'accesso alla creazione di esperimenti anche a persone non pratiche di programmazione e una costruzione della struttura molto veloce anche per un programmatore esperto. Il mio approccio è stato, infatti, quello di costruire gli esperimenti

all'interno del builder e una volta creato il file .py corrispondente, importarlo nel mio progetto come un modulo esterno. A questo punto effettuare le ultime modifiche direttamente sul codice così da adattarlo a ciò che mi serviva nello specifico test, soprattutto per quanto riguarda la gestione dei vari input necessari, su tutti il dataset di fotografie IAPS che ho dovuto indicizzare tramite un file .csv. Uno dei punti di forza di Psychopy è che, essendo pensato come un software per esperimenti neuroscientifici, ha come uno dei requisiti principali il doversi interfacciare con una grande varietà di hardware utilizzato in ambito di ricerca. Lo studio [15] oltre a presentare una carrellata degli stimoli più utilizzati all'interno dell'applicazione, con particolare enfasi sui PatchStim (che permettono la presentazione di contenuti multimediali statici come immagini), va a sottolineare la predisposizione della libreria e del sistema in generale ad interfacciarsi con strumenti esterni tipici della neuroscienza come Eye-Tracker, EEG e fMRI, tramite l'utilizzo, per input e output, delle porte seriali e parallele (principalmente porte USB), anche se spesso si presenta la necessità di chiamare alcune dynamically-loaded libraries (DLL in Windows). Questa facilità di comunicazione era essenziale, come abbondantemente analizzato, per gli scopi e le necessità del progetto, che basano la gran parte della struttura degli esperimenti sulla registrazione dei biomarcatori. Nonostante negli esperimenti sviluppati finora non vi sia bisogno di una grande reattività del sistema, parlando anche con i componenti del gruppo, non è assolutamente da escludere che in futuro si voglia aggiungere qualche test fortemente time-based, così da porre il partecipante sotto una qualche forma di stress prestazionale. Per questo motivo un altro grande vantaggio che ho preso in esame sono le prestazioni del software per quanto riguarda il tempo di risposta registrato, un ampio e approfondito confronto viene esposto in [16] dove vengono confrontati una gran numero di diversi software sia di laboratorio che online, riguardo le reali "timing performance" che riescono a raggiungere nell'utilizzo pratico, con differenze di precisione di pochi millisecondi. In particolare, gli studiosi vanno a confrontare i più famosi software neuroscentifici andando a misurare la precisione e l'accuratezza della presentazione degli stimoli, sia visivi che uditivi, insieme ai tempi di risposta

registrati utilizzando un Black Box Toolkit. Da questi studi Psychopy è risultato uno dei package migliori permettendo una precisione media di circa 1 millisecondo, uno dei migliori a mantenere queste presentazioni indipendentemente dal sistema operativo sottostante, e un tempo di risposta inferiore ai 3.5 millisecondi; un risultato ottimo che mi ha permesso di considerarlo come un'ottima scelta lasciando viva l'ipotesi di futuri esperimenti time-based.

3.2.2 OpenCV

OpenCV (Open Source Computer Vision Library) è una libreria software open source di computer vision e machine learning, pensata per "fornire un'infrastruttura comune per le applicazioni di computer vision e per accelerare l'uso della machine perception nei prodotti commerciali." [17] La libreria è scritta in C/C++ ma ha molti bindings tra cui appunto anche quello nel linguaggio Python che è il motivo per cui è stata scelta per il nostro progetto. Nata nel 1998 come un progetto di ricerca interno ad Intel era inizialmente focalizzata nel migliorare le applicazioni CPU-intensive e tutto ciò che ne conseguiva. Nel 2000 è stata rilasciata sotto licensa BSD, cosa che permetteva alla libreria di essere utilizzata per le proprie applicazioni e essere modificata con facilità, alla IEEE Conference on Computer Vision and Pattern Recognition. Gli obiettivi del progetto erano infatti leggermente cambiati rispetto all'inizio e si identificavano di più in quelli che sono anche ora, cercando di fornire del codice portabile e performante per applicazioni di computer vision in ambito commerciale, ma anche fornire una infrastruttura comune in questo ambito, favorendo la distribuzione della conoscenza in campo computer vision, rendendo più accessibile il codice per i futuri programmatori. Ovviamente il supporto a questa libreria è continuato nel tempo, poiché trattando un argomento che è in esplosione negli ultimi anni si è dovuta tenere al passo per quanto riguarda performance e innovazioni, per questo il suo sviluppo è stato seguito inizialmente da un team di sviluppatori russi indipendenti e ora dalla fondazione no-profit OpenCV.org. Una delle release più importanti è quella risalente al 2010 in cui viene introdotto un nuovo modulo trattante l'accelerazione GPU [18] che va a coprire gran

parte delle funzionalità precedenti della libreria ed è stata sviluppata utilizzando CUDA e beneficiando del suo ecosistema e di tutte le librerie che lo compongono. Questo modulo permette un semplice utilizzo dei processi di accelerazione GPU senza la necessità di un training su questo specifico argomento; permette inoltre utilizzi più avanzati, alternando utilizzo di CPU e GPU, spostando esplicitamente informazioni dall'una all'altra memoria, o per esempio con processi simultanei (grazie a meccanismi asincroni forniti da CUDA) si potrebbe processare con la GPU un dato, mentre quello successivo sta già venendo calcolato e trasferito. Le enormi potenzialità di OpenCV non ne inficiano però la sua semplicità e le grandi possibilità che fornisce all'utente, nel nostro caso infatti si doveva registrare la webcam o la telecamera associata al pc su cui gli esperimenti vengono effettuati. Si deve quindi registrare un video i cui frame saranno poi analizzati dal nostro algoritmo di facial coding e andranno ognuno a influenzare la scelta sull'emozione predetta in quella data finestra temporale.

```
cap = cv2.VideoCapture(0)

# Define the codec and create VideoWriter object
fourcc = cv2.VideoWriter_fourcc(*'DIVX')
out = cv2.VideoWriter('output.avi', fourcc, 20.0, (640, 480))

while (cap.isOpened()):
    ret, frame = cap.read()
    if ret == True:
        #  frame = cv2.flip(frame,0)

        # write the flipped frame
        out.write(frame)

        cv2.imshow('frame', frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    else:
        break

# Release everything if job is finished
cap.release()
out.release()
cv2.destroyAllWindows()
```

Figure 3.2. Frammento di codice per la registrazione della webcam, mostra la facilità di OpenCV nella gestione della periferica

Quello riportato è un frammento di codice di test, utilizzato inizialmente per fare delle prove di cattura della webcam. Come si vede nel frammento di codice dopo aver importato la libreria cv2, con queste poche righe in un ciclo while viene registrata la telecamera selezionata (la numero 0 perché è quella di default) e il risultato della registrazione è un insieme di frame che vengono raccolti separatamente. A questo punto i frame vengono mostrati in sequenza con cv2.imshow, puramente a scopo di test, e infine vengono posti in sequenza per ricostruire il video che in questo caso abbiamo deciso di chiamare output.avi e che verrà salvato nel filesystem. Possiamo quindi osservare come ogni frame è trattato indipendentemente e perciò una volta pronto l'algoritmo di machine learning, il singolo frame potrà essere direttamente elaborato per poter così estrarre l'emozione osservata.

3.2.3 Python-VLC

Python-VLC è un modulo ideato e scritto da Olivier Aubert che fornisce un binding per Python utilizzando il modulo ctypes (che permette di creare funzioni che seguiranno le convenzioni dipendenti dalla piattaforma quando saranno chiamate a runtime) per l'API nativa libVLC del corrispondente video player VLC molto diffuso e conosciuto [20]. Il media player VLC è un player multimediale e portabile in grado di riprodurre un gran numero di formati sia audio che video e che negli ultimi anni si sta concentrando molto anche sullo streaming video in varie forme. È nato come un progetto universitario in Francia, ma attualmente è diffuso in tutto il mondo e ad esso collaborano sviluppatori in oltre 20 paesi. È pensato con una struttura modulare in quanto si può decidere di utilizzare solo alcune delle sue funzionalità nel caso si sia interessati solo ad alcuni tipi di output o a specifici formati. Nello specifico libVLC [**?**] è il core engine, nonché l'interfaccia esterna di programmazione su cui si basa il lettore multimediale VLC, utilizzando questa libreria, si devono quindi poter raggiungere le stesse funzionalità del player originale anche nella propria applicazione. Come detto è modularizzato in centinaia di plugin ognuno riguardante differenti funzionalità, questo permette allo sviluppatore che intende inserire il lettore all'interno del proprio

progetto di creare una vastissima gamma di applicazioni multimediali tutte aventi lo stesso core VLC. Le finalità della libreria sono molteplici, da quelle più classiche e basilari (come poter avviare un gran numero di formati e codifiche, funzionare indipendentemente dalla piattaforma sia essa desktop o mobile, permettere il playback di qualunque tipo di media mentre si naviga nel menù), per arrivare a funzioni più innovative e specifiche (si concentra ad esempio sul supporto a video a 360 gradi e file audio 3D o la possibilità di effettuare lo streaming su render lontani come ad esempio il Chromecast). Ovviamente il modulo utilizzato, il già citato Python-VLC, essendo semplicemente un binding potrà fornire all'utente, seppur con alcune limitazioni, le stesse feature e possibilità della libreria di partenza. La scelta di utilizzare questa libreria si è resa necessaria poiché uno degli esperimenti di neuromarketing citati in precedenza necessitava di visualizzare un video di circa 20 minuti in una buona qualità. Questo provocava dei problemi di lag e desincronizzazione tra audio e video dovuti a problemi riscontrati anche da altri utenti legati alla libreria moviepy utilizzata in ambiente Psychopy; queste problematiche nella riproduzione anche se molto piccole, provocavano sicuramente un fastidio non ignorabile. Con il team di ricerca abbiamo concordato che questo avrebbe potuto falsare leggermente i risultati, introducendo stress e nervosismo; inoltre il controllo del video all'interno di Psychopy non è ottimizzato poiché è pensato per un'altra tipologia di esperimenti. La nostra scelta quindi è stata di costruire un lettore multimediale all'interno della nostra applicazione basandoci sulla libreria Python-VLC.

Come si vede dall'immagine 3.3, si è deciso di mantenere una interfaccia spoglia e minimale a sfondo grigio come il resto dell'applicazione per evitare distrazioni da parte dell'utente. Gli unici bottoni sono quelli utilizzati per fare il "Play" e "Pause" del video e quello del volume, che può essere o attivato/disattivato oppure modificato tramite lo slider. Sotto infine è rappresentata la barra che indica l'avanzamento della puntata. Questa conformazione permette all'utente di mettere in pausa l'esperimento, bloccando anche la ripresa da parte del sistema del volto per il face coding, senza quindi andare a inficiare i risultati, nel caso insorga un qualunque problema tra cui

Figure 3.3. Screen dell'applicazione che mostra la struttura e l'aspetto del lettore multimediale prodotto

quello di dover sistemare ad esempio gli elettrodi applicati sul partecipante che possono essere stati spostati inavvertitamente. Per ora si è deciso di non usare un'opzione fullscreen per evidenziare, durante gli spot, come il partecipante in alcune fasi potesse essere così poco attratto dalla pubblicità da spostare il suo sguardo altrove all'interno dello schermo. L'effettiva utilità di questa feature verrà verificata in futuro solo con gli esperimenti sul campo quando l'applicazione verrà effettivamente utilizzata quotidianamente per lo studio. Per la costruzione di questo semplice media player ho modificato la struttura già proposta da Patrick Fay in un blog, ho dovuto riadattare alcuni componenti e toglierne degli altri, sia per la necessità dell'esperimento che per la diversa versione sul quale era basato il player (Python 3.4 contro 3.6); ma è comunque stato un ottimo punto di partenza che ha semplificato e velocizzato notevolmente il mio lavoro.

3.2.4 CEF Python v66.0

CEF Python [21] è un progetto open source che si pone come obiettivo quello di fornire i bindings Python per il Chromium Embedded Framework, appunto CEF, fondato da Czarek Tomczak nel 2012 e che si trova attualmente alla versione 66.0. CEF nello specifico è anch'esso un progetto open source, fondato nel 2008, che si basa sul progetto Google Chromium, ma mentre quest'ultimo si concentra principalmente sullo sviluppo della famosa applicazione Google Chrome, CEF si concentra sul facilitare l'inserimento di un embedded browser all'interno di applicazione di terze parti. Ha raggiunto una grande notorietà essendo distribuito, come afferma la pagina del progetto [22], in oltre 100 milioni di istanze in tutto il mondo. Questo successo oltre alle sue funzionalità è dovuto alla sua praticità e facilità, che permette all'utente di non dover comprendere e adattare il codice di Chromium che ne è alla base, ma di utilizzare tutte le sue funzionalità grazie alle stabili API fornite, ricalcando le features del prodotto di base con la necessità di solo una leggera integrazione con l'applicazione di destinazione. Vari sono i casi d'uso per cui il progetto è stato pensato, dall'essere utilizzato come un motore di rendering basato su HTML5 per rimpiazzare classici framework per GUI Desktop, fino al testing automatizzato di web application con molte funzioni avanzate rispetto ai concorrenti. Nel nostro caso è stato utilizzato per l'uso più standard, ossia permettere l'embedding di un browser all'interno della nostra applicazione desktop senza l'utilizzo di applicazioni esterne. Questa scelta è stata operata per permettere l'interazione di tutti gli strumenti direttamente all'interno dell'applicazione, senza dover aprire un browser esterno evitando possibili conflitti dovuti dal dover runnare l'applicazione in background, mentre si utilizzano programmi che a volte possono essere anche molto pesanti, come ad esempio Google Chrome. Inoltre, parlando con il gruppo di ricerca, è sorta la necessità di inserire una colonna laterale in cui mostrare le varie istruzioni per il partecipante, così da permettere allo stesso di orientarsi autonomamente all'interno dell'esperimento, evitando di essere guidato eccessivamente dall'operatore e quindi permettendo il non presentarsi dei fenomeni trattati in precedenza come, ad esempio, il già citato "Demand

characteristics".

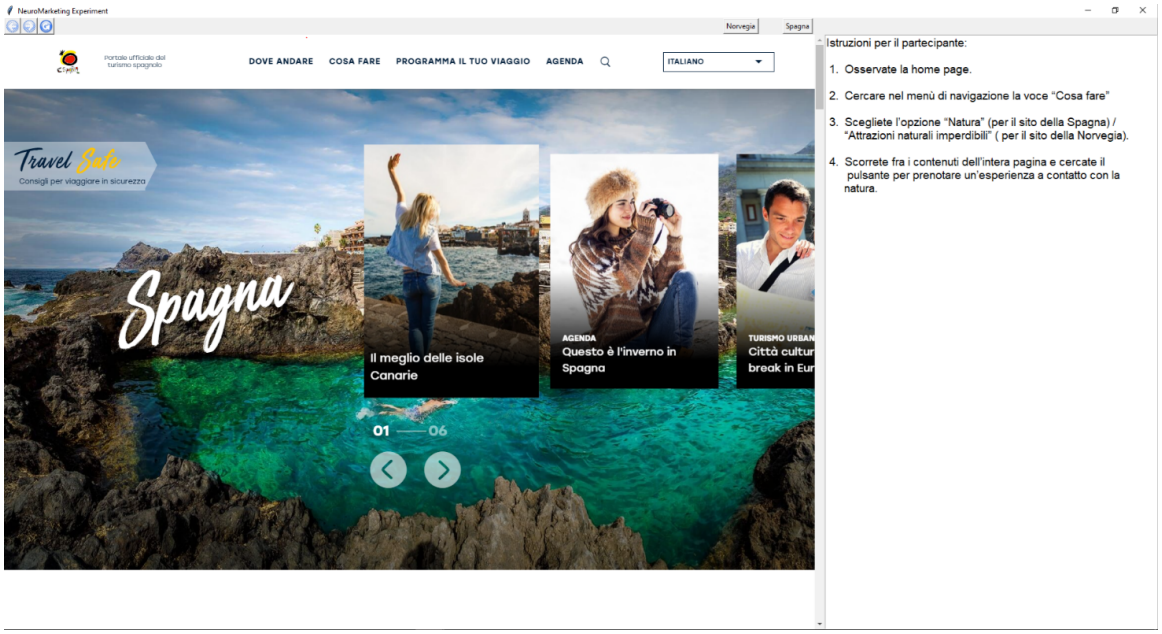

Figure 3.4. Screen dell'applicazione che mostra la struttura e l'aspetto browser embedded prodotto

Come si può osservare nell'immagine 3.4, anche questo componente ha un aspetto molto scarno, ci si è basati per la costruzione su uno degli esempi forniti dal repository Github ufficiale del progetto CEF Python, ma usando questo come base di partenza si sono compiute alcune modifiche mirate. Innanzitutto, come già detto la schermata è divisa in due aree principali, sulla sinistra è presente la classica interfaccia web mostrata dal browser, corrispondente all'URL della pagina al quale si è stati reindirizzati, mentre sulla destra vi è una colonna bianca con in nero descritte le istruzioni associate a quella data pagina web. In alto la top bar è stata resa molto più minimale, mettendo unicamente i tre bottoni corrispondenti alle azioni di avanti, indietro e ricarica della pagina; mentre sulla destra sono stati inseriti dei pulsanti particolari che variano a seconda dell'esperimento e che portano il browser alla pagina su cui il partecipante dovrà compiere delle azioni. La presenza di questi bottoni, che di fatto limitano la mobilità tra le pagine web da parte del partecipante, è stata pensata (insieme alla rimozione della classica barra

di ricerca tipica di qualunque browser) per permettere all'utente di concentrarsi unicamente sull'esperimento, semplificare lo svolgimento di questo ed evitare la partecipazione dell'operatore. Come si è visto si è cercato di dare uniformità agli esperimenti, se non sempre nell'aspetto estetico, di sicuro costantemente su questi accorgimenti che vengono tenuti in considerazione in tutta l'applicazione. A questo punto il partecipante verrà registrato mentre naviga, legge e interagisce con la pagina web, direttamente all'interno della nostra applicazione, così da poter registrare i biomarcatori osservati e procedere a collezionare i risultati che questo esperimento di web design punta a raccogliere.

Chapter 4

Stato dell'arte: verso l'Emotion recognition

Fin'ora si sono introdotti i due ambiti di applicazione del software che sarà prodotto al termine di questo progetto di ricerca, si potrebbe però pensare che queste discipline, strettamente legate alla neuroscienza, non abbiano legami in comune con l'informatica. In realtà tutti questi ambiti di ricerca, insieme a diverse altre, dalla filosofia alla psicologia, dall'antropologia alla genetica, fanno parte di una grande famiglia di scienze, raccolte sotto il nome di scienze cognitive, che si concentrano sullo studio della mente e dei suoi processi. Questi studi cercano di analizzare le facoltà mentali di un sistema pensante, non solo umano, con lo scopo di comprenderlo a fondo e su vari livelli, dalla composizione dei circuiti neurologici, fino ai processi decisionali guardati a più alto livello. In particolare Marr[23] identificava tre diversi livelli di rappresentazione, il terzo e più alto livello è quello definito come "hardware implementation" ed è quello di nostro interesse. Questo livello indicava lo studio del come fosse possibile implementare e replicare gli stessi processi che avvenivano all'interno di un essere pensante, all'interno di una macchina tramite algoritmi e rappresentazioni di vari tipi. Appare quindi evidente come l'Intelligenza Artificiale o AI, faccia parte dell'insieme di discipline che compongono le scienze cognitive come mostrato in figura 4.1.

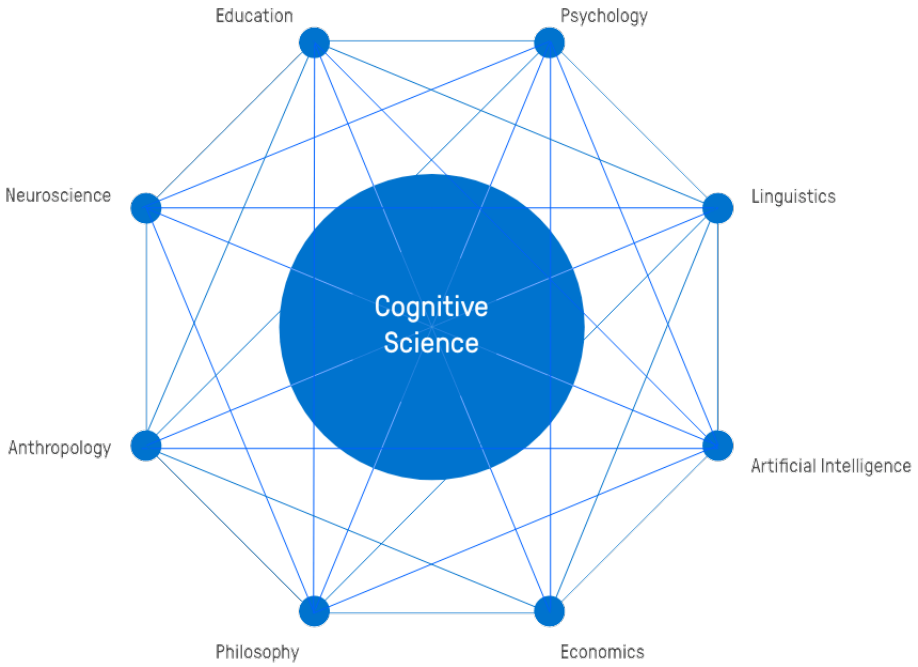

Figure 4.1. Rappresentazione grafica delle maggiori discipline che compongono l'ambito di ricerca noto come Scienze Cognitive. Credit to https://www.meicogsci.eu/cognitive-science.html

L'Intelligenza Artificiale è infatti una disciplina di confine, poiché studia sia a livello teorico che pratico, come produrre sistemi, sia dal punto di vista hardware che software, che presentino un comportamento che sia riconducibile a quello pensante e quindi umano; per questo deve scontrarsi con molti problemi etici, ma soprattutto deve collaborare e interfacciarsi con gli altri campi di ricerca appartenenti alle scienze cognitive. Come si vedrà nel seguito della trattazione, infatti, molte delle innovazioni apportate in questo campo derivano dall'osservazione e dallo studio del funzionamento del cervello, sia esso umano o animale, e dal tentativo di riprodurne le caratteristiche nella maniera migliore possibile. Tra le varie facoltà mentali che le varie scienze cognitive vanno a studiare, rientrano anche le emozioni, che sono il focus e il comun denominatore di questo studio, nonché ciò che ha permesso di legare campi apparentemente lontani come il Neuromarketing, la Neuroriabilitazione e l'Intelligenza artificiale. L'importanza delle emozioni, soprattutto in ambito sociale per un'animale votato alla socialità quale è l'uomo, è da

subito sembrato di grande interesse per gli studiosi, in particolare come le emozioni fossero effettivamente collegate alle espressioni facciali e cosa queste venissero a indicare. Già Darwin aveva infatti ipotizzato come le espressioni facciali fossero universali e fossero un tratto frutto dell'evoluzione dai nostri antenati animali. Egli, infatti, asseriva che le espressioni fossero innate nell'essere umano e quindi essenziali a livello evolutivo per la sopravvivenza della specie, in particolare per quel concetto di socialità tanto caro all'uomo. Numerosi studi sono stati effettuati per dimostrare come persone di culture e parti del mondo totalmente diverse, esprimano però le medesime emozioni con espressioni simili e ricorrenti, essendo quindi in grado di riconoscerle quando le vedono espresse da altri esseri umani, di notevole importanza a tal proposito sono gli esperimenti e gli studi di Ekman [24]; non solo, questi studi sono anche stati allargati su alcune specie animali evidenziando come, ad esempio, gli scimpanzè sono in grado di comunicare molte espressioni facciali che ritroviamo anche nell'uomo [25]. Ovviamente anche in questo ambito il pensiero non è uniforme, vi sono molte critiche e versioni discordanti rispetto alla brevissima panoramica fatta, che si concentrano in particolare sulle espressioni facciali, non solo come caratteristica innata, ma come frutto della società in cui la persona vive oppure come tentativo di trasmettere un determinato messaggio. Ad ogni modo l'importanza che questo aspetto assume per l'essere umano rimane ed è ben chiaro, sia per comprendere qualcosa in più sulla socialità, sia per meglio comprendere i processi emozionali interni alla persona. In particolare, questo capitolo rappresenta il percorso che è stato fatto per arrivare a poter identificare le emozioni provate dal partecipante durante i relativi esperimenti, che saranno poi analizzate e studiate dai relativi gruppi di ricerca per trarne delle conclusioni utili. Per il riconoscimento delle emozioni tramite le espressioni facciali si è fatto uso di una disciplina dell'Intelligenza Artificiale che sta avendo un grande successo negli ultimi anni: il Machine Learning

4.1 Machine Learning

Si è spesso citato il machine learning in precedenza, senza però mai entrare nello specifico. Il machine learning è una branca dell'intelligenza artificiale che raccoglie una grande quantità di metodologie al suo interno e che ha avuto uno sviluppo davvero esponenziale nel corso degli ultimi 30 anni. L'intelligenza artificiale, come già accennato, si caratterizza per il cercare di sviluppare algoritmi o metodi in generale che permettano alla macchina di agire in una maniera che può essere definita razionale, potendola associare così al pensare e agire umano; in particolare la macchina agirà inseguendo il risultato che presenta la maggiore utilità, cercando quindi di massimizzarla avendo anche a che fare nelle applicazioni reali con una buona dose di incertezza, ma gestendola in modo da inseguire il miglior risultato atteso. Il machine learning può essere descritto dalla famosa frase di Tom Mitchell: "Si dice che un programma apprende dall'esperienza E con riferimento a alcune classi di compiti T e con misurazione della performance P, se le sue performance nel compito T, come misurato da P, migliorano con l'esperienza E." [26]. Volendo semplificare si può identificare come appartenente all'ambito machine learning qualunque tipo di applicazione che facendo uso di metodi statistici continua a migliorare le sue performance nel tempo autonomamente utilizzando i dati che gli vengono sottoposti nel corso del tempo; questo è il concetto di esperienza che lo lega all'essere umano e che gli permette di compiere degli enormi miglioramenti senza bisogno di una riprogrammazione. Si desidera infatti che un computer performi bene in una specifica task, senza però essere stato programmato specificatamente per quella, tutto questo compiendo diverse azioni legate principalmente all'individuazione ed estrazione di patterns nei più vari domini a cui le applicazioni si vogliono applicare. Si vuole quindi che il computer scopra autonomamente come performare in maniera ottimale in quello specifico compito basandosi sui dati forniti per il training, infatti in molti casi ci si trova davanti a problemi molto complicati, dove la cosa più semplice è aiutare la macchina a sviluppare la soluzione al problema fornendo dei dati significativi per la task in questione. Al momento le applicazioni di machine learning sono ovunque e penetrano nella

nostra vita di tutti i giorni sfuggendo agli occhi di osservatori non attenti, un esempio lampante sono le pubblicità da cui siamo invasi sul web, che sono sempre aggiornate su quelli che sono i nostri interessi, così da poterci sottoporre i prodotti più appetibili. Questi sistemi non fanno altro che usufruire dei dati che noi stessi gli forniamo in moltissime forme, dalle ricerche sui motori di ricerca alle pagine che seguiamo sui social network, per aggiornarsi e quindi migliorare le loro prestazioni, che in questo caso particolare cambiano punto di ottimo seguendo i nostri interessi che sono estremamente mutevoli. Come si può immaginare da questa breve descrizione due fattori hanno contribuito allo sviluppo di questa branca: il grande aumento della potenza hardware (unita alla maggiore accessibilità in termini economici), in particolare delle GPU, e la creazione dei big data, con annessa disponibilità di un'enorme quantità di dati a disposizione di chiunque abbia voglia di sperimentare. La necessità di esperienza e di migliorare continuamente da parte dei modelli di machine learning è infatti dipendente da una grande quantità di dati, che permettano numerose iterazioni dell'algoritmo ognuna delle quali porterà degli update al modello costruito e quindi idealmente, se ben progettato, ad un miglioramento nelle sue prestazioni. Per concludere, si può osservare l'estremo sviluppo di quest'ambito osservando i due grafici riportati qui sotto 4.2. Quello a sinistra rappresenta la concezione classica del machine learning (e del deep learning di cui tratteremo successivamente), come appunto una pura branca dell'artificial intelligence. A destra troviamo invece un'altra corrente di pensiero che si sta diffondendo che vede queste due discipline come diverse, con alcune metodologie e approcci comuni.

Una possibile visione sulla differenza tra queste due discipline è fornita da Judea Pearl [27], secondo la quale la differenza principale riesede nel fatto che il machine learning utilizza osservazioni passive per poter imparare, migliorare e poter quindi effettuare predizioni sempre più precise. Invece l'intelligenza artificiale necessita di un agente attivo che, per ottenere il suo obiettivo di un outcome vicino il più possibile a quello di una mente pensante, deve interagire con l'ambiente per imparare e di conseguenza agire in modo da massimizzare le sue chance di successo.

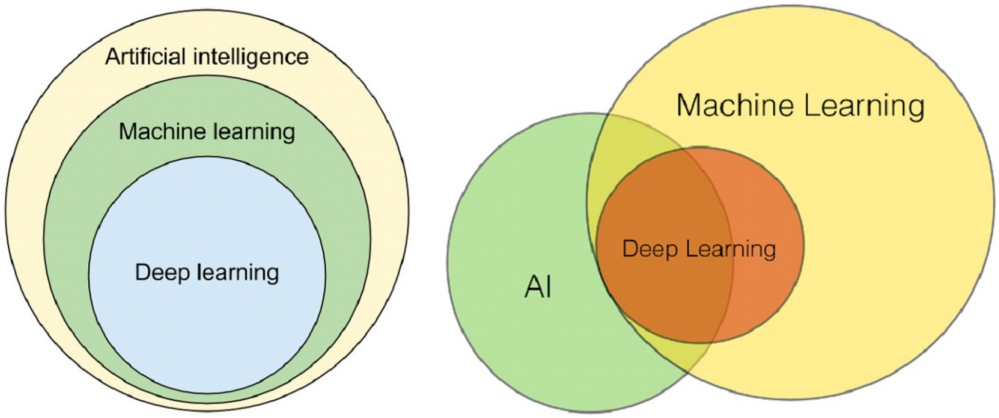

Figure 4.2. Due diversi modi di vedere la relazione tra Machine Learning e Artficial Intelligence. Credit to [28] e https://github.com/rasbt/stat453-deep-learning-ss20

4.1.1 I vari approcci

Come detto il machine learning raccoglie un gran numero di metodologie differenti, in particolare di fondamentale importanza sono i dati forniti al modello costruito, che permettono a quest'ultimo di imparare e migliorare. Di fondamentale importanza risulta quindi essere il concetto di training dataset, l'insieme di dati che viene utilizzato, in maniera diversa a seconda dell'algoritmo scelto, per effettuare l'allenamento del modello. Il training è un processo ripetitivo, in cui i dati vengono analizzati ripetutamente, si può quindi dedurre l'importanza della composizione di questo training dataset, infatti oltre a dover essere di una grandezza minima che permetta al sistema di poter funzionare adeguatamente, sarebbe ideale se appartenesse alla stessa distribuzione di probabilità dei dati reali a cui poi il modello andrà ad essere applicato. E' infatti vero che un modello adeguatamente robusto deve poter compiere la propria task in maniera adeguata anche se si cambia il set di dati sottostante, ma è altrettanto vero che operazioni complesse come il domain shift non sono per nulla immediate e anzi sono permesse solamente da algoritmi molto recenti.

Concetti importanti da introdurre prima di trattare più nello specifico vari

algoritmi e approcci, sono quelli di bias e varianza, due tipi di errori diversi che vanno a caratterizzare i modelli. In breve, il bias è la differenza tra la previsione media effettuata dal nostro modello e invece il valore attuale del dato che stiamo analizzando, rappresenta un errore dipendente dal modello costruito, che per esempio tende a semplificare troppo il modello senza "ascoltare" abbastanza i dati in input, fornendo quindi un errore elevato sia in fase di training che di test. La varianza va a evidenziare invece la variabilità di un modello rispetto alla distribuzione dei dati di training, ossia va a identificare l'errore di un modello che reagisce eccessivamente ai dati di allenamento, comportandosi in maniera molto diversa in caso si cambi il set utilizzato. Questo significa che il modello non generalizzerà, non permettendogli di performare bene su input che non ha mai visto, come avviene nella fase di test. Si parla quindi di bias-variance tradeoff come di un equilibrio che va mantenuto tra questi due diversi tipi di errore, in quanto alzando uno dei due si tende ad abbassare l'altro, ma è necessario trovare il corretto equilibrio per poter permettere al modello di generalizzare nel modo migliore possibile; questa armonia è uno dei principali fattori da considerare quando si va a scegliere il modello da utilizzare per un dato problema. In particolare, la presenza di questi due tipi di errori può portare a due diversi fenomeni che non permettono al modello di performare nella maniera migliore:

- L'overfitting, si presenta quando il modello non generalizza abbastanza, ma anzi si lega eccessivamente ai dati di training, creando nel corso dell'addestramento un modello troppo complesso, che va a considerare feature molto specifiche dei dati in ingresso. In questi casi ad esempio si fa fatica ad identificare gli outlier o il rumore nell'input, questo non permette al modello di analizzare in maniera corretta dei dati che non ha mai visto prima; in questi casi troviamo un alta varianza e quindi un errore molto basso nella fase di training, che cresce però in maniera vertiginosa in fase di test.

- L'underfitting è invece il fenomeno opposto, si presenta quando un modello statistico non riesce a catturare in maniera adeguata quella che è la struttura effettiva dei dati di input, approssimando eccessivamente il

modello che risulterà essere troppo semplice, un caso diffuso è l'applicazione di un modello lineare a input che però sono non lineari e quindi impossibili da descrivere per esso. In questi casi ci troviamo di fronte ad un bias eccessivo e quindi errori alti sia in fase di training che di test.

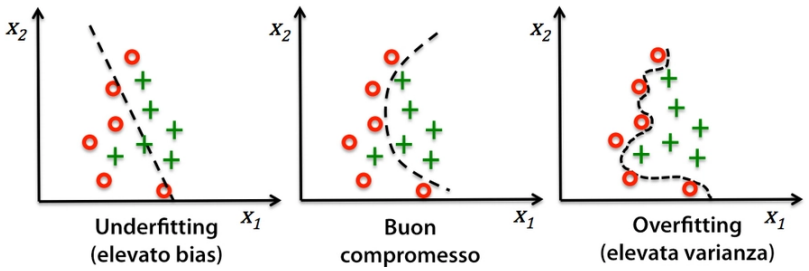

Figure 4.3. Un esempio grafico di underffiting, ovefitting e un buon compromesso tra bias e varianza che porta ad un buon risultato di classificazione. Credito to https://bit.ly/2NoudM3

A questo punto possiamo andare a vedere come a seconda dei dati in ingresso all'algoritmo e al tipo di feedback che è disponibile per lo stesso, si possono riconoscere varie tipologie di approccio al machine learning:

- Supervised learning (apprendimento supervisionato), è un approccio che si caratterizza principalmente per l'utilizzo di dati di training che sono etichettati, ossia che presentano non solo il segnale che sarà utilizzato come input dell'algoritmo, ma anche l'output desiderato. Questo meccanismo permette di utilizzare una funzione che mappi l'input fornito ad un dato output, avendo dati "labelled" la funzione può essere ottimizzata finché non fornisce l'output desiderato in sempre un maggior numero di casi. Lo scopo è quindi migliorare l'algoritmo al punto che generalizzi permettendo di fornire un output adeguato anche quando andrà ad analizzare dati che non ha mai visto, che siano essi il set di test oppure un'applicazione nel mondo reale. L'errore di questi algoritmi è identificato come "generalization error" ed è quello che si mira ad abbassare il più possibile anche seguendo il principio di bias-variance tradeoff. Vi sono un gran numero di algoritmi in questa categoria che

appartengono principalmente a task di classificazione, ossia si cerca di assegnare ad ogni sample la corretta categoria, e regressione, si lavora con un dominio continuo andando a stimare il valore di output. Decidere l'algoritmo da utilizzare in una specifica situazione è spesso una sfida in sé, in quanto non esiste un singolo algoritmo che performi meglio su qualunque problema di apprendimento supervisionato ("No free lunch theorem"), gran parte del peso della scelta ricade sulla tipologia e numerosità dei dati di training, nonché sulla dimensionalità dello spazio di input, che possono far preferire modelli più semplici o più complessi a seconda della situazione.

- Unsupervised learning (apprendimento non supervisionato), questo approccio, come si può immaginare, si basa invece su dati che non hanno ad essi collegati il corretto output. Gli algoritmi appartenenti a questo approccio hanno un diverso scopo rispetto ai precedenti, si dice che essi siano data-driven, in quanto la struttura dei dati è al centro della loro analisi. Cercano, infatti, di scoprire pattern nascosti nell'insieme in input o la struttura sottostante ai dati con le annesse connessioni tra questi. La conoscenza dei dati forniti al modello è quindi spesso il fine ultimo di questi studi, si può voler etichettare i dati in maniera approssimativa, dividerli secondo delle specifiche feature o semplicemente analizzarli in maniera migliore. I tipi di apprendimento non supervisionato più diffuso sono il clustering, che mira a scoprire gruppi di sample con feature simili, la density estimation, che invece cerca di meglio comprendere come sono distribuiti i dati nello spazio in esame, e la dimensionality reduction, un'insieme di tecniche che cercano di trasformare i dati da uno spazio a molte dimensione ad uno che ne contiene di meno.

- Reinforcement learning (apprendimento per rinforzo) è un approccio ancora differente che non si basa tanto su insiemi di dati, ma su un agente intelligente che si muove all'interno di un ambiente e che impara dai propri errori. L'agente infatti cerca di effettuare delle azioni all'interno di questo ambiente cercando di massimizzare il reward cumulativo che può raggiungere, solitamente infatti ad un'azione corrisponde un reward,

che in ambienti più complessi non è immediato, ma potenziale e che può essere identificato solo alla fine del percorso. Ci troviamo in un contesto in cui si utilizzano tecniche di programmazione dinamica e l'ambiente è spesso modellato secondo il processo decisionale di Markov. Si tratta quindi di un apprendimento guidato dagli errori, dove l'agente deve riuscire a bilanciare nel modo migliore esperienza dei passati tentavi ed esplorazione di nuove possibilità che sembrano poter indicare alte possibilità di guadagno in termini di premio cumulativo. Molto famosi sono gli esperimenti in questo ambito in cui si aiutano i computer ad imparare a completare i vecchi giochi platform.

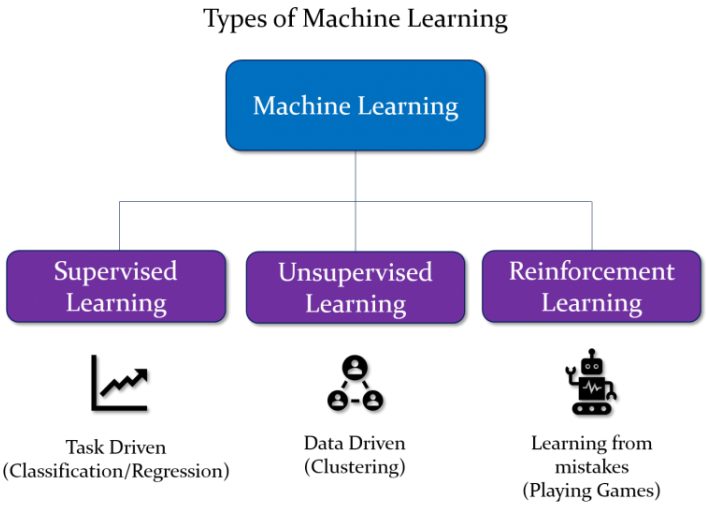

Figure 4.4. I tre principali approcci del Machine Learning. Credit to https://bit.ly/38LoFCA

Questi sono gli approcci principali a cui la letteratura si riconduce in maniera più frequente, in realtà con il passare del tempo ci sono molte altre sotto branche che sono nate e che hanno rivelato produrre ottimi risultati riguardo problemi o situazioni particolari. Ci sono ad esempio casi in cui vi è la disponibilità di una grande quantità di dati, ma solo pochi di questi risultato essere etichettati, questo sarebbe un grosso impedimento per un modello complesso che si basa sulla supervised learning poiché lo porterebbe di sicuro ad una situazione di overfitting; proprio in questi casi può essere utile

utilizzare tecniche di semi-supervised learning che sono state anche definite di supervisione debole ("weak supervision") [29]. Queste tecniche permettono di reperire set di dati molto grandi in formato "raw" e quindi non etichettati manualmente dall'essere umano e usarli in combinazione con una piccola quantità di labelled data per aumentarne l'efficacia, per fare ciò in molte situazioni si cerca di combinare il clustering con vari tipi di classificatori [30] per ottenere il risultato migliore possibile. Altro approccio che sta prendendo molta importanza è il Multi-Task Learning in cui invece di analizzare una sola task e ottimizzare un unico algoritmo in quella direzione, si analizzano i segnali di training di altre task ad essa correlate, per aiutare il nostro modello a generalizzare meglio. Un modo semplice per capire se siamo in presenza di Multi-Task Learning è osservare se si stanno ottimizzando più funzioni di Loss, in questo caso infatti vuol dire che stiamo osservando più problemi che però hanno delle caratteristiche in comune e che condividendo delle rappresentazioni tra queste task potremmo trarne dei benefici sui singoli compiti [31]. Questo concetto risulterà sicuramente più chiaro quando si presenteranno degli esempi pratici più avanti. Oltre alle varie tipologie di approccio al problema, un elemento che caratterizza il machine learning sono i modelli che devono essere allenati su dei dati di training per poi essere pronti a processare dati nuovi. Esistono vari tipi di modelli con caratteristiche e punti di forza differenti, dai Support Vector Machines ai Decision Trees, dalle reti bayesiane alla semplice linear regression; la trattazione dei vari modelli però esula da questa tesi e per questo verrà trattato nello specifico unicamente il modello che è stato utilizzato nel progetto, ossia le reti neurali artificiali spesso abbreviate come NN (neural network).

4.2 Le reti neurali artificiali

Come si è detto lo scopo dell'intelligenza artificiale è quello di produrre un sistema che possa avere comportamento e capacità assimilabili a quelle di un essere pensante. Per raggiungere questo scopo è sembrato quindi un ottimo inizio provare a riprodurre l'effettiva struttura del cervello umano, per tentare in questo modo di replicare i processi decisionali e di apprendimento. Da questa intuizione nasce il concetto di rete neurale artificiale, di cui in questo paragrafo si darà una breve panoramica. Essendo una rappresentazione del cervello umano si può immaginare come l'elemento costitutivo base sia il neurone. I vari neuroni sono disposti in colonne chiamate layer e sono interconnessi a neuroni del layer precedente e successivo tramite dei collegamenti che su di essi hanno dei pesi; tutta questa struttura fa a formare la cosiddetta rete neurale. I dati entrano in input ai neuroni e scorrono nella rete da un layer all'altro fino a che si produce l'output finale, ogni neurone reagisce ad una frazione dell'input in maniera diversa a seconda dei pesi a cui è soggetto, questa operazione è chiamata "forward pass".

Andiamo a capire meglio come è strutturato un neurone analizzando l'immagine qui presentata 4.5.

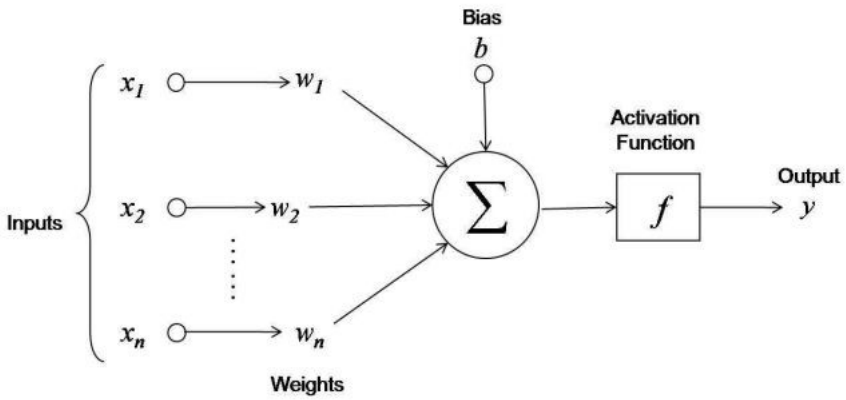

Figure 4.5. La struttura standard di un neurone.
Credit to https://bit.ly/3lnMOVa

Come si può osservare ogni neurone compie azioni molto semplici, nell'immagine

gli input che arrivano al neurone sono rappresentati come x (x1, x2, ... , xn), questi input sono ogni volta moltiplicati per i vari pesi o "weights" (w1, w2, ..., wn) che collegano le fonti di input al neurone vero e proprio. Questi pesi sono molto importanti, poiché mentre la struttura della rete viene decisa ad inizio processo di apprendimento e non viene cambiata, i pesi sono i valori che vengono modificati nel corso del processo e che determinano il cambiare dell'output finale. A questo punto gli input trasformati arrivano tutti al neurone che li somma e aggiunge ad essi un valore di bias scelto a priori. A questo punto al valore viene applicata una funzione, chiamata funzione di attivazione. Si tratta di una famiglia di funzioni con diverse caratteristiche, che però hanno lo scopo di portare il valore tra 0 e 1 (a volte tra -1 e 1) evitando una serie di problemi in fase di apprendimento, le funzioni di attivazione hanno inoltre l'importantissimo compito di introdurre non linearità all'interno della rete, permettendo quindi a quest'ultima di imparare pattern più complessi all'interno dei dati. Ne esistono diverse dalla funzione sigmoidea, tra le prime essere usate, alla ReLU, molto utilizzata, fino a funzioni derivate e più complesse come la ELU (Exponential Linear Unit) che però spesso richiedono operazioni dispendiose come in questo caso l'esponenziale. Da queste funzioni esce poi l'output che sarà a sua volta uno dei dati in ingresso per i neuroni successivi. La forma più semplice di rete è il Perceptron, un classificatore binario formato da un solo neurone, quindi con una struttura estremamente semplice, senza nemmeno la presenza della funzione di attivazione. Il passo successivo è il Multilayer Perceptron o MLP, un insieme di perceptron organizzati in più layer, qui possiamo per la prima volta vedere la divisione tra i vari layer tra input layer, lo strato di provenienza dei dati, output layer, lo strato che fornisce il risultato, e uno o più hidden layer, gli stati intermedi attraverso cui i dati subiscono le trasformazioni. Per passare dal MLP alle effettive reti neurali vanno aggiunte alcune caratteristiche. Come detto le funzioni di attivazione, non presenti nel perceptron, e una serie di iperparametri, ossia parametri che vengono utilizzati per controllare e migliorare il processo di apprendimento; essi vengono infatti ottimizzati nel corso del training per trovare la sistemazione della rete che performi in

maniera ottimale. Gli iperparametri sono molto vari, ma come suggerito da [32] possiamo dividerli tra iperparametri legati alla struttura stessa della rete e quelli legati all'algoritmo di training. Riguardo la struttura della rete possiamo citare le già analizzate activation function; il numero di hidden layers che si vogliono inserire per rendere la rete più o meno complessa; il dropout, una tecnica che disattiva i singoli neuroni secondo una percentuale stabilita, permettendo di trovare ad ogni forward pass una rete diversa da quello precedente, è quindi un'ottima tecnica di regolarizzazione che tende ad evitare l'overfitting; l'inizializzazione dei pesi, che determina l'evoluzione successiva della rete, ci sono vari modi di farla tra questi uno dei più famosi è il cosiddetto Xavier initialization. Per quanto riguarda gli iperparametri legati all'algoritmo, sono davvero tanti e sono quelli su cui si fa maggiormente tuning, ossia si prova la rete con diverse conformazioni approfondendo (fine-tuning) quelle che dopo poche iterazioni danno indicazione di performare meglio. Citiamo quindi principalmente il learning rate (che determina la velocità di apprendimento nel percorso di ricerca dell'ottimo), l'algoritmo di ottimizzazione e la decisione sui vari valori di epoche, iterazioni e grandezza del batch. La caratteristica principale che caratterizza le reti neurali e permette effettivamente di andare ad ottimizzare i pesi della rete è il processo chiamato backpropagation, di cui si chiariranno i passaggi principali senza soffermarsi troppo sui dettagli matematici. Come abbiamo visto la predizione da parte della rete viene effettuata grazie al forward pass, in cui i dati di input passano attraverso i vari neuroni e vengono trasformati, fino ad arrivare all'output layer che produce una predizione. Una volta arrivati a questo punto dobbiamo ricordarci come siamo nel caso di apprendimento supervisionato, abbiamo quindi a disposizione l'output corretto; per questo possiamo confrontare il risultato atteso con quello predetto per mezzo di una funzione di costo o di errore. Lo scopo della backpropagation è proprio minimizzare questa perdita da parte della rete e per farlo l'unico modo è andare a variare i pesi della rete; andare a calcolare singolarmente qual è il peso con una maggiore perdita è però altamente inefficiente e oltretutto potrebbe rivelarsi errato, per questo motivo si va ad utilizzare questo metodo. Come

sottolineano Rumelhart et al., nell'ormai lontano 1988 [33], la backpropagation "regola ripetutamente i pesi delle connessioni nella rete in modo da minimizzare la misura della differenza tra il vettore di uscita effettivo della rete e il vettore di uscita desiderato" inoltre "la capacità di distinguere nuove features utili distingue la backpropagation dai metodi precedenti e più semplici". A livello pratico, una volta trovato l'errore si fa la derivata parziale della funzione di costo rispetto agli input, usando quella che viene chiamata "chain rule"; questa derivata parziale che è detta gradiente, viene propagata nel verso opposto della rete raggiungendo tutti i neuroni, i cui pesi sono aggiornati calcolando le varie derivate parziali locali, parliamo infatti di "backpropagation with gradient descent".

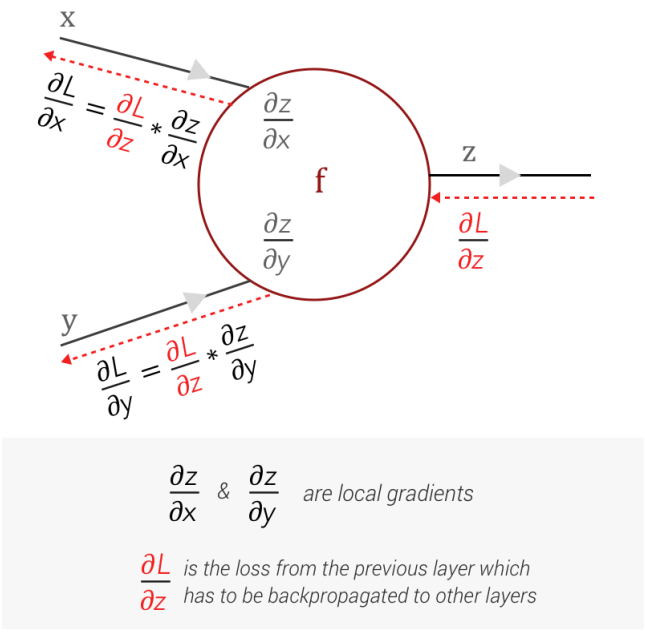

Figure 4.6. Schematizzazione di come viene effettuata Backpropagation with gradient descent. Credit to https://bit.ly/3rUBCS6

Si è quindi ora fatta una breve panoramica sui concetti alla base del funzionamento delle reti neurali, possiamo perciò andare più nello specifico di quello che sono le tecnologie specifiche utilizzate in questo lavoro. Partiamo con il dire che negli ultimi anni sono stati esaminati problemi sempre più complessi che quindi richiedono una maggiore complessità della rete stessa

per ottenere risultati accettabili. E' in questo momento, con la creazione di reti più profonde e quindi con un maggior numero di layers, che si comincia a parlare di Deep Learning, una branca che comprende un gran numero di tecnologie, gran parte delle quali basate su reti neurali, che permettono di estrapolare un numero di feature più elevato e di livello più alto rispetto ai loro predecessori. A lanciare questo trend è stata senz'altro AlexNet [34] con la sua vittoria nell' "ImageNet Large Scale Visual Recognition Challenge" (ILSVRC) del 2012, che ha mostrato l'efficacia delle reti deep rispetto a quelle shallow, nonché il grande potenziale delle reti convoluzionali. Questo trend è proseguito in maniera discreta negli anni successivi per poi esplodere con l'avvento di ResNet [35] che grazie all'invenzione dei "residual block" ha permesso di allenare reti nettamente più profonde dei loro predecessori, evitando il problema del "vanishing gradient" che nel corso della propagazione del gradiente provocava la sua attenuazione passando da un layer all'altro, impedendo l'allenamento della rete.

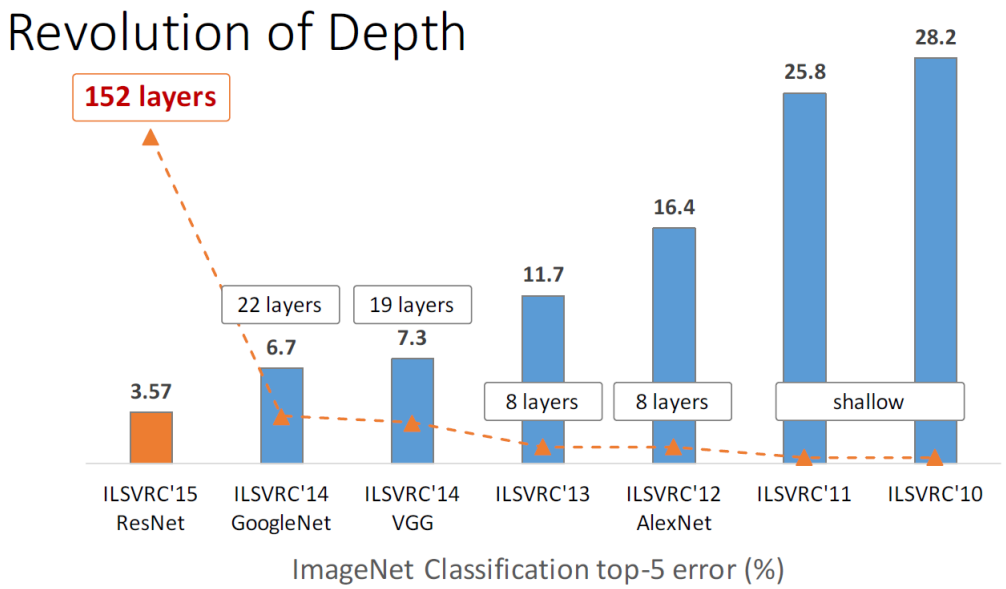

Figure 4.7. Risultati della ILSRVC in 5 anni consecutivi che dimostrano la rivoluzione apportata prima da AlexNet e poi da ResNet. Credit to https://bit.ly/3vtzUth slide 6

Si è detto che AlexNet è stata la prima Deep Convolutional Neural Network a performare in maniera migliore rispetto alle reti neurali classiche, andiamo quindi a vedere cos'è una rete neurale convoluzionale.

4.2.1 Le reti neurali convoluzionali

Una importante caratteristica del Deep Learning, oltre alla maggiore profondità delle reti, è la possibilità di uscire dal modello classico del connessionismo, che pone alla base i neuroni, e poter creare layer eterogenei e di diversa natura a seconda dell'applicazione o del tipo di dati in input. In questa ottica nascono le reti neurali convoluzionali, introducendo dei nuovi tipi di layer, detti appunto convoluzionali, che permettono un'analisi più efficace quando si parla di immagini. Andando con ordine si può osservare come anche in questo caso l'invenzione si è basata sull'osservazione e lo studio di processi biologici, in particolare provando a utilizzare i neuroni nello stesso modo in cui funziona la corteccia visiva di alcuni animali [36] da cui deriva il concetto di "Neocognitron", predecessore dei layer convoluzionali. Come visto precedentemente nel Multilayer Perceptron abbiamo una serie di layer detti "fully connected", questo significa che ogni neurone era collegato a tutti i neuroni dello strato precedente e a tutti quelli dello strato successivo; questo tipo di struttura oltre a essere soggetta a overfitting, nel caso di immagini complesse non riusciva a catturare le dipendenze tra i pixel all'interno dell'immagine. Per ovviare a questo problema entrano in gioco le reti convoluzionali, come si è detto ci si basa sul funzionamento della corteccia visiva dove i singoli neuroni corticali rispondono agli stimoli che provengono da un'area ristretta del campo visivo dell'essere vivente, quest'area è detta campo recettivo del neurone; i vari campi recettivi, anche sovrapponendosi, vanno così a coprire l'intero campo visivo dell'organismo. Questo stesso meccaniscm, che va contro il concetto di fully connected layer, è ripreso anche negli strati convoluzionali, infatti il neurone del layer n riceve gli input solo da una ristretta area del layer n-1, che anche in questo caso prende il nome di campo recettivo del neurone. Lo scopo è ridurre l'immagine ad una forma che sia facile da processare, rispetto all'analizzare tutti i pixel individualmente,

ma che non faccia perdere le features essenziali. Per fare questo nei layer convoluzionali si applica un filtro all'immagine [37], questo filtro è una matrice di grandezza fissa che scorre sull'immagine con un certo passo o "stride". Quando il filtro si muove, viene fatta una moltiplicazione fra matrici fra il filtro e la porzione dell'immagine sotto di esso, da ogni moltiplicazione si ottiene come risultato un numero che andrà a formare la successiva rappresentazione dell'immagine anche detta "activation map". Questo processo viene ripetuto finchè tutta l'immagine non viene analizzata. È bene inoltre ricordare che un'immagine ha un vario numero di canali, come nel caso delle RGB dove troviamo 3 canali, in questo caso il filtro deve avere la stessa profondità dell'immagine che va ad analizzare. Quest'operazione è quella che viene chiamata Convoluzione e da cui prendono nome i layer, il suo scopo è quello di estrarre le feature di alto livello dall'immagine di partenza; in particolare come visto ci troviamo in presenza di reti profonde e più si va in profondità con gli strati più si vanno ad analizzare feature di alto livello. I primi layer convoluzionali estraggono le feature di livello più alto e più generiche come ad esempio i bordi, per poi passare a feature sempre più di alto livello man mano che si aumentano il numero dei layer. Un'altra tipologia di layer che le reti convoluzionali introducono sono i pooling layer, questi sono responsabili di ridurre la grandezza delle activation map, ossia delle feature che sono state già convolute dai convolutional layer; questi permettono di rendere le features molto più gestibili, aumentando enormemente il carico computazionale della rete senza però perdere del contenuto informativo, anzi spesso aiutano a estrarre le feature dominanti dalle mappe precedenti. Essendo una riduzione dei valori ottenuti, vi sono vari modi di fare pooling, ma il più usato è solitamente il max pooling, che prende il valore massimo della specifica area interessata dal filtro, permettendo in alcuni casi in questo modo di ignorare del rumore che non sia eccessivamente forte. Una serie di layer convoluzionali e pooling layer, uniti da eventuali strati utilizzati per la batch normalization [38] che permette risultati migliori e più efficienti normalizzando i dati tra uno strato convoluzonale e l'altro, permettono di effettuare quella task chiamata feature extraction, che prima del deep learning

era spesso fatta ad hoc dall'uomo per poter fornire i dati nella maniera più consona al classificatore. A questo punto si deve però classificare l'immagine, per questo l'activation map finale ottenuta viene spianata e data in pasto ad una rete neurale classica che funzionerà da classificatore, su però una serie di dati gestibili molto più facilmente. L'immagine seguente può riassumere bene la struttura base di questo tipo di reti che negli ultimi anni, dall'avvento di AlexNet in poi, hanno ottenuto grandissimi risultati diventando la nuova baseline per quanto riguarda l'ambito della Computer Vision, una disciplina che si pone come scopo quello di permettere ai computer di ottenere una comprensione di alto livello di contenuti multimediali, siano essi immagini o video, arrivando a migliorare ciò che il sistema visivo umano può fare [39].

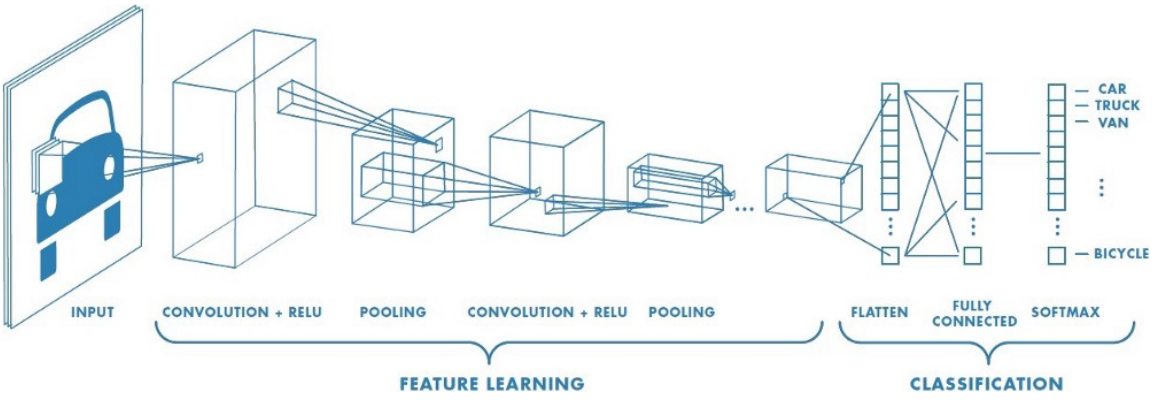

Figure 4.8. Schematizzazione di una Convolutional Neural Network, nell'esempio per la classificazione di veicoli. Credit to https://bit.ly/3cCEug3

4.3 Alcune Task significative in ambito Computer Vision

Il concetto di rete convoluzionale è alla base del lavoro che si è compiuto in questa parte finale del lavoro di tesi. Come ampiamente detto, infatti, gli esperimenti si basano sul riconoscere le emozioni, rispettivamente nel paziente o nel partecipante, per poi poter trarre delle conclusioni che siano dal punto di vista neurologico in ottica riabilitativa, oppure per quanto riguarda il loro effetto in ambito neuromarketing. Per fare ciò, dopo la spiegazione del funzionamento di base di una rete convoluzionale risulta evidente come utilizzando delle tecnologie da essa derivate si possa andare a provare a classificare le emozioni identificate attraverso i frame di video del partecipante catturati durante gli esperimenti. Per arrivare alla fase di emotion recognition e quindi alla classificazione delle emozioni identificate, bisogna passare attraverso al conosce di altre task del machine learning più specifiche, che si basano principalmente sull'utilizzo di reti neurali convoluzionali con adattamenti specifici a seconda dell'ambito di utilizzo.

4.3.1 Semantic Segmentation

L'imagine segmentation può essere pensata come un raffinamento nel concetto di classificazione di immagini. Da quello che abbiamo detto ora, se viene fornita un'immagine alla rete questa cercherà di classificare l'immagine, quindi ad esempio se individua un gatto in un prato, una rete funzionante identificherà il gatto e gli assegnerà l'etichetta corretta. Non abbiamo però in questo modo nessuna informazione né spaziale, né in molti casi quantitativa, sull'elemento che viene riconosciuto. Spesso si necessita di una precisione migliore, entra quindi in gioco la segmentazione semantica o appunto "semantic segmentation". Questa tecnica va ad assegnare una specifica etichetta non all'immagine nel suo insieme, ma ai singoli pixel che la compongono; si chiama infatti semantica poiché i pixel vengono assegnati ad una categoria senza curarsi delle istanze dei singoli oggetti, ma identificando i confini tra una categoria di oggetti e l'altra.

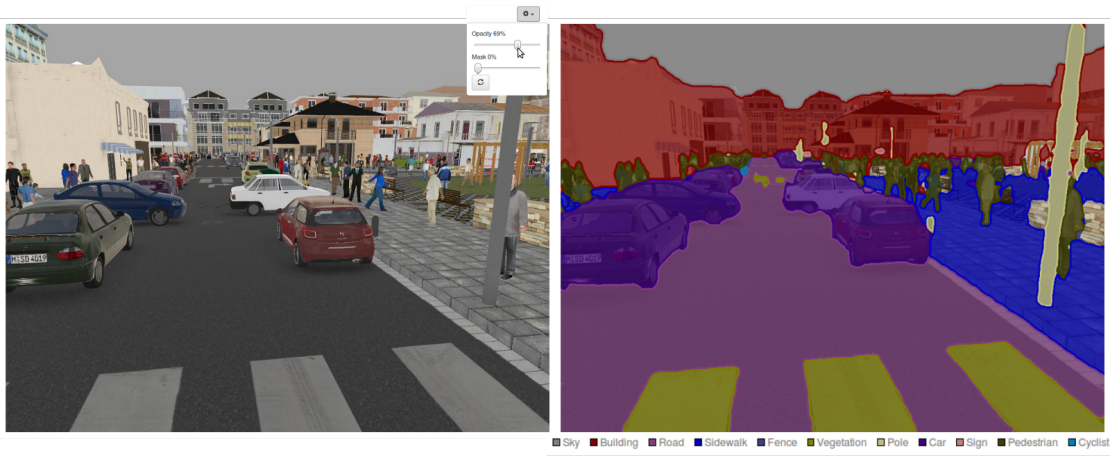

Figure 4.9. Un esempio di Semantic Segmentation applicata in ambito automotive. Credit to https://bit.ly/3rSfFDl

Come si può osservare nell'immagine ai vari pixel viene assegnata una categoria specifica, colorandoli con un dato colore a cui è collegata una specifica etichetta; per questo tutte le auto sono indifferentemente colorate con il colore viola, indipendentemente se i pixel appartengano a due oggetti differenti. Questa tecnologia può rivelarsi di grande utilità e sta infatti trovando numerosissime applicazioni, come ad esempio per quanto riguarda l'analisi di immagini mediche in cui è importante il riconoscimento e la divisione dei vari organi o tessuti, ma soprattutto nell'ambito della guida autonoma, in cui il riconoscimento della strada, unita alle altre macchine o agli ostacoli di vario genere è di importanza primaria per la riuscita della guida del veicolo. Per fare ciò si usano una rete con una struttura particolare divisa principalmente in due parti: un encoder e un decoder. L'encoder è una classica convolutional neural network utilizzata come estrattore di feature che va ovviamente a fare un'operazione di downsampling sull'immagine in input, in quanto, come già spiegato in precedenza, vengono prese le feature più importanti dall'immagine che viene trasformata in una forma più facilmente gestibile, all'uscita dall'encoder avremmo quindi un vettore che codifica le probabilità delle classi. A questo punto dobbiamo tornare all'immagine di partenza in

cui però tutti i pixel sono associati ad una specifica classe, operando nel decoder quindi una funzione di upsampling, passando da una rappresentazione a dimensionalità più bassa ad una più alta. Per fare questo il decoder è strutturato anch'esso come una CNN, ma potremmo dire costruita a specchio rispetto all'encoder, la rete nel suo insieme è quindi detta Fully Convolutional Network [40] e come output restituisce una segmentation mask. Il punto da approfondire leggermente è però come fare upsampling, poiché partiamo da una rappresentazione sparsa dell'immagine e dobbiamo aumentare i dati presenti. Ci sono vari modi di fare upsampling, alcuni molto simili al concetto di pooling, ma applicati in maniera inversa, metodi che prenderanno il nome di "unpooling". Per esempio si possono prendere le informazioni a bassa risoluzione e utilizzare la singola informazione per riempire gli spazi vicini (nearest neighbor unpooling), oppure ricordare la posizione del massimo in caso di max pooling e scrivere l'informazione a bassa risoluzione nella posizione del precedente massimo (max pooling indices). Uno dei più usati è il metodo chiamato "Transopose convolution", esso si basa sullo stesso concetto dei filtri dei layer convoluzionali, ma in questo caso il filtro scorre sull'input che è più piccolo, con un passo minore rispetto alla rappresentazione di output, su cui viene trasmessa l'informazione e che si va quindi a riempire; la problematica riguarda i punti in cui i filtri si sovrappongono nei quali si possono adottare diverse soluzioni come sommare i pixel, ma si cerca di insegnare alla rete ad evitare queste situazioni adattando il kernel. Adesso è quindi chiaro concettualmente come avviene la segmentazione semantica dell'immagine. In molti casi però non ci serve assegnare una categoria ad ogni pixel, ma necessitiamo invece di localizzare indicativamente le zone in cui risiede uno specifico oggetto di interesse, in questo caso dovremo utilizzare un diverso tipo di tecnologia.

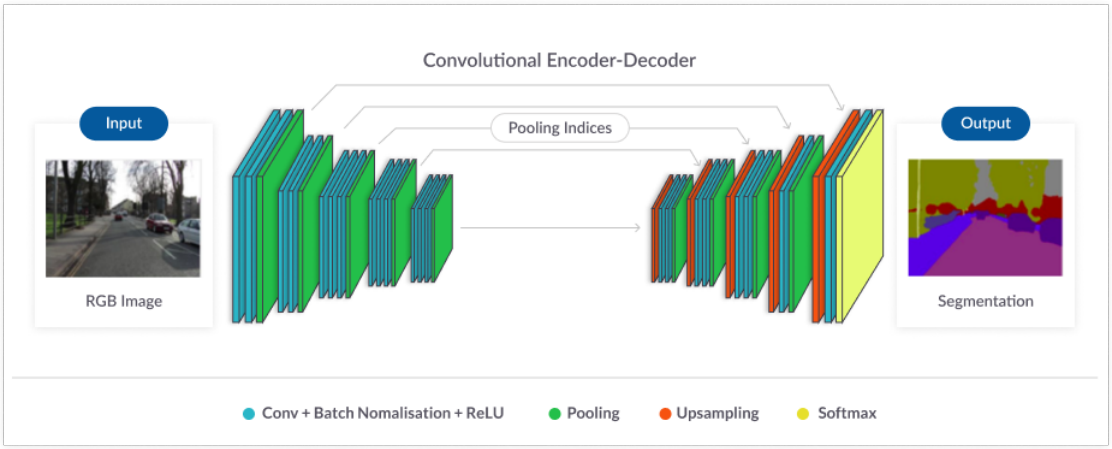

Figure 4.10. Struttura di una rete Fully Convolutional, nello specifico la struttura della rete SegNet. Credit to https://bit.ly/3tuw4OP

4.3.2 Object Detection

La specifica task di machine learning che permette di rispondere alle richieste presentate precedentemente è chiamata appunto object detection e si basa su due operazioni principali:

- Localization, si cerca nell'immagine proposta la posizione degli oggetti di interesse
- Classification, gli oggetti localizzati vengono analizzati dalla rete e classificati

Alla fine del processo quindi nell'immagine non verranno classificati tutti i pixel come avveniva precedentemente nella segmentazione semantica, ma verranno identificati gli oggetti singolarmente e identificati classificandoli. Una volta identificati gli oggetti classificati saranno rappresentati racchiusi in un bounding, ossia il più piccolo rettangolo trovato che racchiude l'oggetto identificato; solitamente questi bounding box sono identificati da 4 valori: le cordinate dell'angolo in alto a sinistra, altezza, larghezza e a volte anche l'incertezza del classificatore associato. Il modo più comune di effettuare object detection, consiste prima nell'ipotizzare la posizione di questi bounding boxes, poi effettuare dei resize per far si che comprendano l'oggetto in

questione completamente, infine inviare al classificatore il rettangolo di pixel identificato. Per questo motivo spesso fare object detection può rivelarsi un ottimo passo di preprocessing delle immagini, così da far concentrare il classificatore solo sugli oggetti da classificare e non su tutto l'input che potrebbe contenere una grande quantità di rumore.

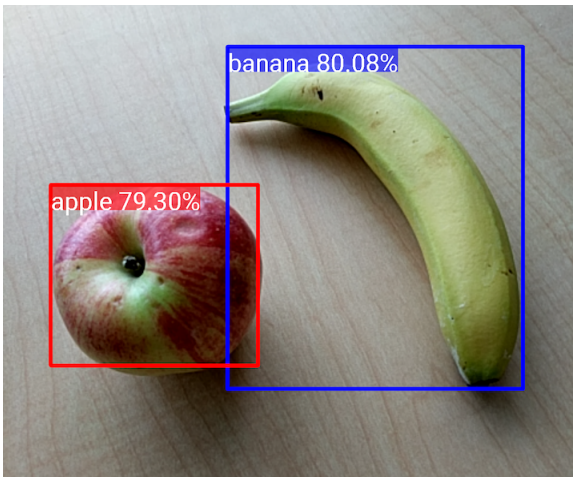

Figure 4.11. Struttura classica di bounding boxes per object detection con associata probabilità della classe assegnata.
Credit to https://bit.ly/3vxWiS4

Nel corso degli anni sono stati create una larga serie di tecniche per effettuare object detection, forse la prima che può essere identificata e che ha ottenuto ottimi risultati nell'estrarre le feature dei volti è stata quella utilizzata da Viola e Jones nel 2001 [41] che per effettuare la localizzazione di volti all'interno delle immagini utilizzavano delle Haar-features all'interno di un classificatore a cascata, come risultavano riuscivano a fare effettivamente face detection con buoni risultati, ma soprattutto in real-time caratteristiche questa che permetteva all'algoritmo di essere applicabile anche nel mondo reale per applicazioni quali ad esempio telecamere di sorveglianza. Al momento le tecnologie più utilizzate e studiate basate su reti neurali sono di tre tipi:

- Sliding window
- Region proposal CNN (R-CNN, Fast R-CNN, Faster R-CNN)

- You Only Look Once o YOLO
- SSD: Single-Shot MultiBox Detector

Il metodo basato sulle sliding window come dice il nome si basa sul concetto di una finestra che incornicia la porzione di immagine che si vuole analizzare e che andrà data in pasto al classificatore, questa finestra scorrerà su tutta l'immagine fornendo la possibilità di analizzare tutta l'immagine. Già così appare un metodo computazionalmente pesante, ma a questo bisogna aggiungere che la finestra non può essere una sola, ma va provata a diverse risoluzioni poiché non si conosce la grandezza dell'oggetto. Quindi se da un lato non dobbiamo allenare nessun'altro classificatore se non quello necessario a riconoscere l'oggetto, appare chiaro come questo metodo risulti eccessivamente lento e dispendioso. L'approccio successivo è quello delle region proposal o R-CNN, ossia si vogliono identificare delle regioni di interesse in cui guardare se effettivamente si identifica un oggetto di interesse. Solitamente si identificano zone in cui i pixel hanno colori o texture simili, con finestre di grandezze differenti, si cerca di identificare poche migliaia di regioni di interesse, di solito circa 2 mila, che saranno sempre molto di meno delle regioni analizzate nel metodo delle sliding windows. A questo punto la regione di interesse veniva data come input ad una CNN a cui poi era collegato in output un algoritmo di box regression, per stabilire i confini del bounding box, e una Support Vector Machine per la classificazione vera e propria, il risultato di entrambi viene gestito utilizzando una Multi-task Loss unione delle rispettive Loss (questo è un esempio del Multi-Task Learning descritto qualche paragrafo fa). Questi accorgimenti miglioravano la velocità di allenamento e predizione della rete, ma il processo in sé rimaneva ancora molto lento. Ciò che rendeva lento il processo era che la CNN era applicata a tutte le migliaia di regioni prese in considerazione, per questo si pensò che i calcoli fatti dalla CNN potessero essere condivisi. Siamo davanti a quelle che vengono chiamate Fast R-CNN, in cui l'intera immagine viene passata alla rete convoluzionale e poi sull'ultimo layer di questa, viene effettuata l'operazione di region proposal; a questo punto si deve solo riformattare la dimensione di queste regioni con uno "spatial pyramid pooling" per passarle nella taglia

corretta alla parte di classificazione. Quest'ultima subisce dei cambiamenti, si un softmax layer al posto della svm e un regression layer per le bounding box. L'algoritmo è nettamente più veloce di prima, fino a 9 volte, ma non è ancora real time che è il nostro obiettivo finale. Per velocizzare ancora di più queste reti si passa alle Faster R-CNN, queste sono una integrazione delle precedenti, si aggiunge infatti una Region Proposal Network che sarà quella a trovare le regioni di interesse, ma il suo lavoro sarà molto più veloce rispetto al precedente selective search, poiché condivide gran parte del calcolo con la rete addetta all'object detection. Il tutto si basa sul definire delle anchor box con differenti scale e proporzioni che verranno poi analizzate dalla Region Proposal Network che indicherà quelle con più probabilità di ottenere oggetti, questo riduce di molto le regioni da analizzare. Questo accorgimento velocizza ulteriormente il processo, rendendolo quasi real time, ma l'obiettivo non è ancora raggiunto. Questa slide del corso su "Segmentation e Object Detection" di Andreas Maier [https://bit.ly/3qNwaiN] fornisce un ottimo riassunto di queste 3 architetture appena descritte.

Per riuscire ad effettuare veramente object detection in real time dobbiamo arrivare alle utime due soluzioni, YOLO e SSD, questi due metodi condividono l'idea chiave di unire la predizione delle bounding box e la classificazione all'interno della stessa rete. Queste tecnologie saranno però trattate più approfonditamente nel prossimo capitolo, in cui si analizzeranno nello specifico le scelte operate nel progetto.

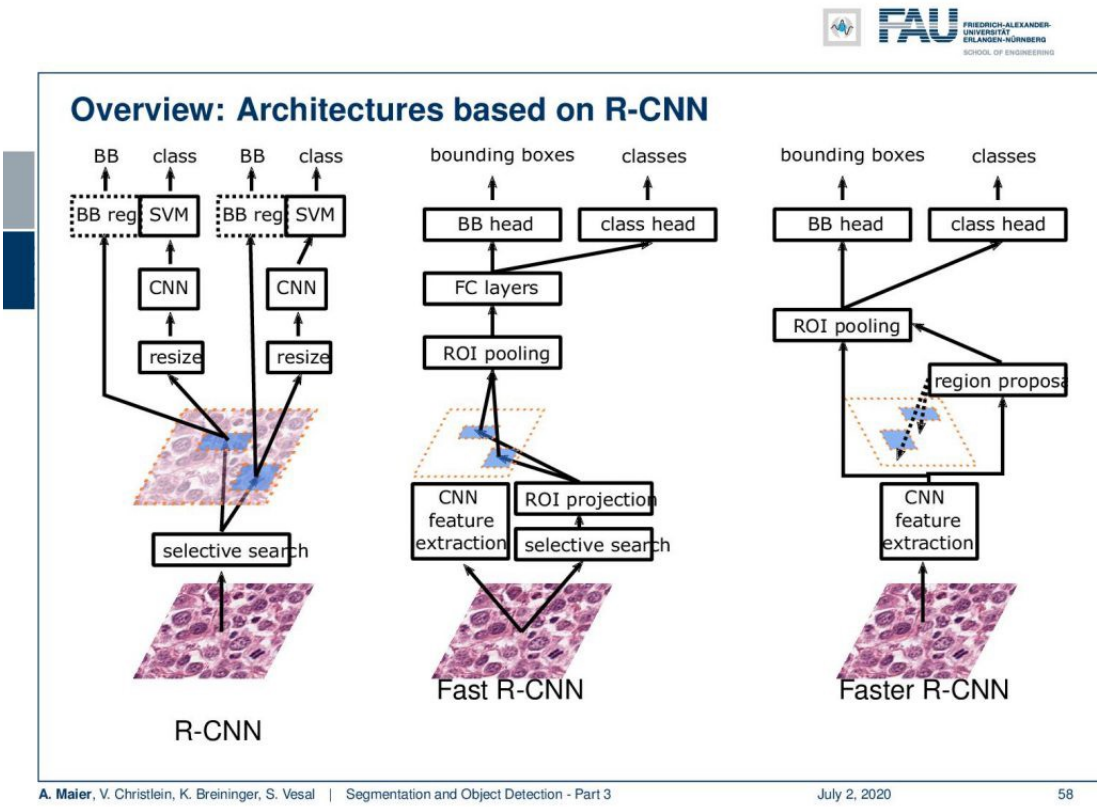

Figure 4.12. Slide del corso tenuto da Alex Maier in cui vengono riassunte le strutture di R-CNN, Fast R-CNN e Faster R-CNN. Credit to https://bit.ly/3qNwaiN

4.3.3 Instance Segmentation

Per completare la trattazione adesso parleremo di quella che può essere vista come la tecnologia più fine, in quanto va ad unire le caratteristiche delle precedenti metodologie presentate, l'Instance Segmentation. Stiamo parlando sempre di segmentazione, infatti sono i singoli pixel a ricevere un'etichetta, la differenza con la semantic segmentation è però che i pixel ad essere categorizzati sono solo quelli corrispondenti agli oggetti identificati, inoltre sono definiti i confini tra un oggetto e l'altro, in questo modo due automobili una dietro l'altra non saranno un'indistinta macchia di colore appartenente alla stessa categoria. Riassumendo si può dire che l'instance segmentation è il risultato dell'unione di object detection a cui viene applicata semantic

segmentation, gli oggetti sono infatti identificati singolarmente e successivamente vengono definiti i loro confini e i loro pixel etichettati singolarmente. Questa immagine può ben riassumere la differenza tra i vari metodi.

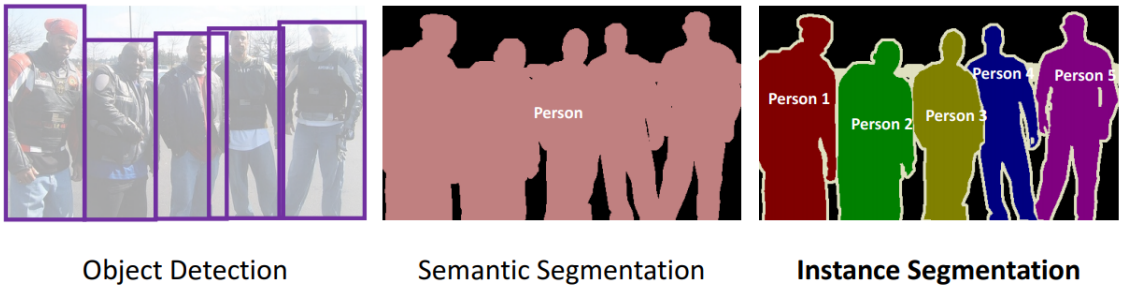

Figure 4.13. Riepilogo delle differenze di output tra Object Detection, Semantic Segmentation e Instance Segmentation.
Credit to https://bit.ly/3eLqmUG

A seguire questo concetto si trovano diverse reti in letteratura, ma quella di maggior successo è probabilmente la Mask R-CNN [42]. In questa rete si va a unire la funzione di object detection fornita da un Faster R-CNN, che permette di separare le istanze dei vari oggetti le une dalle altre e dopo su queste istanze viene applicata la segmentazione semantica, rifinendo di fatto i bordi della bounding box calcolata. La procedura si articola principalmente di due fasi, la prima è comune e in essa avviene la fase di region proposal con il calcolo delle regioni di interesse. Successivamente il processo si divide e la parte di classificazione con le bounding box e quella di segmentazione, vengono eseguite in parallelo. Anche in questo caso viene usata una multi-task loss, che però questa volta ha tre termini in quanto si va ad aggiungere anche la loss propria della segmentazione. Ovviamente come anche identificato nello stesso paper vi sono più modi di dividere questo processo a due fasi, sia a seconda del punto in cui avviene la divisione, sia a seconda della rete di backbone che viene scelta. Un'idea più chiara della struttura può essere fornita dall'immagine sottostante presa direttamente dal paper ufficiale di presentazione della Mask R-CNN.

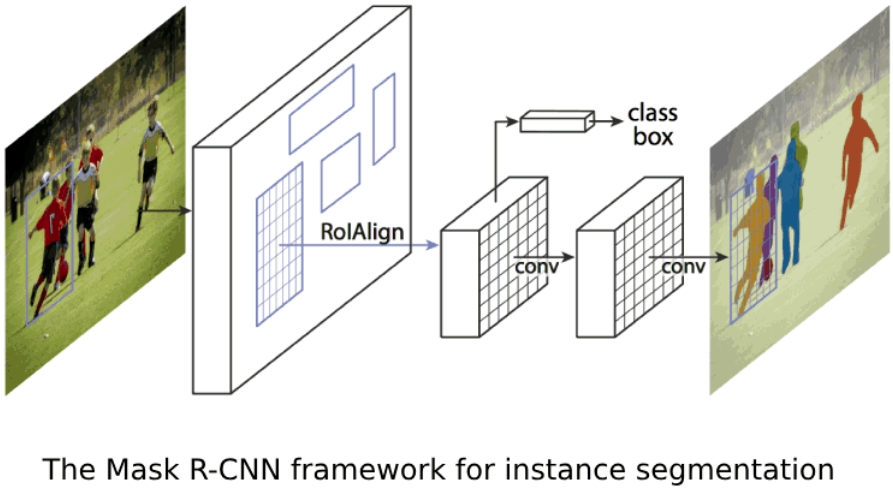

Figure 4.14. Semplificazione della struttura della Mask R-CNN direttamente dal paper ufficiale [42]

A questo punto abbiamo introdotto gran parte dei concetti cardine e che saranno analizzati e utilizzati nella pratica del progetto. L'ultimo passo è andare più nello specifico di quello che è effettivamente il nostro obiettivo.

Chapter 5

Le soluzioni adottate

Nel precedente capitolo abbiamo analizzato i concetti di base che permetteranno di comprendere in maniera più chiara gli argomenti trattati nei paragrafi seguenti. Verranno affrontate le scelte tecniche adottate, cercando di spiegare perché un algoritmo è stato preferito ad un altro e verranno approfondite le tecnologie specifiche che sono state utilizzate, anche se solo in una fase iniziale del lavoro, per questa seconda parte di progetto. Rispetto alla produzione della applicazione desktop dove il mio lavoro tecnico è stato principalmente affrontato in solitaria, con però costanti riunioni e feedback con i due team di ricerca; in questa seconda parte del mio lavoro di tesi, sono stato affiancato da un altro laureando del Politecnico di Torino e da un team di ragazzi che frequentano la magistrale presso la University of Essex, studenti del correlatore a capo del progetto, il professor Vito De Feo. Insieme abbiamo collaborato per cercare di dare vita a questo algoritmo di facial emotion recognition, curando in gruppi separati le varie parti del progetto, ma lavorando in stretto contatto durante tutta la durata di questo. Per ogni fase abbiamo potuto analizzare più di un algoritmo, confrontando risultati e scegliendo quello che forniva performance migliori o un carico di lavoro più adatto a quella che doveva essere la nostra applicazione. Cercherò di analizzare brevemente tutte le opzioni che abbiamo valutato, includendo la trattazione teorica comprendente i numerosi articoli che ci sono stati di

supporto, il tutto per arrivare ad un risultato che verrà sicuramente migliorato nei prossimi mesi, ma che è senz'altro un ottimo punto di inizio visto il poco tempo a disposizione.

5.1 Il punto di inizio

In questa sezione sarà analizzato quello che è stato il punto di partenza del nostro lavoro, ossia un precedente progetto, la ricerca di un buon dataset da cui poter partire e la lettura di molti paper per comprendere lo stato dell'arte e indirizzarci su quelle che saranno le tecnologie che abbiamo tentato di utilizzare e che saranno mostrate successivamente.

5.1.1 L'applicazione Vibe

Questo grande progetto di ricerca ha come predecessore un progetto più piccolo iniziato l'anno precedente da un gruppo di studenti della University of Essex sempre sotto il professor De Feo, che però collaboravano con unicamente con il centro neuriabilitativo Puzzle. Il gruppo aveva come obiettivo quello di creare un'applicazione che aiutasse i professionisti del centro con il loro lavoro riabilitativo, aiutando a verificare se il trattamento effettuato sui pazienti stesse procedendo nel senso corretto e se i risultati ottenuti fossero incoraggianti. Da questi obiettivi è nata Vibe, un'applicazione che permetteva all'utente di scattare una foto con la fotocamera del telefono e da essa derivare l'emozione che l'utente esprimeva con la sua espressione facciale. Potremmo dire che questa applicazione ha rappresentato la fase embrionale del nostro progetto, dalla cui idea noi siamo partiti e che abbiamo poi potuto modificare e ampliare. Per predire l'emozione data l'immagine dell'espressione facciale, i ragazzi hanno deciso di utilizzare il Deep Learning, in particolare le Convolutional Neural Network, visto il suo grande vantaggio di non necessitare di grandi passi di pre-processing in quanto non è l'uomo a gestire la feature extraction. Hanno verificato il funzionamento di diversi modelli conosciuti come VGG16 e LeNet-5, per poi passare al tentativo di creare delle reti neurali custom, per riuscire a ricavare dei risultati che ritenessero accettabili.

Dopo varie fasi di training e tuning degli iperparametri della rete, sono riusciti ad ottenere il miglior risultato di accuratezza sul dataset, di circa il 65%, tramite uno dei modelli da loro costruiti che riporto qui sotto.

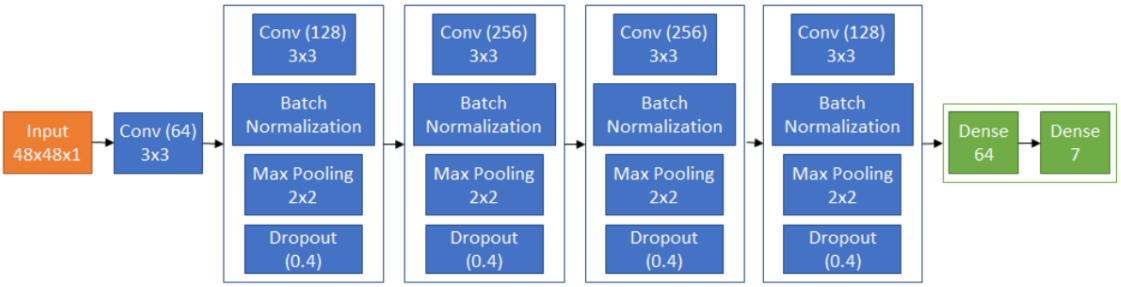

Figure 5.1. Rappresentazione a blocchi dei vari layer che compongono la CNN Custom utilizzata

Come descritto nella loro documentazione il modello è composto da 6 layer, di cui il primo è una layer convoluzionale con filtro 3x3 è che usa come funzione di attivazione una Relu. Dal secondo al quinto layer la struttura è identitica, un layer convoluzionale, batch normalization, max-pooling a cui poi viene applicato Dropout con rate 0.4. Mentre in tutti la funzione di attivazione e il filtro rimangono gli stessi, ciò che cambia è il numero di neuroni che, come si può osservare in figura, è rispettivamente 64-128-256-256-128. Per finire L'ultimo strato è composto da due dense layer, sempre comprendenti batch normalization e dropout, che terminano con una funzione di softmax che ha come output 7, ossia il numero delle categorie al quale l'immagine può essere associata, nonché il numero di emozioni che sono state selezionate come etichette nel dataset e che cercavano di racchiudere il gran numero di espressioni facciali prodotte dal viso umano: rabbia, disgusto, paura, felicità, tristezza, sorpresa e neutro. Il dataset è stato fornito da una challenge del famoso portale Kaggle e, nonostante alcuni errori di etichettatura, che però sono spesso riscontrati in questo ambito, è sembrato al gruppo il più affidabile e vicino al tipo di scopo per cui Vibe era stata pensata. Per giudicare i modelli si è utilizzata come metrica l'accuratezza, senza andare a considerare penalizzazioni particolari o lo sbilanciamento delle classi, questo sia perché

non vi erano state indicazioni più specifiche dal gruppo di ricerca, sia perché il dataset fornito risultava decentemente equilibrato. Recuperati report e codice di questa applicazione, soprattutto la parte riguardante il modello, abbiamo potuto usare questi risultati come baseline per mostrarli ai gruppi di ricerca e capire che cambiamenti effettuare. Ovviamente la prima cosa che risulta evidente si trova a livello applicativo, mentre in questa applicazione l'utente doveva scattarsi una foto in posa, nel nostro progetto era necessario riprendere il paziente durante l'effettuazione dell'esperimento. Questo voleva dire che per cominciare l'input era parzialmente differente, si passava da foto a video; questo non rappresentava un gran problema in quanto era comunque possibile classificare le immagini, facendo una parte di preprocessing e dividendo il video in frame. La vera differenza comportata da questo modo di raccogliere i dati è la naturalezza dell'espressione. In Vibe viene chiesto all'utente di mostrare una data emozione tramite un'espressione facciale, questo porta la persona a cercare di accentuare quei tratti caratteristici di una data espressione, come ad esempio sopracciglia o bocca, per questo l'individuazione risulta sicuramente più semplice e immediata. Questo non è per forza un male, poiché nello specifico caso d'uso, si doveva verificare se il paziente riusciva a mostrare nella maniera corretta le emozioni per poter empatizzare con chi si trova di fronte. Nel nostro esempio il tutto è leggermente differente, il partecipante viene ripreso mentre è intento a fare altro, le espressioni mostrate sono del tutto naturali, frutto di una reazione spontanea allo stimolo a cui sta venendo sottoposto. Questo comporta delle espressioni facciali mediamente più contenute che verrebbero molto spesso inserite nella cateogoria "neutro" o comunque raramente associate alle emozioni più estreme come ad esempio la rabbia. Per ovviare a questo problema sarebbe stato sicuramente necessario un nuovo allenamento, non potendoci basare sui pesi della rete forniti, e probabilmente una rete più profonda per cercare di andare a cercare delle feature di più alto livello. La scelta di intraprendere un percorso diverso e completamente nuovo è però stata decisa dopo un'ultima questione che è stata sollevata in particolare con il gruppo di ricerca legato alla neuroriabilitazione.

5.1.2 Attivazione e valenza emozionale

Il vecchio modello scegliendo di dividere le emozioni secondo sette categorie fisse, si basava sulla teoria delle emozioni, secondo la quale "Ogni emozione è indipendente dalle altre nelle sue manifestazioni comportamentali, psicologiche e fisiologiche, e ognuna nasce dall'attivazione di percorsi neurali unici del sistema nervoso centrale (SNC)" [43]. In questo modello quindi ogni emozione ha una sua caratterizzazione ben definita e può essere distinta quindi in maniera evidente e univoca dall'individuo. Nel tempo però studi cliniche e ricerche, soprattutto con l'avvento della neuroscienza, hanno mostrato la difficoltà delle persone nel distinguere e descrivere in maniera univoca le proprie emozioni, indicando come dall'uomo non vengano percepite come entità del tutto separate e distinte, ma anzi come esperienze che in alcune situazioni si sovrappongono, producendo sfumature nuove, come se parlassimo dello spettro dei colori. Per questo si è pensato di ricondurle non più a delle categorie ben definite, ma a un modello continuo bidimensionale, le cui due dimensioni sono state concettualizzate in vari modi a seconda dello studio, ma sicuramente una delle più famose è la descrizione secondo attivazione e valenza emotiva [44] . Indipendentemente dal nome attribuito alle due variabili, si è visto tramite un gran numero di studi come questo modello bidimensionale sia assolutamente fondato e coerente con i risultati ottenuti, questo perché i sostenitori del "Circumplex Model of Affect" (così è chiamato questo modello di interpretazione delle emozioni) indicano che le emozioni derivano tutte dalla una combinazione lineare di due sistemi neurofisiologici indipendenti, che quindi possono essere associati uno all'attivazione e l'altro alla valenza. Dall'esperienza cognitiva di questi due sistemi che vengono attivati, l'essere umano deduce quella che è la sua emozione e che prova a ricondurre al nome di un'emozione conosciuta, questo spiega anche perché molti individui con distorsioni cognitive hanno difficoltà a distinguere alcuni stati emotivi che a livello logico sembrano essere molto distanti, ma che magari risultano molto vicini se uno dei due assi viene schiacciato. Per fornire un'idea più chiara può essere molto utile una rappresentazione grafica di questo dominio bidimensionale continuo, entro cui le varie emozioni vanno

a posizionarsi in una zona indicativa. Le zone relative alle varie emozioni spesso tendono a sovrapporsi creando delle sensazioni che non possono essere descritte in maniera univoca, ma che sono filtrate dall'esperienza di ognuno.

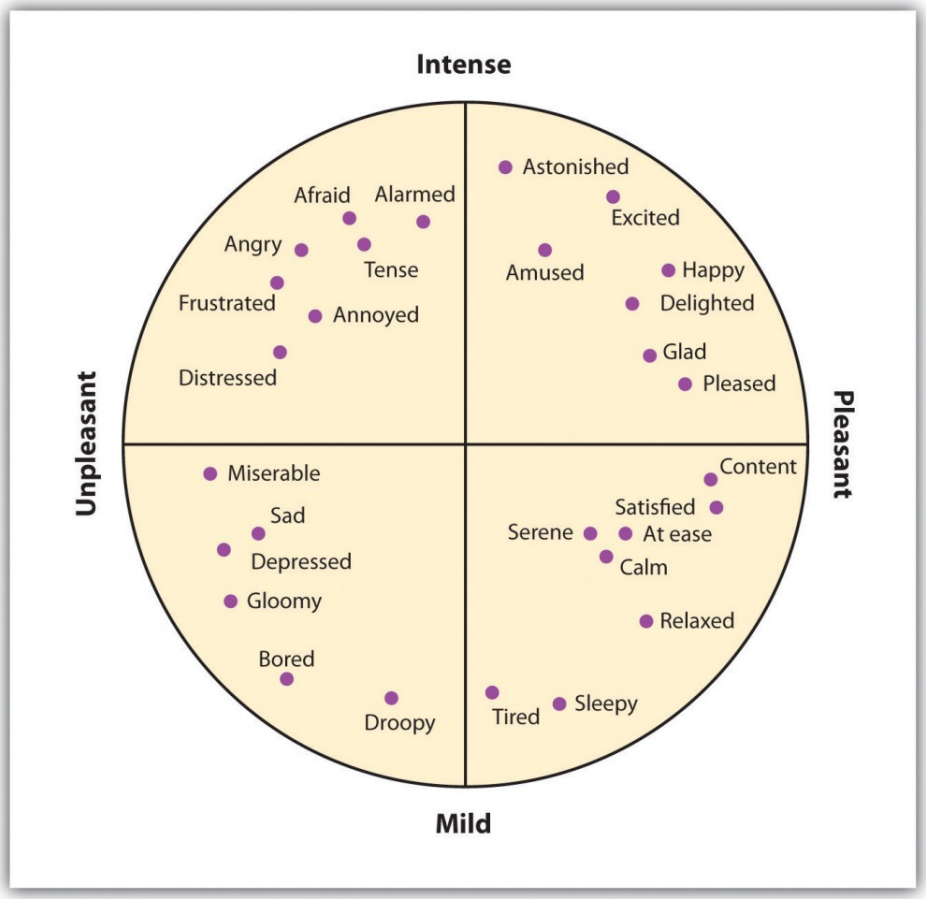

Figure 5.2. Rappresentazione dello spazio bidimensionale di attivazione e valenza emotiva con all'interno rappresentate le emozioni più famose e il punto in cui indicativamente si posizionano. Credit to https://bit.ly/3tLvRXH

Come si può osservare le emozioni sono distribuite in questo spazio bidimensionale, dove l'asse orizzontale rappresenta la valenza, mentre l'asse verticale rappresenta l'attivazione. Infatti, con valenza si indica il livello di piacere che un dato evento provoca sull'individuo e quindi può andare da spiacevole, negativo, a piacevole, positivo. Mentre con attivazione, chiamata

anche intensità, andiamo appunto ad indicare il livello di "attivazione autonoma" che l'evento crea, qui si può passare da una bassa intensità, calmo, ad un'alta intensità, eccitato. Osservando il grafico può essere più chiaro il discorso precedente, molte emozioni si raggruppano in zone molto piccole dello spazio bidimensionale, non permettendo di fornire dei confini chiari e definiti tra le varie emozioni. Allo stesso modo è possibile vedere come emozioni abbastanza distanti, in caso di disagi cognitivi come quelli dei pazienti che saranno sottoposti a neuroriabilitazione, possono andare a sovrapporsi e quindi a creare confusione nel paziente che è in dubbio su cosa stia realmente percependo.

Con il gruppo di ricerca abbiamo quindi concordato che questa modalità di definizione delle emozioni risultava sicuramente più adatta e conforme a quello che è il nostro studio, anche perché permetteva una maggiore flessibilità riguardo all'intensità delle emozioni provate dal partecipante. Inoltre, ad esempio, nell'esperimento legato alla neuroriabilitazione abbiamo potuto fornire un numero di etichette maggiore al paziente, così da permettergli di scegliere in maniera più fedele possibile il nome dell'emozione che pensava di provare, rispetto alle 7 iniziali.

5.1.3 Datasets e distribuzione delle immagini: AffectNet e Aff-Wild

Il cambio di dominio che abbiamo deciso di produrre ha portato ovviamente dei problemi, tra cui la necessità di trovare in primis dei dati su cui poter allenare i nostri modelli che sono stati utilizzati per le predizioni. Non potevano essere utilizzate le stesse risorse del precedente lavoro della University of Essex, poiché ovviamente era stato deciso un cambio di dominio e i dati di training da loro utilizzati erano formati unicamente da dataset etichettati in un dominio categorico, rappresentante le sette emozioni principali. Come è risaputo infatti alla base del Deep Learning ci sono i dati, più le reti sono complesse e profonde, più si vanno a ricercare feature complesse e ciò richiede un gran numero di dati e anche una buona diversità tra questi ultimi. I rischi della pochezza di dati di training sono molteplici, dall'incapacità della rete di

estrarre delle feature degne di nota, fino al classico overfitting, in cui la rete non riesce a generalizzare e impara eccessivamente i dati di training adattando le proprie predizioni ad essi, ne derivano delle performance buone in fase di allenamento, ma estremamente povere quando si va ad utilizzare il modello nel mondo reale. Era quindi necessario individuare un dataset contenente un gran numero di immagini non solo di visi, ma più nello specifico immagini appositamente pensate per lavori legati alle emozioni e soprattutto con le giuste annotazioni, legate al dominio di attivazione e valenza emotiva. Varie sono inoltre le problematiche relative ai vari dataset, sono spesso presenti degli errori nelle annotazioni delle emozioni nelle foto poiché sono spesso fate autonomaticamente da software che hanno dei loro margini di errore. Un altro problema molto presente è la distribuzione di provenienza delle immagini, la loro fonte è spesso la stessa, sono quindi, ad esempio, presenti dataset di foto fatte in laboratorio, con persone in posa atte a mostrare una data espressione facciale. Negli anni si è identificato come i modelli allenati con questi dataset [45] non generalizzino in maniera ottimale una volta portati nel mondo reale non permettendo un riconoscimento adeguato delle emozioni individuate. Questi problemi sono dovuti principalmente alla grande variabilità dovuta alla popolazione molto differente tra loro, ma anche all'ambiente reale in cui i modelli devono poi performare, dovendosi adattare a condizioni di luce e a pose non tradizionali e che nei dataset di training non venivano considerate. La ricerca ha quindi virato sulla composizione di dataset che fossero "in-the-wild", quindi formati da immagini scattate in maniera naturale e in condizioni il meno artefatte possibili. Oltre a questo tipo di attenzione, un altro punto da tenere in considerazione era quello legato alla varietà della popolazione, risulta quindi importante ottenere immagini di persone il più possibile differenti, con tratti molto dissimili, così da permettere alla rete di generalizzare in maniera ottimale anche dai tratti somatici. Infine, un altro problema estremamente presente all'interno dei dataset disponibili di questo genere è la distribuzione non equilibrata dei vari tipi di emozione; in ambiente non controllato risulta non solo molto più

frequente registrare un sorriso, ma anche più semplice individuarlo e annotarlo all'interno di un gran numero di dati. Meno frequente e di più difficile individuazione risultano immagini legate ad emozioni come ad esempio il disgusto ed altri stati emotivi di più rara rappresentazione. Per questo anche in fase di valutazione sarebbe utile tenere presente la distribuzione delle emozioni che può creare dei falsi risultati, dovuti a modelli che tendono a predire molto bene alcune categorie molto e male delle altre che hanno una presenza meno numerosa. Dopo un po' di ricerche abbiamo individuato AffectNet [AffectNet: A Database for Facial Expression, Valence, and Arousal Computing in the Wild Ali Mollahosseini, Behzad Hasani, Mohammad H. Mahoor], come un ottimo database contenente espressioni facciali raccolte in ambiente naturale, quindi non controllato, che godono inoltre di una doppia annotazione, sia nel dominio discreto e quindi con una categoria associata, sia nel dominio continuo con annotazioni riguardo ad attivazione e valenza emotiva. Gli autori del paper di riferimento riportano come motivazioni della creazione di questo database gli stessi problemi evidenziati precedentemente, anche nei database "in-the-wild" è presente un evidente carenza di soggetti differenti e di emozioni diverse, inoltre tra questi database vi è una bassissima presenza di immagini annotate secondo il modello dimensionale da noi scelto. Da questa necessità nasce quindi AffectNet, il cui nome deriva dall'unione di Affect (parola inglese usata in psicologia per indicare le emozioni) e InterNET. Questo dataset è infatti stato costruito utilizzando tre diversi motori di ricerca (Google, Bing e Yahoo) e costruendo circa 1250 diversi tag legati alle emozioni. Questi tag sono stati prodotti unendo il nome delle emozioni associate ad altre caratteristiche come età, sesso e provenienza (es. "ragazza europea gioiosa"), così da permettere di costruire un database il più vario e completo possibile. A quel punto la grandissima quantità di URL, legate alle immagini che erano state ottenute in seguito a numerosi filtraggi per togliere foto con soggetti non umani o con loghi o scritte che coprivano il soggetto, sono state trattate per essere sottoposte a face detection così da isolare il viso della persona. A questo punto sono state assunte 12 persone, esperte del dominio e che quindi avevano compreso a fondo i concetti di attivazione

e valenza grazie anche ad un training iniziale, ad esse sono state fornite circa 450,000 immagini che hanno dovuto annotare attentamente sia nel dominio categorico che in quello continuo. Il risultato di questo enorme lavoro può essere riassunto dalle figure prese dal paper ufficiale [46] e riportate qua sotto.

Expression	Number
Neutral	80,276
Happy	146,198
Sad	29,487
Surprise	16,288
Fear	8,191
Disgust	5,264
Anger	28,130
Contempt	5,135
None	35,322
Uncertain	13,163
Non-Face	88,895

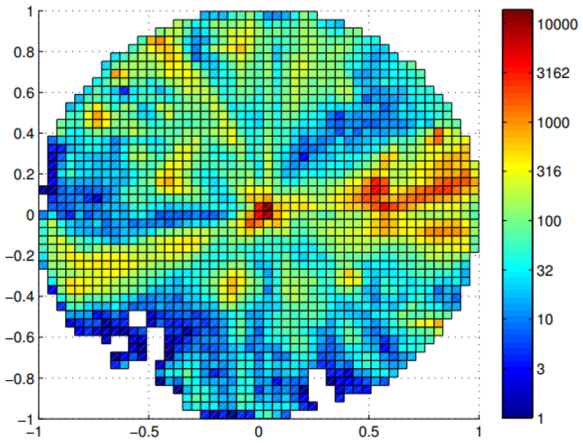

Figure 5.3. A sinistra la distribuzione categorica delle emozioni, con il numero di sample per ogni etichetta. A destra una distribuzione grafica che rappresenta come nelle heat-map le zone dello spazio bidimensionale in cui vi è una maggiore densità di immagini.

A sinistra possiamo osservare la quantità di immagini per ogni categoria, con l'aggiunta di immagini che non rappresentano visi e altre che rappresentano emozioni eccessivamente incerte per poter essere categorizzate. A destra è rappresentata invece una distribuzione del dataset all'interno dello spazio bidimensionale di attivazione e valenza emotiva. Anche questo segue, ovviamente, il trend riscontrato nelle emozioni categoriche, vi sono infatti zone molto popolate, come quelle ad alta valenza e media attivazione, rappresentanti emozioni come la contentezza e la gioia moderata, e zone con pochi o nessun sample, in corrispondenza delle zone estreme dello spazio e soprattutto nel quadrante a bassa attivazione e bassa valenza, quindi emozioni negative e con poca intensità. Come si vede nella tabella vi è sicuramente una disparità evidente tra una categoria e l'altra, ma perlomeno anche in emozioni

poco rappresentate come il disgusto, il numero di sample è comunque notevole. Dovendo utilizzare metodi di Deep Learning, i quali richiedono un gran numero di dati per funzionare in maniera ottimale, si è deciso di aumentare il numero di immagini a disposizione dei modelli e di pari passo migliorare leggermente la distribuzione dei dati. Per ovviare a questo problema si è deciso di utilizzare insieme ad AffectNet un secondo database, chiamato Aff-Wild. Anche questo database contiene immagini legate alle emozioni, ma che sono state raccolte in maniera diversa. Non provengono infatti da foto, ma sono frame di video presi dalla piattaforma YouTube e cercati usando tag che richiamassero reazioni di persone ad eventi. In molti di questi video venivano riprese delle persone che reagivano alla vista o di un video o di una scena particolare e che destasse in loro una qualche sorta di emozione. Il database contiene 298 video con una durata complessiva di circa 30 ore [47]. Le problematiche che ci hanno fatto decidere di utilizzarlo non come unica fonta, ma solo in unione con un'altra, è la scarsa differenziazione tra i soggetti; infatti nel dataset compaiono meno di 200 persone diverse, un campione non troppo esteso che potrebbe impedire alla rete di generalizzare ottimamente. Quello che invece ci ha convinto ad utilizzarlo, oltre alla distribuzione che mostrerò a breve, è che anche questo database è stato annotato seguendo il modello bidimensionale basato su attivazione e valenza emotiva ed è stato fatto manualmente da delle persone addestrate, questo permette un livello di annotazioni riconosciuto come molto buono e affidabile. Infine confrontando le distribuzioni dei due database possiamo vedere come Aff-Wild permette di riempire delle zone che erano poco rappresentate in AffectNet, come in particolare le fasce ad attivazione molto alta, permettendo di coprire un maggior numero di emozioni e situazioni specifiche.

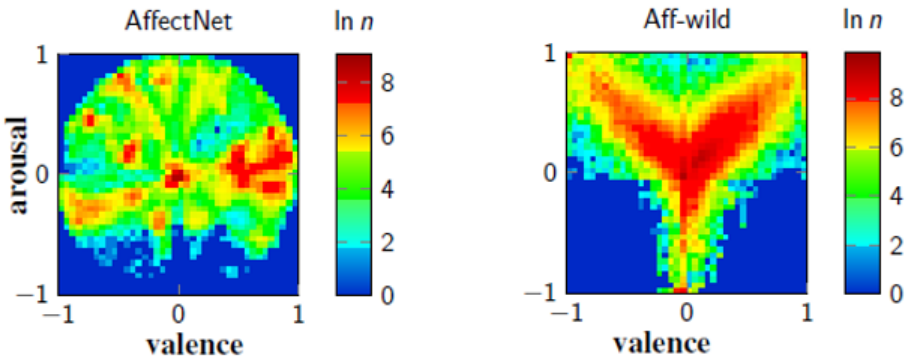

Figure 5.4. Rappresentazione della distribuzione delle immagini nei due database di riferimento, nello spazio bidimensionale di arousal (attivazione) e valence (valenza). Credit to [68]

L'unione di più fonti per i dati può essere un ottimo modo di ampliare la quantità di input per i nostri modelli, ma deve essere fatta con molta attenzione, poiché se le distribuzioni dalle quali i dati vengono presi sono molto diverse tra loro, si può rischiare che la rete impari delle feature non relative alle immagini, ma proprie dei dataset. In questo modo ad esempio se nel database avessimo poche immagini di disgusto e una preponderanza di persone anziane, mentre nel secondo l'opposto, il modello potrebbe pensare che l'età della persona possa andare ad avere un ruolo sull'emozione mostrata. Bisogna quindi essere sicuri che questo tipo di problemi non accadano e che il modello generalizzi nella maniera migliore possibile.

5.2 Le scelte tecniche per il Facial Emotion Recognition

Esplorato quello che è il punto di partenza del progetto, nonché uno stimolo su cui ragionare, e individuate la sorgente e la forma dei dati di input necessari per lo studio, bisogna vedere come si è pensato di implementare il nostro modello e quali tecnologie, più o meno innovative, si sono sperimentate. Essendo diverse persone a poter analizzare questa parte del lavoro, abbiamo

deciso di dividere il processo in tre parti, così da poter studiare nello specifico ogni passaggio e provare un maggior numero di soluzioni. Solitamente la task di Face Emotion Recognition può essere divisa in tre diversi compiti: Face Detection, Feature Extraction e infine Predizione.

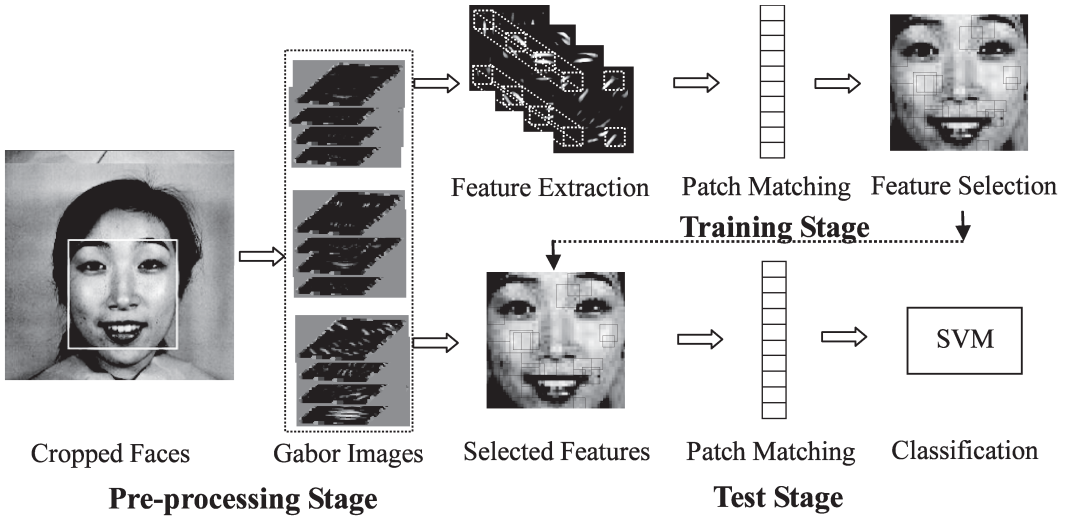

Figure 5.5. Esempio delle fasi di un processo per l'estrazione di informazioni da un volto umano.
Credit to https://bit.ly/3c7TkfC

L'immagine riassume, anche se in un contesto e con tecnologie diverse dalle nostre, quale sarà il nostro approccio al problema. Bisogna infatti prima accedere alla camera del pc e permettergli di registrare l'input, poi prendere i singoli frame e ad essi applicare un algoritmo di Object detection, che ha come unica classe da riconoscere quella corrispondente ad un volto umano. Una volta individuato il viso, l'immagine verrà ritagliata ed inviata alla parte di Feature Extraction, che attraverso vari metodi cercherà di individuare le caratteristiche di interesse per il nostro problema all'interno dell'immagine. Una volta selezionate le caratteristiche più significative, queste saranno utilizzate da un modello che produrrà la predizione desiderata riguardo l'emozione rappresentata dall'immagine. Prima di passare ad analizzare nello specifico è necessario fare alcune precisazioni, la prima è che tutte le tecnologie analizzate sono solo l'inizio di questo processo. Avendo dovuto dapprima produrre una parte applicativa e organizzativa, si sono potute solo gettare le basi

per la parte di Face Emotion Recognition utilizzando machine learning, nei prossimi mesi continueremo infatti ad esplorare soluzioni più complesse e specifiche. Personalmente sono stato assegnato al primo gruppo, quello che si occupava della Face Detection, per questo la trattazione sarà più approfondita in questa sezione. Ciò non significa che ci sia stata una divisione estremamente netta dei lavori, anzi tutti quanti abbiamo studiato la teoria che è dietro tutti i passaggi di questo lavoro, così da poter uscire da questo progetto il più arricchiti possibile. L'ultima premessa è che, come già sottolineato, stiamo cercando di produrre qualcosa che possa funzionare nel mondo reale, ma soprattutto nel nostro ambito applicativo, per questo alcune soluzioni non sono state al momento considerate oppure sono state scartate in corso d'opera visto l'eccessivo carico computazionale che richiedevano ai computer a disposizione, in particolare nel centro di riabilitazione che non basa la propria quotidianità su esperimenti di questo tipo.

5.2.1 Face Detection

La Face Detection, o più semplicemente Localizzazione del volto, è un compito che ha avuto molto successo negli ultimi 20 anni e che è stato risolto con risultati ottimi in molti modi differenti. In particolare, potremmo descriverlo come un caso particolare di Object Detection, in cui però l'unico oggetto fornito come classe di output è proprio il volto umano. Ha applicazione in moltissimi ambiti, soprattutto come primo passo per poter poi passare al riconoscimento facciale, in cui non solo si identificano uno o più volti all'interno dell'immagine, ma si cerca anche di associare il volto identificato ad una persona reale i cui dati sono contenuti all'interno di un database. Mentre la tecnologia in questo ambito è già molto matura, quello su cui si continua a lavorare è l'affidabilità dei classificatori e il loro rendimento in real-time, soprattutto in situazioni di particolare affollamento, come potrebbe essere la situazione di una telecamera per la sicurezza all'interno di un evento pubblico ad alta densità come in uno stadio o in una piazza. Tornando però nello specifico alla Face Detection, uno dei metodi più famosi, che è anche stato sperimentato all'inizio del progetto tramite la libreria OpenCV a scopo

principalmente didattico, è quello presentato dal paper [51], dove viene descritto il primo face detector funzionante in real-time. La cosa interessante di questo progetto, oltre al suo primato, è senza dubbio l'utilizzo di quelle che sono chiamate Haar-like feature, delle caratteristiche che vengono rilevate applicando in zone rettangolari adiacenti della finestra selezionata dei filtri particolari. Questi filtri vengono usati sommando e sottraendo i pixel nelle zone bianche e nere del filtro, in questo modo calcolando il risultato si possono categorizzare alcune piccole zone dell'immagine. Questa tecnica produce contenuto informativo a seconda della differenza dei pixel nel risultato e questo può avere un'interpretazione poiché vi sono alcune caratteristiche che costantemente sono riscontrate e vengono rispettate nel caso di presenza di un volto umano. Una parte che viene presa molto in considerazione sono gli occhi, che sono di fondamentale importanza per il riconoscimento di un volto, in particolare si è visto che la zona degli occhi risulta sempre più scura rispetto a quella sottostante delle guance. Quindi riassumendo viene presa una finestra che viene traslata su tutta l'immagine e per ogni posizione vengono applicati questi filtri secondo dei pattern ben definiti, per estrarre feature a più livelli di astrazione. Prima di fare la predizione, viene applicata un'altra idea del paper, quello di Cascade of classifier, ossia per ridurre il carico computazione e permettere la riuscita real-time i filtri vengono applicati a cascata partendo da quelli legati a feature di alto livello, così in caso di fallimento in questo passo si passa subito alla finestra successiva senza applicare i filtri seguenti. Da qui poi c'è stata un'evoluzione estremamente rapida di questo concetto, riguardo i risultati di accuratezza che erano comunque abbastanza carenti, grazie alle migliorie in ambito di Object Detection. Grazie al Deep Learning si è potuto utilizzare un approccio più black box in cui quindi in alcuni casi non c'è più bisogno di un intervento dell'uomo per quanto riguarda l'estrazione delle feature specifiche del dominio. Nel nostro lavoro abbiamo perciò deciso di utilizzare le due tecnologie che nel capitolo precedente ho nominato, ma che non avevo ancora approfondito e che risultano essere lo stato dell'arte in questo campo: You Only Look Once (YOLO) e Single-Shot Multibox Detectors. Prima di parlare di queste due tecniche

volevo solo spiegare due termini che verranno utilizzati in entrambe:

- IoU, significa Intersection over Union ed è una metrica di accuratezza di un object detector. In particolare, questo concetto è utilizzato quando si parla di bounding box per capire quanto una data predizione sia affidabile. Avendo infatti quello che è definito come ground-truth bounding box, ossia il vero quadrato che racchiude l'oggetto, e il bounding box calcolato dall'algoritmo, possiamo calcolare IoU come area di sovrapposizione tra i due box, diviso area di unione di questi due. Quindi maggiore è l'IoU, migliore è la predizione che il nostro algoritmo ha effettuato rispetto all'oggetto reale.

- NMS, significa Non-Maximum Suppresion ed è un algortimo di filtrazione sulle proposte di bounding boxes. Nei vari modelli infatti vengono calcolati un gran numero di bounding boxes, NMS si occupa di prima di tutto di ordinare per confidenza i box. Partendo dal primo, se ha un IoU maggiore di una data soglia viene ammesso alla lista delle proposte. A seguire i box successivi non solo dovranno avere un IoU rispetto al bounding box reale sopra un certo minimo, ma dovranno anche avere un IoU, calcolato tra sé stessi e tutte le altre proposte già inserite in lista, minore di una data soglia altrimenti verranno scartati. Questo permette non solo di scartare tutti i box che contengono male l'oggetto, ma anche di proporre un solo bounding box per un singolo oggetto, invece di averne diversi sovrapposti.

You Only Look Once o YOLO

Come visto nel capitolo precedente parlando della Object Detection si era arrivati ad analizzare le immagini identificando gli oggetti in maniera davvero buona grazie alle R-CNN e alle sue variazioni (Fast R-CNN e Faster R-CNN), senza però riuscire ancora a fare queste operazioni real-time. Questa limitazione era dovuta principalmente al gran numero di passaggi da eseguire, infatti volendo dividere le azioni fatte dalla R-CNN in due fasi, possiamo dire che prima si cercano le zone in cui potrebbe risiedere un oggetto con

una Region Proposal Network, che produce quindi anche un approssimativo bounding box, e successivamente, con le feature ricavate dall'area presa in esame, si va a determinare la classe legata all'oggetto rappresentato in quella data area. YOLO porta una novità in questo contesto, la parte di feature extraction, di localizzazione dell'oggetto e di classificazione dello stesso vengono unificate in un unico blocco. Vi è quindi un'architettura a singolo stage prettamente convoluzionale e invece di avere due output vi è un unico vettore di feature che va a comprendere le varie qualità cercate nell'oggetto (posizione e tipologia). Come si legge nel paper di presentazione della tecnologia [52] "Abbiamo riformulato l'object detection come un singolo problema di regressione, dritto dai pixel dell'immagine alle cordinate dei bounding box e alle probabilità delle classi". Il nome You Only Look Once prende l'idea dal fatto che quindi non vengono analizzate solo specifiche zone dell'immagine, ma l'intera immagine viene analizzata dai layer convoluzionali, permettendo alla rete di cogliere delle informazioni del contesto che sfuggirebbero analizzando solo zone specifiche. L'immagine in input viene divisa in una griglia di SxS celle, ognuna responsabile di catturare l'oggetto, se presente, che è contenuto al suo interno. Ogni cella produce B bounding box con una relativa confidenza, legata alla probabilità secondo il modello che sia presente un oggetto nel box evidenziato (deve quindi essere zero nel caso non siano presenti oggetti). Questa probabilità non solo va a definire la probabilità che un oggetto risieda nel riquadro, ma anche che il bounding box vada a corrispondere con l'effettivo riquadro che contiene perfettamente l'oggetto. Infine ogni cella predice anche una probabilità condizionata che data la presenza dell'oggetto, esso appartenga ad una determinata classe, ossia Probabilità(Classe i | Oggetto). In questa maniera si vanno a creare non solo dei bounding box con relative confidenze, ma anche una mappa della probabilità delle classi che va ad assegnare ad ogni cella la classe più probabile a cui appartiene, da quelle di uno specifico oggetto, fino alla classe sfondo se non è stato identificato nulla.

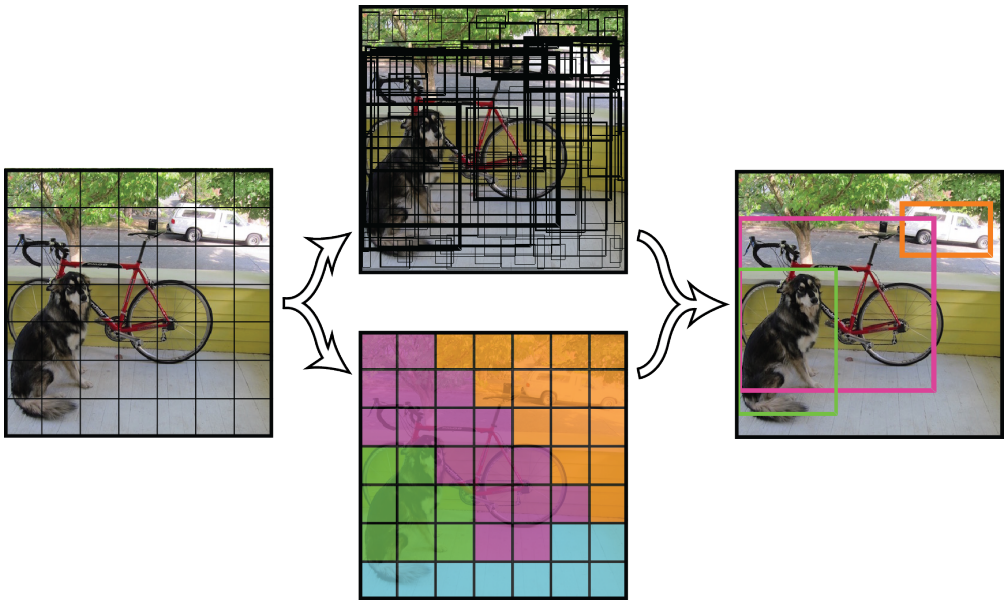

Figure 5.6. Immagine del paper ufficiale in cui sono mostrati i due porcessi che vengono compiuti sull'immagine: predizione dei bounding boxes e classificazione delle varie celle.
Credit to https://pjreddie.com/darknet/yolov1/

Applicando Non-Maximum Suppression si vanno ad eliminare sia i box sotto a un certo livello di IoU, sia quelli che hanno una sovrapposizione molto elevata andando a prendere quello con IoU più elevata, non permettendo di produrre più box che vanno ad identificare lo stesso oggetto. Da questo processo esce un tensore di dimensioni SxS x (B * 5 + C), con SxS numero di celle, B numero di bounding box per cella e C numero di classi possibili; si viene così a creare un output molto più piccolo dei precedenti e che quindi viene calcolato in maniera più veloce. Dal paper originale sono derivate due ulteriori versioni di questa rete, andiamo quindi ad analizzarne brevemente le caratteristiche: - YOLO, permette di processare frames in real-time, nello specifico a 45 fps con ottimi risultati per quanto riguarda la mAP (mean Average Precision) che sul dataset COCO ha raggiunto mAP = 63.4. Viene presentata inoltre una versione alternativa di questa rete chiama Fast YOLO che a prezzo di un calo di accuratezza, visto un minore numero di layer convoluzionali, raggiunge però i 155 fps. La struttura di YOLO comprende 24

convolutional layer, alternando molti layer 1x1 e 3x3, riprendendo l'idea dalla famosa GoogLeNet per ridurre il numero di parametri, e due fully connected layer alla fine della rete. La rete impara meglio le feature dell'immagine fornendo meno falsi positivi riguardo lo sfondo; ha però problemi di localizzazione e inoltre ha problemi con oggetti piccoli e in gruppo visto che ogni cella può identificare una sola classe. - YOLO v2 o YOLO9000 [53], raggiunge mAP= 76.8 a 67 FPS. Sono stati fatti alcuni cambiamenti mirati a migliorare i problemi di localizzazione e di "recall". In particolare: viene aggiunto Batch Normalization a tutti i layer convoluzionali; la rete viene allenata su immagini a più alta risoluzione (da 224x224 a 448x448) e, di conseguenza, aumentate il numero di celle in cui viene divisa l'immagine; vengono rimossi i fully connected layer permettendo la predizione dei bounding box non più da questi, ma utilizzando degli Anchor Boxes e utilizzando il k-means clustering per ricavarne la dimensione ottimale da cui partire; utilizza come rete DarkNet a 19 layer convoluzionali. Il nome YOLO9000 deriva dal fatto che può identificare oltre 9000 categorie di oggetti. - YOLO v3 [54], permette un incremento sia in accuratezza che in velocità, la rete accetta e lavora su input di varia grandezza, mantenendo ottimi risultati. Alcuni dei cambiamenti effettuati sono: la predizione dei bounding box avviene utilizzando la regressione logistica, come offset partendo dal centroide del cluster; utilizza una Darknet a 53 layer come estrattore di feature che fa uso di "skip connections" (concetto preso dalla ResNet) permettendo di prevenire la scomparsa del gradiente e quindi di aumentare di molto anche il numero totale dei layer; la capacità di fare predizioni in varia scala dipende dall'uso della Feature Pyramid Networks (FPN). Questo concetto merita un approfondimento, si tratta di una tecnica pensata dal FAIR [55] che pone un cambiamento nel metodo classico con cui fare predizioni. Il problema nel predire oggetti piccoli è che facendo predizioni sulla feature map calcolata dopo molte convoluzioni, la dimensione di questa diminuiva molto rischiando di far perdere molte caratteristiche legate ad oggetti di piccole dimensioni. Con la FPN arrivati alla feature map finale, viene prodotto un upsample e le caratteristiche ottenute sono fuse insieme con quelle precedenti, permettendo di cogliere feature a più

livelli di astrazione e di operare predizioni su risoluzioni diverse. L'immagine qui in basso dovrebbe permette di chiarire notevolmente il concetto che a parole può risultare confuso.

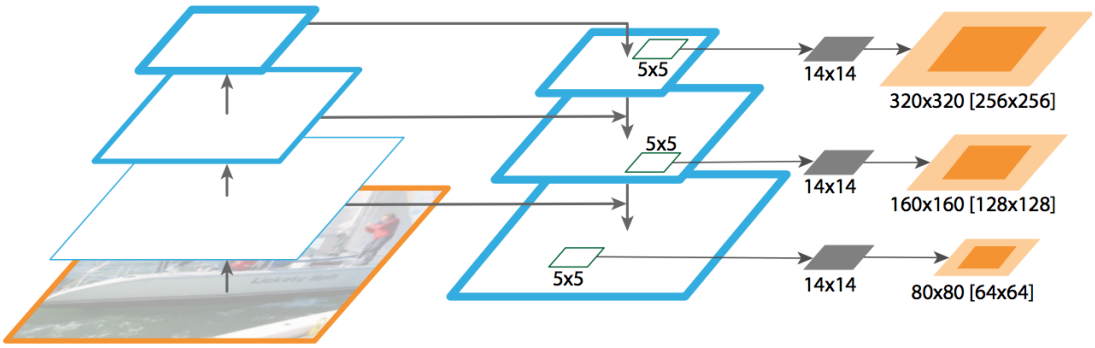

Figure 5.7. Spiegazione grafica di come è gestita la creazione delle feature map in seguito al processo di upsampling in una Feature Pyramid Networks. Credit to https://bit.ly/3rbjBhe

Single Shot Multibox Detector

Sia YOLO che Single Shot Multibox Detector, che da ora chiameremo SSD per comodità, sono state denominate tecnologie Single Shot per metterle in contrapposizione con le R-CNN e il loro processo a due fasi. Questa volontà di ridurre un processo di training troppo lungo, ma soprattutto di non spezzare il flusso in due fasi, porta a trovare, come visto nel paragrafo precedente, delle soluzioni ad un solo step che permettono di produrre object detection in tempo reale. SSD viene presentata successivamente alla prima versione di YOLO e cerca di imporsi come una tecnologia migliore e migliorata, a questo YOLO risponderà con le versioni successive già trattate. Per la trattazione ho trovato molto utile ed esplicativo l'articolo [56] da cui ho ripreso la struttura della spiegazione. SSD si compone di due parti principali, una rete che ha il ruolo di estrattore di features, nel paper originale viene utilizzata VGG16, e dei layer convoluzionali successivi alla rete che servono a identificare gli oggetti all'interno dell'immagine. Per andare più a fondo nella spiegazione l'immagine del paper [57] può essere di grande utilità per capire i concetti.

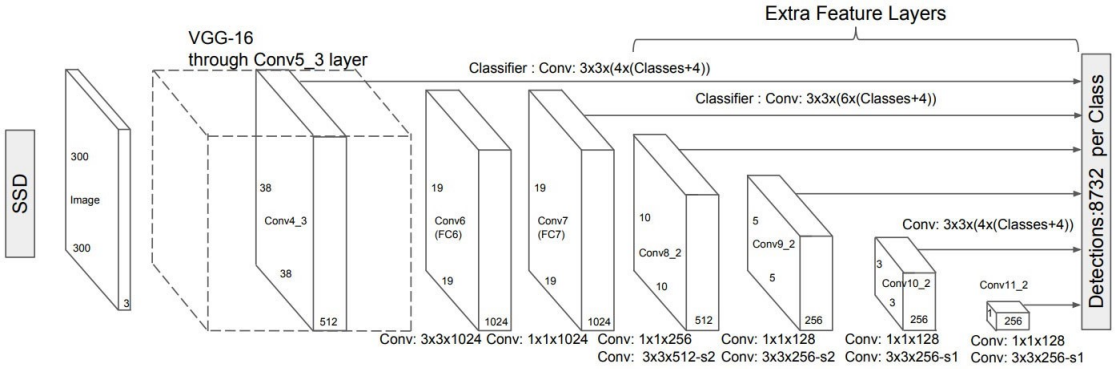

Figure 5.8. Immagine del paper ufficiale in cui viene mostrata la struttura di SSD.
Credit to [57]

Come si può vedere al layer finale, in cui viene elaborato l'output, arrivano informazioni da vari layer convoluzionali, applicati successivamente. In particolare vengono prodotte una serie di feature maps, che sono divise in celle o locazioni, ognuna delle quali produce 4 bounding boxes con annesse predizioni per ogni classe (compresa la classe "no object" ovviamente). Questa parte del produrre multiple predizioni su box e confidenze è ciò che fornisce a questo metodo il nome di Multibox. Inoltre, come si può vedere dall'immagine ad ogni layer convoluzionale vengono applicati filtri convoluzionali 3x3 sulle feature maps ricavate, questa operazione permette di ricavare punteggi di confidenza sia sulla localizzazione degli oggetti che sulle classi a cui appartengono, come rappresentato con le frecce che escono dagli strati convoluzionali per arrivare a quello finale. L'operazione appena descritta però non viene fatta una sola volta in uscita dalla rete per l'estrazione di feature, che potremmo indicare come rete di backbone, ma anzi viene riproposta dopo ogni layer convoluzionale. Questa scelta è stata utilizzata per identificare oggetti che abbiano una diversa dimensione, da piccoli a molto grandi, rilevandoli in maniera indipendente e senza variare la tecnica per stabilire i bounding box da applicare. Come sappiamo infatti le CNN dopo ogni layer riducono la dimensione spaziale dell'input in maniera graduale grazie all'applicazione di un filtro, questo significa che ogni feature map prodotta avrà una diversa

dimensione spaziale e di conseguenza anche un minor numero di celle, si parla infatti di "multi-scale feature maps". Questo processo produce delle feature maps a risoluzione sempre minore man mano che si procede nella rete, permettendo di individuare oggetti di dimensione maggiore. Questo nella struttura originale viene fatto con 6 layer convoluzionali successivi, in alcuni dei quali invece di 4 predizioni, ne vengono fatte 6. Per capire completamente il processo, però, bisogna comprendere in che modo vengono stabiliti i bounding boxes su cui vengono fatte le predizioni. Si può ovviamente partire in maniera casuale, ma si è visto come soprattutto nelle prime fasi di training ci sarebbero stati grandi problemi per arrivare a dei risultati decenti e stabili. Per questo si è deciso di utilizzare un concetto che abbiamo già sentito nominare quello di anchor o default bounding box. Questo significa che i bouding boxes vengono scelti manualmente, in modo che coprano un ampio spettro di oggetti della vita reale, che vengono raggruppati per forma e scala. Da questi default bounding boxes si cerca di calcolare quelli reali, calcolando l'offset da quelli di partenza e rispetto al centro della cella stessa. Ovviamente sapendo che a seconda della feature map, si stanno cercando diversi oggetti poiché si guarda ad una scala diversa, anche i deafult bounding boxes utilizzati saranno specifici di quella data feature map e studiati appositamente per quella data scala. Quindi sulla feature map più grande, quella in uscita dalla rete VGG16, si andranno a cercare gli oggetti più piccoli e man mano che si procede verso la fine, le feature map caleranno di risoluzione e si andranno a cercare oggetti sempre più grandi. Ovviamente anche qui per capire quale bounding box tra quelli proposti ha prodotto un match positivo con un oggetto, si usa il concetto di IoU, quindi ad esempio tutti quelli che hanno un default bounding box con IoU > 0.5, sono considerati esempi positivi. A questo punto si può capire l'immagine di esempio riportata nel paper ufficiale.

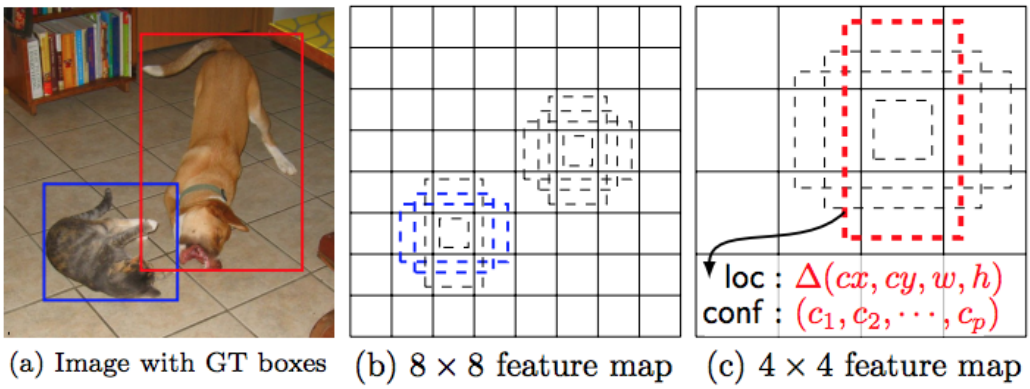

Figure 5.9. Esempio di utilizzo delle feature maps su più scale per individuare oggetti di dimensione diversa.
Credit to [57]

Come si può vedere i due oggetti, in questo caso due animali, hanno dimensioni molto diverse e non potrebbero essere identificati da box con la stessa scala. Con questa metodologia invece il gatto che è più piccolo viene identificato su una feature map precedente, mentre il cane su una successiva caratterizzata da celle più grandi. Come si può vedere vengono evidenziate due misure, localizzazione e confidenza, infatti la funzione di Loss finale è composta da 2 diverse funzioni. La localization loss che indica la differenza tra i box effettivo e quello predetto, penalizzando solo gli esempi positivi (in cui si è riscontrato un match) e non quelli negativi, e la confidence loss che si occupa dell'errore sulla predizione della classe, per gli esempi negativi si penalizza sulla confidenza nella classe corrispondente al sopracitato "no object". Come possiamo vedere gli esempi negativi andranno ad influenzare l'allenamento e saranno un gran numero rispetto a quelli effettivamente accettati. Perciò come spiegano gli autori si è pensato di effettuare "hard negative mining" andando ad ordinare gli esempi negativi nell'ordine della loro confidence loss e prendendo all'interno del training solo quelli con un valore più alto, tenendo il rapporto negativi/positivi al massimo come 3:1. Alla fine di questo processo saranno effettuate un gran numero di predizioni, come

si può vedere nell'immagine, 8732 per l'esattezza, per questo quelle a confidenza eccessivamente bassa sono eliminate, inoltre si applica Non-Maximum Suppression per le predizioni che puntano allo stesso oggetto. Presentando, come YOLO, problemi sul riconoscimento di oggetti di piccola dimensione, uno dei test degli ideatori è stato agire su immagini a risoluzione più alta, passando da 300x300 a 512x512, in questo modo si sono ottenuti risultati leggermente migliori ma con una notevole riduzione degli fps e quindi della velocità di elaborazione.

Le successive due sezioni, riguardanti le rimanenti due fasi del processo di Face Emotion Recognition, saranno meno dettagliate, sia perchè si trovano ad un livello di sviluppo meno avanzato, sia perchè sono state da me approfondite in maniera minore.

5.2.2 Feature Extraction

Individuata e isolata la porzione dell'immagine contenente il volto della persona di cui si cerca di individuare le emozioni, si deve procedere a tentare di estrarre quelle che sono le feature che permetteranno di cogliere l'espressione facciale dell'individuo. A seconda del processo, può essere necessaria una fase di preprocessing, quello che in praticamente tutti i modelli viene fatto è il cropping del volto, ossia al modello successivo viene inviata un'immagine più piccola e quindi ritagliata, in cui è presente la porzione di interesse. Da questo processo base ne possono seguire molti che variano a seconda che ci si trovi in fase di training o di testing. Quando si sta allenando il modello, infatti, si è rivelato estremamente utile applicare sulle immagini utilizzate il concetto di data augmentation, che permette di aumentare notevolmente la dimensione del dataset di allenamento. Fare data augmentation significa applicare delle trasformazioni alle immagini, così da poter ottenere un maggior

numero di sample leggermente differenti partendo da un insieme più piccolo di input. Queste trasformazioni possono essere di vario tipo, ci può essere infatti un cambio di scala o una rotazione del volto individuato, ma anche forme più aggressive come il cambio dei colori nell'immagine o l'aggiunta di rumore. Tutte queste trasformazioni non solo aumentano la quantità di input, ma rendono notevolmente più stabile il modello aiutando a generalizzare meglio sui risultati. Altre trasformazioni che possono essere applicate, invece, in fase di test, riguardano la normalizzazione con varie tecniche per normalizzare l'illuminazione dell'immagine oppure per stabilizzare la posa della faccia, soprattutto in ambito 3D dove si cerca di riprodurre il viso sempre nella posizione frontale [58]. Finite queste operazioni di pre-processing l'immagine risultante può arrivare al modello scelto che si dovrà occupare di estrarre delle caratteristiche dall'immagine sulla base delle quali si potrà poi associare l'espressione facciale più adatta. Il modo più diffuso di esportare contenuto informativo dall'immagine è quella di individuare i cosiddetti "facial landmarks", ossia i punti di riferimento del viso, letteralmente dei puntini che vanno ad individuare alcuni punti cruciali che permettono di tracciare l'espressione facciale all'interno di un viso umano.

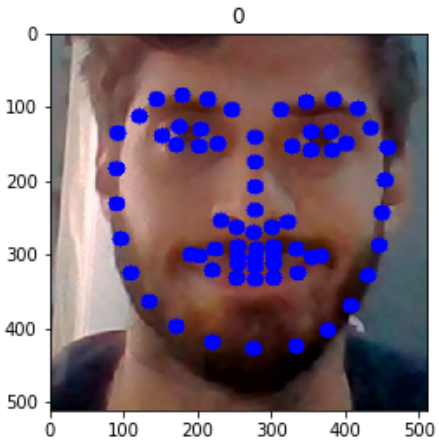

Figure 5.10. Estrazione dei facial landmarks su una foto del mio viso durante il training.

Se ne possono individuare un numero a piacere e il numero di punti individuati da un dato modello dipende ovviamente dal dataset su cui viene addestrato; l'importante è, però, andare a tracciare i punti che permettono sia di disegnare la forma del volto umano, sia di identificarne l'espressione. Per questo le zone dove si raggruppano i punti, come si può anche vedere dall'immagine, sono quelle che vanno a individuare l'ovale del volto e poi all'interno quelle utili a determinare le caratteristiche chiave dello stesso: bocca, naso, occhi, sopracciglia e nei modelli più complessi anche alcune rughe espressive se presenti. In questo momento del progetto si sta cercando ricondurre l'estrazione nello specifico a 68 facial landmarks, poiché i modelli sono stati allenati sul dataset I-BUG 300-W [59] che presenta appunto annotazioni seguendo questo scema di 68 punti, avendone dimostrato l'efficacia. Uscendo da questa fase, l'obiettivo è quindi di fornire un vettore che rappresenti questi punti, basandosi sul quale l'algoritmo successivo potrà fare le sue predizioni. Una strada che si sta analizzando e sul quale si sta lavorando in parallelo è quella di diminuire i facial landmarks su cui andare ad effettuare le predizioni, in due modi differenti: si può effettuare feature selection con una grande varietà di metodi, individuando quelli che sono i punti a più alto contenuto informativo tra i 68 individuati, in questo modo solo questi ultimi saranno inviati al modello predittivo; un'altra strada è quella di allenare il modello direttamente su un dataset con annotazioni relative a meno punti, tra le 14 e le 20, poiché si è visto come si possano ottenere comunque buoni risultati. Questi tentativi sono tenuti in considerazione, visto il desiderio di mantenere basso il carico computazionale in fase di utilizzo, visto che la qualità delle macchine a disposizione non potrà sempre essere ottimale. L'intuizione di utilizzare i facial landmarks per estrarre le caratteristiche espressive di un volto non è così recente, sono quindi diversi i metodi per farlo, sia legati alle reti neurali che non, il gruppo sta quindi provando entrambi gli approcci per testare non solo i risultati, ma anche il carico computazionale ottenuto.

Ensemble of regression tree

Il metodo non Deep, quindi non legato all'ambito Deep Learning, che si sta utilizzando è un ensemble di regression tree. Ci sono due concetti da chiarire brevemente per poterne comprendere il funzionamento, poiché il concetto di regressione è stato già affrontato. Un albero decisionale, appunto tree, è un modello predittivo molto semplice utilizzato sia per classificazione che per regressione. Può essere profondo a piacere ed è composto da nodi, archi e foglie. I nodi rappresentano una variabile e da ogni nodo partono due o più archi, a seconda del valore o del vincolo posto su quella data variabile, che saranno collegati ad un nodo figlio; un nodo senza figli è detto foglia e contiene la predizione effettuata. Ensemble vuol dire insieme, infatti il nostro modello è un'insieme di alberi di regressione che lavorano insieme per il risultato secondo l'algoritmo chiamato Gradient Boosting. In particolare, il Boosting è un meta-algoritmo che permette di creare dei modelli di apprendimento forti, Strong Learner, basandosi sull'unione di più modelli di apprendimento deboli, Weak Learner, come sono appunto gli alberi di decisione. Per fare ciò a livello generale si producono più modelli consecutivamente e in ognuno si cerca di porre rimedio agli errori del modello precedente, in maniera diversa a seconda dell'algoritmo specifico, così che il modello generale diventi più attento a quelli che sono stati gli errori che nella generazione precedente non gli hanno permesso delle performance ottimali. Uno degli algoritmi più famosi basati su questo concetto è AdaBoost, che nei classificatori successivi non fa altro che aumentare i pesi dei sample in cui si è avuta più difficoltà di predizione precedentemente. L'algoritmo da noi utilizzato, Gradient Boosting, può essere visto come una generalizzazione di AdaBoost. In questo procedimento si genera una funzione di loss, che è tipica dello specifico problema, che deve essere differenziabile, scelti i Weak Learner utilizzati (nel nostro caso i regression tree) questi saranno aggiunti al modello in maniera additiva ad ogni passaggio e verrà seguita una procedura di Gradient Descent per minimizzare la loss. Invece di applicare questa funzione ai parametri di una rete, qui viene applicata direttamente ai modelli deboli, gli alberi vengono infatti parametrizzati e questi parametri vengono modificati nella direzione

del gradiente così da ridurre la loss. Ogni passaggio viene, quindi, aggiunto un albero che teoricamente va a ridurre la loss generale o finchè non si raggiunge un valore soglia prestabilito o finchè non si raggiunge un massimo di alberi fissato a priori. Nello specifico il modello utilizzato è basato su quello descritto in [60], che utilizza i concetti descritti precedentemente per fornire un modello che raggiunga performance competitive e lo faccia tranquillamente in real-time, applicando molte tecniche accanto al processo principale, come regolarizzazione e feature selection, di cui non si entrerà nello specifico. L'output di questa rete, e che sarà passato al modello successivo, è un vettore composto da 68 coppie di valori, rappresentanti ognuna un punto e che lo mappano spazialmente con la coppia x,y all'interno dell'immagine.

Approcci CNN-based [61]

Come nella maggior parte degli ambiti legati alla computer vision, anche in questa particolare task le CNN stanno diventando la tecnologia predominante negli ultimi anni. Si è visto infatti come, nonostante l'individuazione dei facial landmarks sia una questione non così recente, si siano ottenuti risultati davvero ottimi applicando approcci che si basino sulle reti convoluzionali e le loro varianti. Se un tentativo di estrazione di facial landmarks è stato effettuato basandosi su alberi di regressione, utilizzando un modello già allenato, quando si è deciso di provare ad allenare un modello da zero, training from scratch, su un numero minore di punti si è optato per una CNN. In particolare, questo genere di tecnlogie si dividono in due categorie principali: approcci di regressione, che vanno a dedurre le cordinate spaziali direttamente dall'immagine, e approcci con heatmap, che invece identificano la posizione in un set di heatmap bidimensionali. I primi si dividono in:

- Modelli a regressione diretta, in cui il vettore con le cordinate dei landmark che sono stati individuati viene ricavato direttamente dall'immagine attraverso una qualunque rete di backbone per l'estrazione di feature.

- Modelli a regressione a cascata, concetto utilizzato anche nell'ensemble

of regression tree descritto precedentemente, un sotto-rete crea un vettore di landmark che però non è definitivo, ma anzi ad ogni iterazione si aggiorna e cambia, fino a raggiungere il numero di iterazioni stabilite.

Come metrica di distanza tra punto effettivo e punto predetto in fasi di training si usa la distanza L2 per allenare entrambe le tipologie di modelli. I modelli a cascata sono solitamente più efficaci, ma il problema è che non c'è una tecnica per definire in maniera corretta quale sia il numero migliore di passaggi necessari a questo genere di modelli. Nei modelli a regressione si creano relazioni molto forti tra i vari landmark visto che le informazioni strutturali legate alla forma del viso vengono imparate dai layer fully connected alla fine della rete, questo è da un lato una cosa positiva poiché si riescono ad ottenere buoni risultati e anche in caso di parti della faccia non visibili si riesce a ricostruirne la forma, dall'altro è un problema perché il modello invece di posizionare i punti dove davvero risiedono spesso tende a prioritizzare il concetto imparato di faccia, seguendo quella che sa essere di solito la sua forma anche se questa è per qualche motivo in un formato non standard. Il secondo approccio, come detto, si basa sull'indicare la posizione dei landmark in una heatmap bidimensionale e si basa sul concetto di fully convolutional, una tipologia di rete già descritta in ambito object detection e che si compone di una parte convoluzionale per l'estrazione delle feature e di una parte de-convoluzionale che codifica queste feature in un set di heatmaps. Anche queste possono essere divise in tre tipologie diverse: distribution models, in cui la posizione del landmark è indicata tramite una distribuzione multivariata, il cui centro è posizionato alle coordinate del landmark, viene quindi prodotto un canale nella heatmap per ognuno di questi; heatmap regression models, inseriscono un canale in più per lo sfondo e di conseguenza variano anche metrica e funzione di loss; pixel-wise classification models, qui le heatmap non indicano la posizione del landmark, ma è generata da un set di vettori che indicano la probabilità che un dato pixel appartenga ad un landmark specifico oppure allo sfondo, dalle probabilità dei pixel si può risalire alle posizioni dei landmark. In generale gli approcci basati su heatmap predicono in maniera accurata la posizione dei landmark,

ma questi non godono di relazioni strette fra di loro per questo in caso di parti coperte o non visibili i landmark non vengono rilevati e sono posti agli angoli o ai bordi dell'immagine, distorcendo notevolmente la figura derivante.

5.2.3 Predizione

Questa è l'ultima fase del processo in cui i risultati di tutti i passaggi dovranno essere esplicitati e quindi ad un dato frame saranno associati i due valori corrispondenti ad attivazione e valenza emotiva. Questi due valori andranno entrambi in un range che va da -1 a 1 e si dovrà decidere una metrica per valutare la bontà del modello costruito. Queste predizioni, quindi, non sono fatte su etichette categoriche, come il precedente lavoro, ma su valori continui, per questo non siamo in presenza di una classificazione, ma di una regressione. Per questo motivo non possiamo utilizzare l'accuratezza per valutare le performance del modello, ma dovremmo utilizzare delle metriche specifiche per la valutazione della regressione, dove infatti non ci interessa granché sapere se la predizione è esatta al millesimo, ma anzi dobbiamo capire di quanto si discosta dal valore identificato come reale. Per questo tutte le principali metriche non fanno altro che evidenziare quanto i nostri risultati predetti siano mediamente vicini, o lontani, da quelli effettivi. Una delle metriche di errore per problemi di regressione più famosa e che abbiamo deciso di utilizzare, seguendo l'esempio di molti paper che trattano la stessa tipologia di applicazione, è il Mean Squared Error o in breve MSE. Il MSE è la media delle differenze quadratiche tra i valori predetti e quelli attesi, secondo la formula:

$$MSE = \frac{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}{N}$$

Con x_i valore atteso e $\hat{x}_i$ valore predetto.

Il quadrato risulta importante perché, oltre a rendere l'errore sempre positivo levandone il segno, va ad ampliare l'errore, punendo quindi i modelli che commettono errori molto grandi. Ovviamente utilizzando il MSE come funzione di loss questo andrà minimizzato, come era intuibile anche dalla formula, per questo più il suo valore sarà basso, migliore sarà il funzionamento

del modello. Accanto a questa metrica seguendo il modello di alcune challenge su dataset di questo ambito si vorrà affiancare a questo valore anche il Concordance Correlation Coefficient o CCC, che va a misurare la concordanza di due set di misurazioni, di cui uno è preso come "gold set" ossia come standard. Questo valore di concordanza può andare da -1 (perfetta discordanza) a +1 (perfetta concordanza) e nelle challenge è stato indicato come di grande significato per quanto riguarda dati di origine video da cui derivano quindi immagini che hanno un legame temporale e quindi una correlazione. Riflettendo su questi concetti e sul fatto che la nostra applicazione dovrà si fare riconoscimento delle emozioni basandosi su espressioni facciali, ma che le immagini non saranno altro che frame di video, si è deciso di allargare ulteriormente il dataset in futuro utilizzando un altro dataset basato su video come lo era Aff-Wild. Il dataset selezionato è chiamato CK+ [62][63] e contiene 593 video per un totale di 123 soggetti differenti. La scelta di separare le fasi di Feature Extraction e predizione, non è molto comune negli ultimi lavori, poiché si è visto che le reti neurali profonde con layer convoluzionali riescono a fare entrambe le cose con buoni risultati. Il nostro tentativo è però quello non solo di ottenere dei buoni valori di precisione, ma anche di avere delle performance decenti con l'hardware a disposizione. Per questo motivo la scelta effettuata è stata quella di utilizzare modelli più semplici o anche solo meno profondi che facessero le loro predizioni sull'array di landmarks che gli veniva fornito.

Nella letteratura tra i modelli applicati unicamente in fase di predizione si trovano principalmente SVM e alberi di decisione, soprattutto i primi però hanno avuto un discreto utilizzo. Volendo fornirne una breve descrizione, SVM o Support Vector Machine è un modello che permette di trovare l'iperpiano che separa i punti del nostro dataset nelle due classi che abbiamo deciso di individuare, non solo separandoli ma con il massimo margine possibile. I punti che risiedono sul margine sono chiamati support vector e sono gli unici che influenzano la scelta dei confini di decisione. I confini decisionali possono essere semplicemente lineari, quindi stiamo parlando di una SVM lineare, oppure se siamo in presenza di dati linearmente non separabili possiamo usare il

cosiddetto “kernel trick”, cioè portare i dati in uno spazio con una dimensione in più in cui questi saranno linearmente separabili. Per fare questo abbiamo bisogno di mappare le feature in uno spazio dimensionale superiore (nella pratica utilizzare delle “kernel function” per semplificare i calcoli), questo ovviamente implicherà che nella dimensione originale i confini decisionali non saranno più lineari. Questo modello è però utilizzato per effettuare classificazione, per questo per adattare il suo utilizzo abbiamo dovuto utilizzare un SVR o Support Vector Regression. Qui la differenza è che non si cerca più di dividere lo spazio in delle regioni da associare alle classi, ma si cerca di trovare l'iperpiano ottimale che contenga più punti e di stabilire i suoi confini decisionali, tali che questi individuino il massimo dell'errore ammesso da un punto per essere approssimato in maniera accettabile, potremmo dire che i confini decisionali rappresentano il nostro margine di tolleranza. Oltre a questi metodi si possono applicare anche modelli basati su reti neurali in fase di predizione, che però possono essere ampliati con backbone convoluzionali per operare anche feature extraction. Inoltre, viste le osservazioni già prodotte sul fatto che le varie immagini che andranno ad essere categorizzate saranno frame di un video, ci è sembrato logico tenere conto di questo rapporto temporale fra le immagini e quindi utilizzare delle Recurrent Neural Network.

Recurrent Neural Network

Le Recurrent Neural Network o RNN sono delle reti neurali particolarmente efficaci quando si vanno ad analizzare sequenze di dati. Sono una tecnologia non molto recente [67], ma che sta ottenendo ottimi risultati in alcuni ambiti di recente formazione, come speech recognition o nella traduzione di testi e con le dovute modifiche possono essere utilizzate per lo studio e l'analisi di video multimediali. Le RNN sono abili con le sequenze poiché riescono ad estrarre informazioni temporali dai dati, questo perché utilizzano informazioni ricavate nel passato per andare interpretare in maniera differente i dati che stanno analizzando. Ovviamente anche le reti neurali normali imparano, ma lo fanno in fase di training aggiornando i pesi della rete durante il processo di

learning. Questa componente funziona in maniera simile nelle RNN, con la differenza che nella fase di predizione ricordano delle informazioni che le aiutano nella produzione dell'output successivo. Per questo mentre in una rete neurale normale a un dato input dovrebbe corrispondere un dato output, la risposta di una RNN dipenderà non solo dall'input stesso, ma dai precedenti input di quella sequenza che sta analizzando. Gli hidden layer infatti ricevono due input, uno che è quello effettivamente immesso nella rete, l'altro che è lo stato, per l'appunto "hidden", prodotto dalla rete in precedenza. Per permettere questo è creata una connessione all'interno del neurone stesso, che non passa il suo output solamente al layer successivo come nelle normali reti, ma con quello che potremmo chiamare un loop, fornisce il suo output a sé stesso, modificato con una matrice di pesi, da abbinare all'input esterno che gli viene proposto in un istante temporale successivo. L'immagine può sicuramente aiutare a comprendere il concetto.

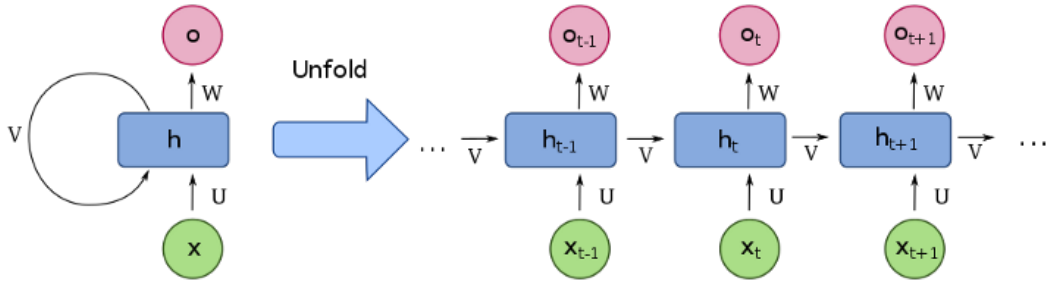

Figure 5.11. Neurone di una RNN prima e dopo aver aperto il loop sull'asse del tempo.
Credit to https://bit.ly/2PolA4S

A sinistra troviamo la rappresentazione statica del neurone, a destra la sua rappresentazione se prendiamo in considerazione anche l'asse temporale, andando quindi a sciogliere il loop. Le RNN sono perciò delle reti che si sviluppano su due piani, poiché essendo ricorrenti hanno anche una profondità dovuta alla lunghezza della sequenza di input che gli viene sottoposta. Questo ci fa riflettere e risulta quindi evidente che, come le reti neurali soffrono del già

descritto Vanishing Gradient che non permette l'allenamento di reti troppo profonde, allo stesso modo vi sarà una difficoltà di utilizzo delle RNN quando le finestre temporali che si vanno a guardare diventano troppo grandi. Per l'allenamento infatti viene utilizzata una variazione della backpropagation chiamata Backpropagation Through Time o BPTT. La BPTT non solo permette di aggiornare i pesi partendo dall'errore riscontrato, ma tiene anche conto della dimensione temporale della rete estendendo la derivata della funzione per ogni step della sequenza temporale che utilizziamo. Tutto questo processo implica che la profondità della rete avrà un limite non solo legato al numero di layer, ma anche al numero di passi temporali di una data sequenza, legami temporali lontani sarebbero troppo difficili da individuare e il problema della scomparsa o esplosione del gradiente si presenterebbe sicuramente. Quindi più due elementi sono lontani in una sequenza, meno la conoscenza di uno influenzerà l'analisi dell'altro, questo è il problema delle dipendenze "Long-Term", si dice quindi che le RNN hanno una "short-term memory". Per risolvere questo problema sono state introdotte delle variazioni della RNN, che modificando la struttura del neurone e di conseguenza il modo in cui gli hidden state vengono trasmessi a quest'ultimo, permettono di mantenere una memoria a lungo termine. L'immagine può essere utile da abbinare alla spiegazione, così da chiarificare i concetti trattati a breve, che saranno spiegati da un punto di vista concettuale e non strettamente matematico.

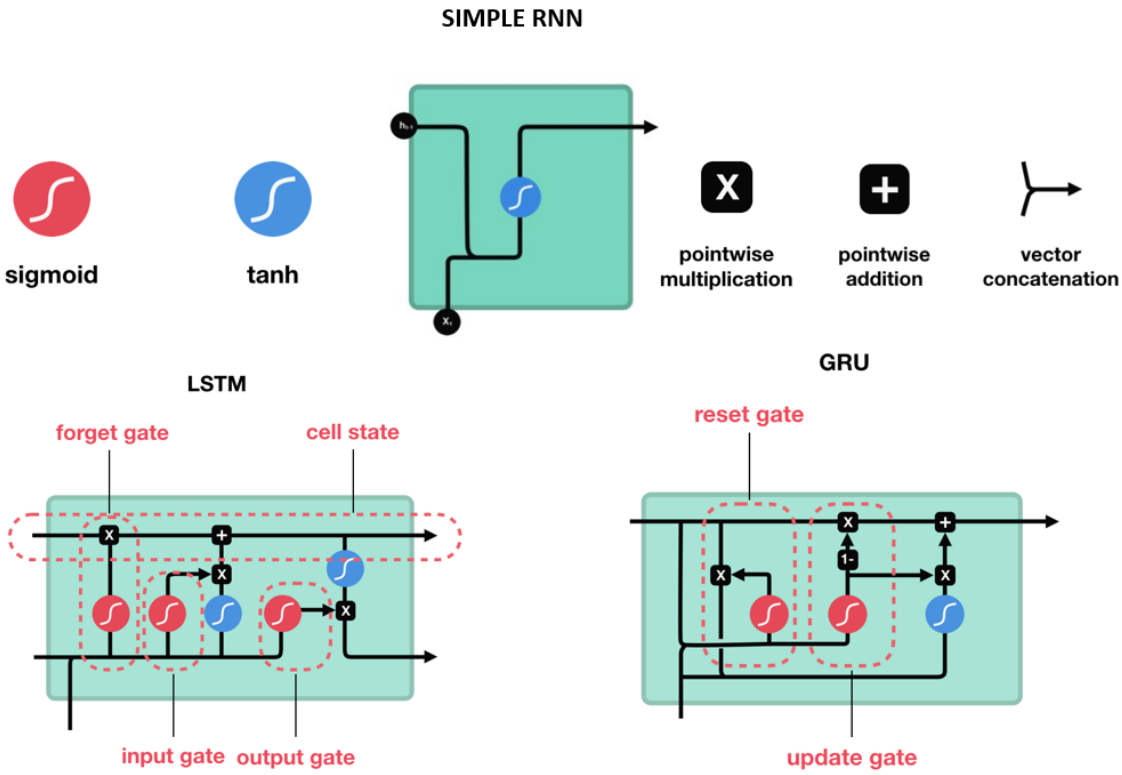

Figure 5.12. Struttura delle celle in una RNN semplice e nelle sue due variazioni: LSTM e GRU.
Modified from https://bit.ly/3fyfkSX

La rete più famosa è quella chiamata Long Short-Term Memory o LSTM, che, come suggerisce il nome, cerca di allungare la memoria della rete. Il neurone di una normale RNN è caratterizzato da un input esterno che viene sommato al precedente hidden state, il risultato passa poi attraverso un layer caratterizzato da una tangentoide che porta i valori tra -1 e 1. Questi valori sono mandati sia allo strato successivo, che al neurone stesso nell'istante temporale successivo. LSTM ha una struttura della cella più complicata, questo perché la cella dovrà gestire un ulteriore dato, rappresentato dalla memoria della cella. Questa non sarà semplicemente l'output del precedente istante temporale, che sarà comunque gestito, ma una sorta di memoria a lungo termine che rappresenterà quello che la cella ha bisogno di tenere in

memoria per prendere delle decisioni adatte. Per permettere la gestione di questi tre flussi di dati (cell state, hidden state precedente e dati di input) la LSTM si avvale di diverse tipologie di gate, che si occupano di gestire il modo in cui dati vengono trattati, conservati o scartati. Il forget gate serve a capire quali dati della memoria a lungo termine possano servire oppure no, basandoci su quelli che sono i dati di input e il precedente hidden state; per ottenere ciò, questi due dati passano attraverso una rete con una sigmoide come attivazione e sono poi moltiplicati con lo stato della cella. La sigmoide permette di portare i dati tra 0 e 1, così un dato moltiplicato per 0 sarà scartato, mentre uno moltiplicato per 1 sarà mantenuto. L'input gate permette di scegliere quali dati, tra quelli in ingresso e quelli dell'hidden state, possa essere utile aggiungere alla memoria della cella. I dati passano sia da una sigmoide, per decidere cosa ha importanza, sia da una tangente, per permettere alla rete di regolarsi grazie all'inserimento di numeri negativi, e poi vengono moltiplicati. La moltiplicazione permette di decidere cosa è meglio aggiungere alla memoria della cella, rispetto ai dati regolati dalla tangente. Infine, l'output gate permette di gestire il nuovo hidden state, moltiplicando lo stato della cella con i dati di partenza, l'utilizzo concettuale delle funzioni è lo stesso descritto sopra, tranne che la tangente in cui passa la memoria a lungo termine è questa volta solamente una funzione e non una rete. In questo modo creiamo una sorta di filtro che trasmette nell'hidden state, che ricordiamo verrà trasmesso anche al layer successivo della rete, solo le informazioni necessarie. Questa struttura permette a queste reti di imparare attraverso un gran numero di time steps, si parla di oltre mille, permettendo un utilizzo più adatto al mondo reale, dove il risultato di una certa azione è dilazionato nel tempo e non immediatamente successivo, per questo creare connessioni tra dati lontani tra loro può essere estremamente utile. Un'altra variante delle RNN più recente è la Gated Recurrent Unit o GRU. Anche questa tipologia di rete ha la stessa finalità di cancellare il problema "short-memory" permettendo connessioni tra elementi lontani nel tempo e si caratterizza per la composizione della sua cella, che presenta due

diversi tipi di gate rispetto alla LSTM. L'update gate aiuta il modello a determinare quanto e cosa delle informazioni passate attraverso l'hidden step debba essere portato avanti nel risultato. Il reset gate invece decide quanto e cosa delle informazioni passate debba essere dimenticato. In generale le GRU sono più veloci da allenare e hanno performance migliori in caso di sequenze non così lunghe, campo invece dove le LSTM performano meglio, a costo di un maggior tempo di training, la differenza risiede infatti nella presenza della memoria di cella in una rete rispetto all'altra.

Quindi, riassumendo, sono stati provati due diversi tipi di approccio, il primo che utilizza modelli più semplici solo per la fase di predizione basandosi sul vettore con i 68 landmarks, utilizzando per la regressione modelli come la SVR; mentre nel frattempo si allena una CNN su un numero minore di landmark. Il secondo è l'approccio che riunisce la fase di Feature Extraction e quella di Predizione in un unico modello, composto da una rete convoluzionale per l'estrazione di feature, che tra i layer fully connected inserisce dei layer LSTM o GRU, per tener conto delle caratteristiche temporali dell'input in fase di predizione.

Chapter 6

Conclusioni

Al termine del mio periodo di collaborazione in questo progetto, posso dire di aver non solo ottenuto dei risultati, ma anche di aver, insieme agli altri ragazzi, gettato le basi per il lavoro futuro. L'applicazione desktop è al momento funzionante e già installata sia nel Caserta Hub Lab, che nel centro riabilitativo Puzzle. Sono stati sviluppati in totale cinque esperimenti differenti e la metodologia di questi è stata testata sia dagli ideatori stessi, che li hanno identificati come aderenti a quella che era la loro idea di partenza, sia da alcuni pazienti o possibili partecipanti all'esperimento, che hanno fornito alcuni feedback agli operatori. Su queste opinioni e su altre piccole problematiche considerate come secondarie, lavoreranno alcuni colleghi nei prossimi mesi. Di sicuro interesse sarà la modalità di conservazione dei dati, in quanto a seconda dell'ambito vi saranno dei protocolli di privacy differenti da seguire. Mentre si dovrà lavorare per permettere l'inclusione e il funzionamento dei vari moduli che saranno sviluppati per l'utilizzo degli strumenti descritti nei capitoli precedenti. Gli esperimenti sono ovviamente stati testati solo per poterne giudicare il flusso, in quanto i dati per i biomarcatori non vengono ancora registrati nella versione dell'applicazione rilasciata nei due centri.

Per quanto riguarda l'algoritmo di Facial Emotion Recognition, come detto è stato trattato nell'ultimo periodo della mia tesi, collaborando con colleghi con scadenze successive alle mie. Per questo non risulta ancora terminato e funzionante, ma le tecnologie che sono state descritte nel capitolo precedente

sono state studiate e sono state tutte sperimentate oppure si trovano in corso di sperimentazione. Il grafico mostrato può spiegare in maniera più chiara i tre tipi di approccio affrontati fino ad ora.

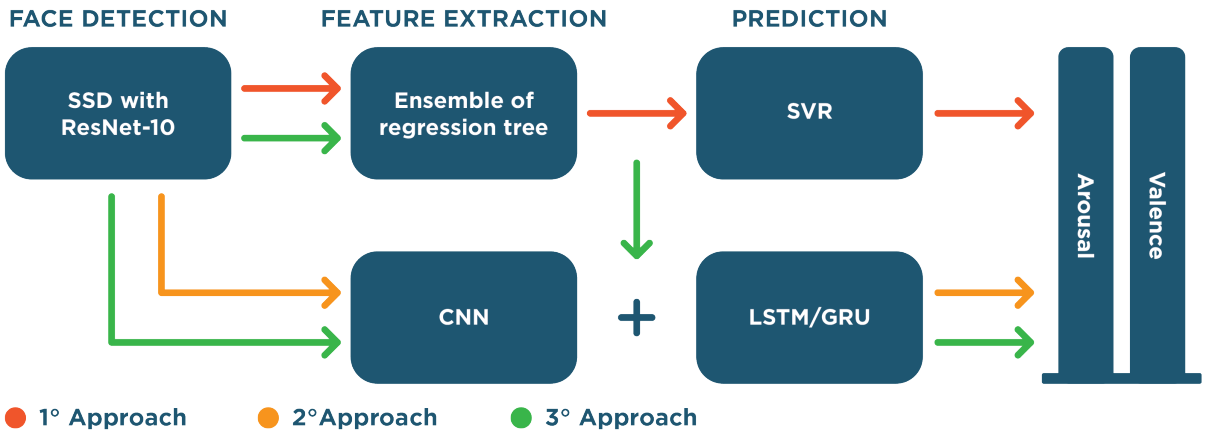

Figure 6.1. Schematizzazione dei tre diversi approcci investigati durante lo studio.

La parte di face detection, ossia quella a cui sono stato assegnato, è l'unica in cui al momento è stata fatta una scelta abbastanza definita. Le due tecnologie confrontate, come detto, sono state YOLOv3 e SSD Multibox, entrambe hanno ottenuto ottimi risultati, riuscendo a identificare il volto all'interno dell'immagine con grande facilità e anche in caso di condizioni ambientali non ottimali. Come detto però la nostra scelta deve tenere conto anche dei limiti hardware ai quali il nostro utilizzo è sottoposto, bisognava quindi cercare un buon tradeoff tra accuratezza e carico computazionale. Sicuramente il modello più leggero era l'"Haar Cascade", il classificatore basato sulle Haar-like feature, ma questo dimostrava delle prestazioni non ottimali, sia in caso di luce scarsa, sia nel caso la faccia non fosse correttamente allineata alla telecamera. Nel caso di un utilizzo unicamente in laboratorio la sua applicazione non sarebbe stata eccessivamente problematica, ma è bene ricordare che l'algoritmo di Facial Emotion Recognition sarà in comune anche con la Web App, quindi il detector dovrà riuscire a lavorare anche in condizioni non controllate come quelle di un laboratorio.

Come detto sia YOLOv3 che l'SSD utilizzato avevano ottime performance, essendo infatti modelli pensati per fare object detection di una gran quantità di oggetti in situazioni sicuramente più complesse. Il modello scelto è stato l'SSD con una backbone ResNet, poiché ha dimostrato di necessitare di un minor carico computazionale riuscendo a elaborare 30 fps anche su dei laptop non di prima categoria. Questo è dovuto probabilmente al fatto che mentre la YOLOv3 era un'architettura generica e riadattata per lo specifico scopo, l'SSD fornita dalla libreria OpenCV era una versione molto più leggera delle SSD con backbone ResNet presenti in letteratura e solitamente caratterizzate da 101 layer. La particolarità della ResNet, che è già stata ampiamente nominata e che ha permesso a quest'ultima di rivoluzionare il modo di fare Deep Learning, è la sua capacità di eliminare il problema del Vanishing Gradient anche in reti molto profonde. Questo è stato reso possibile grazie al concetto di Residual Block, un nuovo tipo di layer tipico della ResNet, questo è un layer particolare caratterizzato da una "Skip Connection" che spesso viene definita scorciatoia. Tramite questa scorciatoia in cui viene applicata la funzione identità, l'output di layer precedenti si va a sommare a quello di layer successivi, trasmettendo informazioni in parti più profonde della rete senza che queste subiscano l'effetto dei filtri convoluzionali. Concettualmente ricorda quello che accade nelle LSTM, con la differenza che sulla Skip Connection non è presente nessun gate che applichi funzioni tangente o sigmoide.

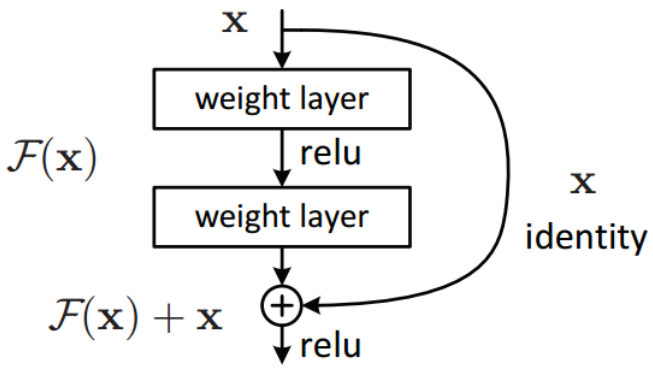

Figure 6.2. Rappresentazione di un Residual Block, elemento fondante delle reti ResNet

La SSD ResNet-10 è stata inizializzata con i pesi forniti nella repository ufficiale di OpenCV, si è provato anche a aumentare le performance di quest'ultima provando utilizzare il nostro dataset, ma i risultati sono rimasti grossomodo gli stessi, si è quindi deciso di mantenere i pesi di partenza. La rete è strutturata per trattare immagini 300x300, come spiegato nella sezione del Face Detection sarebbe possibile passare ad immagini a risoluzione più alta, ma ne risentirebbero notevolmente le prestazioni. Inoltre, il problema principale a cui si cercava di porre rimedio, quello dell'individuazione di oggetti piccoli e in gruppo, non è qui presente in quanto si cerca di individuare un solo volto umano. Infine, in accordo con i team degli altri gruppi, si è deciso di ridurre il carico computazionale dell'intero modello, operando una riduzione delle immagini da analizzare. Per fare questo si è sia ridotto il numero di frame effettivamente mandati alle fasi successive del processo, sia si cercato di eliminare quei frame che non sarebbero stati eccessivamente utili al riconoscimento delle emozioni, ponendo particolare attenzione sulle immagini in cui anche se il viso veniva effettivamente riconosciuto, la rotazione della testa era eccessiva tanto da non permettere la vista di uno dei due occhi, che abbiamo detto essere fondamentali in fase di decisione.

A questo punto il processo si è differenziato a seconda dei vari approcci. Il primo è stato quello di utilizzare 2 modelli più semplici tenendo separati gli stati di Feature Extraction e Predizione (percorso rosso sull'immagine), dall' Ensemble of regression tree sono stati tirati fuori dei vettori di lunghezza 136, contenenti x e y dei 68 landmarks individuati. Il vantaggio della produzione dei landmarks è anche quello di avere un maggior potere esplicativo, in quanto lavorando anche in contatto con l'ambito medico, avere la possibilità di giustificare le nostre scelte con dei dati in mano, invece di affidarci ad un approccio black box come quello fornito dalle CNN, può avere un grande valore per permettere ai neurologi di avere più fiducia in quelli che sono i risultati che gli vengono forniti. Su questi vettori è stato applicato SVR per identificare i valori di attivazione e valenza emotiva e ricavare i risultati riguardo l'errore riscontrato. Utilizzando il MSE come metrica principale, ci si è basati sui risultati ottenuti nella Aff-Wild Challenge del 2017, condotta

appunto sul dataset Aff-Wild. I modelli finalisti hanno tutti ottenuto intorno allo 0.1 di MSE, il nostro modello non aspira ad una precisione così elevata viste le limitazioni che ci siamo imposti, ma risultati compresi tra lo 0.2 e lo 0.3 non ci hanno lasciato soddisfatti. Per questo abbiamo cominciato ad analizzare altri approcci, prendendo spunto dal modello sviluppato specificatamente per questa task chiamato AffWildNet [64]. Come si può vedere nello schema, le fasi di Feature Extraction e Predizione vengono unificate, poiché sono entrambe svolte da una rete convoluzionale con annessi pooling layer, che pone tra il primo e il secondo fully connected layer dei layer RNN con delle unità GRU, per poter cogliere i legami temporali tra i vari frame. Questo è lo schema del paper di presentazione del modello.

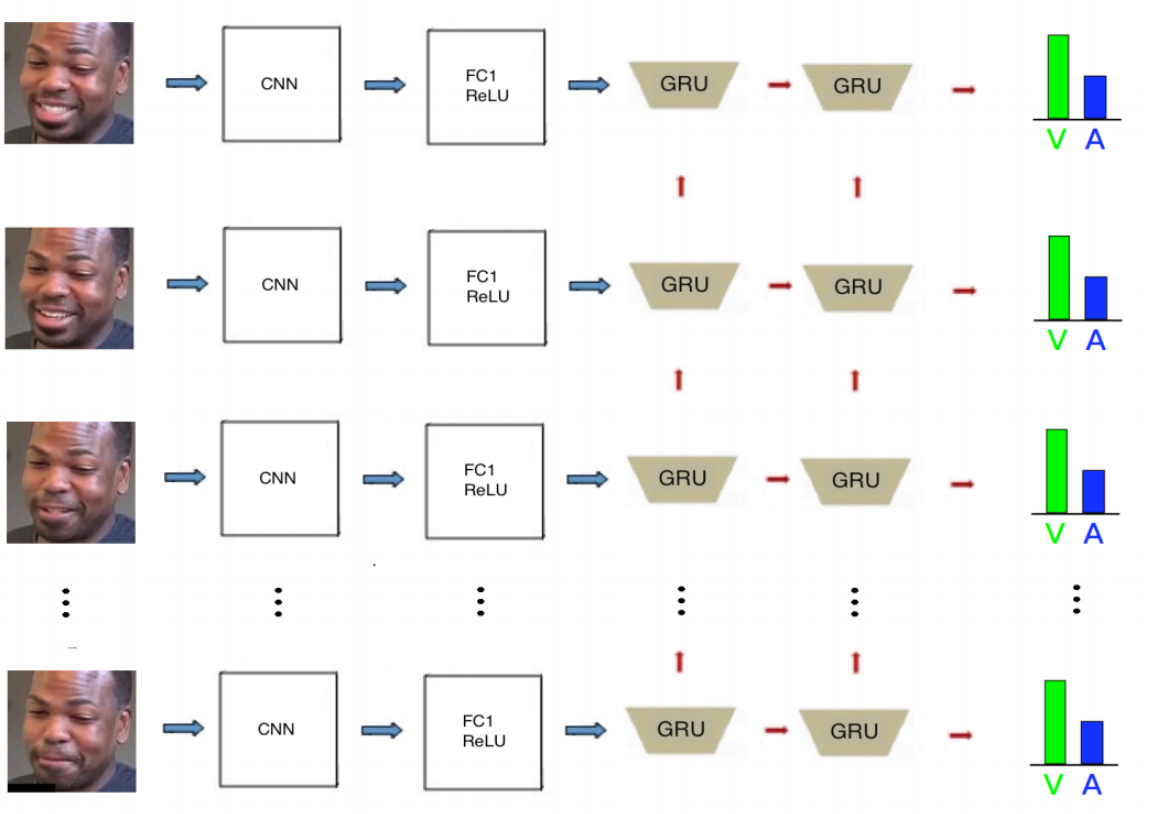

Figure 6.3. Schematizzazione concettuale della struttura della rete AffWildNet.
Credit to [64]

Nell'immagine non è rappresentato l'ultimo fully connected layer, cioè quello da cui poi vengono estratte le predizioni su attivazione e valenza emotiva. Il secondo (strada gialla) e terzo (strada verde) approccio stanno venendo sviluppati in parallelo e sembrano dare risultati incoraggianti rispetto a quelli ottenuti fino ad ora. Abbiamo cercato di utilizzare sia unità GRU che LSTM nei layer RNN, ma siamo più propensi per la prima possibilità in quanto si è visto che i moduli GRU sono più leggeri e veloci, anche se permettono connessioni meno lontane tra dati, fattore che nel nostro specifico campo applicativo non dovrebbe rappresentare un enorme problema. Per quanto riguarda la backbone CNN utilizzata, si è visto come una ResNet-50 ottenga ottimi risultati, ma architetture meno profonde come VGG-16 permettano di mantenere alte le performance ad un minor costo computazionale, per questo sono la nostra prima opzione al momento. Per quanto riguarda il terzo approccio è quello che nel paper ha dimostrato fornire i risultati migliori, consiste nel permettere la lavorazione in parallelo sia dell'Ensemble of regression tree che della CNN. Il vettore di landmarks identificato dal primo modello sarà fornito al primo fully connected layer della rete, che quindi si allenerà non solo sulle feature estratte dai layer convoluzionali, ma anche sui landmarks stessi. Il motivo per cui non abbiamo adottato direttamente questo approccio è che, come si può immaginare, è quello a più alto carico computazionale poiché va ad utilizzare un gran numero di modelli. Per questo mentre si provano i modelli, il gruppo sta anche cercando un modo per diminuire ulteriormente i frame da analizzare, concentrandosi in particolare nell'eliminare la ridondanza di questi ultimi, che in molti casi, come per esempio nel mantenimento di un'espressione neutra da parte del partecipante, potranno non essere sottoposti al modello ma saranno scartati in una fase più o meno precoce del processo. Al momento si sta valutando di basare il criterio di somiglianza sulla posizione dei landmarks interni al viso, ma le modalità specifiche sono ancora in sperimentazione e i risultati fin troppo acerbi.

Si può quindi ipotizzare che nell'arco di un paio di mesi questi approcci da me descritti saranno stati sperimentati e testati accuratamente e saremo

davanti o al modello definitivo, oppure a delle nuove proposte di modello che al momento non sono state ancora definite. Lascio infine nel prossimo paragrafo una serie di possibili miglioramenti o modifiche da poter esplorare sia per ottenere risultati migliori, sia perché di interesse nell'ambito di questa ricerca.

6.1 Future works

Parlare in questo contesto di idee su come poter migliorare le cose in futuro, o comunque di altre strade da esplorare, può sembrare inutile, poiché come ampiamente spiegato il progetto è ancora in corso e in continua trasformazione. Quello che posso limitarmi a fare è però elencare alcune strade che discutendo con i colleghi ci sono sembrate percorribili e che meritano sicuramente un approfondimento dal momento che l'applicazione sarà pienamente funzionante. Prima di provare ad applicare queste altre idee bisognerà, come detto, continuare a migliorare il modello al quale siamo giunti fino ad arrivare a risultati migliori e in linea con quello che è almeno lo stato dell'arte, in accordo con quelle che sono le nostre potenzialità hardware. Questo vuol dire che queste idee non saranno effettivamente sperimentate nell'immediato, ma sicuramente potrebbero essere il punto di partenza per un ulteriore lavoro.

- **Inserimento di tutti i biomarcatori discussi.** Come ampiamente spiegato nel capitolo 2, molte sono le tecnologie che si è pensato a livello teorico di poter inserire nella valutazione degli esperimenti. Al momento, volendo procedere in maniera modulare, si sta inserendo il modello di Face Emotion Recognition trattato nel precedente capitolo, ma dei colleghi sono già a lavoro per integrare questo algoritmo con i risultati di Eye-Tracker e GSR. Queste due tecnologie, infatti, sono spesso utilizzate insieme così da fornire risultati più affidabili e significativi. L'Eye-Tracker come si è visto permette di comprendere quali zone dell'immagine raccolgono l'attenzione dello spettatore, permettendo di capire anche gli elementi che nel bene o nel male hanno un impatto maggiore su di lui. Il GSR può fornire quello che è indicato come biofeedback,

permettendo di comprendere la reazione psicologica del partecipante allo stimolo su cui l'Eye-Tracker evidenzia la sua concentrazione, oppure i livelli di stress che sono trasmessi dal completare determinate task. Questa accoppiata in particolare sarà estremamente utile per gli esperimenti di neuromarketing per i quali è stata sottolineata l'importanza dell'utilizzo di queste tecnologie, in particolare per gli esperimenti di web design. Per quanto riguarda gli esperimenti neuroriabilitativi risulterà invece di primaria importanza l'implementazione di un modello che tenga in considerazione i risultati forniti dall'elettroencelografia, che permetta di osservare l'evoluzione o le mancanze del paziente in presenza di specifiche emozioni che ha difficoltà ad esprimere. Infine, altri biomarcatori che possono essere inseriti anche senza necessitare di un grosso effort sono il controllo del battito cardiaco e del ritmo respiratorio, due feedback estremamente semplici e che singolarmente non hanno grande contenuto informativo, ma che uniti come supporto alle tecnologie sopracitate possono contribuire ad identificare variazioni nella condizione dell'individuo, nonché l'insorgenza di stati d'animo di ansia o stress provocati dagli esperimenti.

- **Speech Emotion Recognition.** La voce è per l'uomo uno dei modi più immediati per esprimere le proprie emozioni insieme alle espressioni facciali. Attraverso varie componenti come intonazione e ritmo della voce utilizzata, le persone tendono, spesso inconsciamente, ad esprimere quello che è il loro stato emotivo permettendo alla persona che hanno di fronte di empatizzare con loro. Gli studi in questo ambito sono molteplici e il tentativo di produrre dei classificatori prosodici (dove la prosodia è il "Complesso delle leggi che regolano l'intonazione, l'accentazione e il tono di una lingua, spec. per quanto riguarda la versificazione" [65]) procede da circa 40 anni con risultati alterni e spesso incoraggianti. Si sta infatti cercando di applicare queste metodologie in molti campi, come, ad esempio, quello degli assistenti vocali così da permettere una migliore interazione con l'essere umano, poiché non si andrà più solo a comprendere a livello semantico le richieste che vengono fatte, ma si potrà instaurare

un dialogo più umano comprendendo le emozioni della persona che interagisce con la macchina. Un grande problema di queste tecnologie risiede nel fatto che risulta assolutamente fattibile riconoscere la presenza di una data emozione in dati audio in cui si vuole esplicitarla in maniera chiara, come ad esempio in monologhi di attori [66]. Più complicata è la distinzione sia quando si vogliono distinguere un maggior numero di emozioni differenti, sia quando il soggetto da cui si raccolgono i campioni ha più modi di poter esprimere il proprio stato emozionale, come ad esempio appunto l'espressione facciale. Anche in questa tipologia di analisi, ritroviamo una non uniformità nella rilevazione delle emozioni, alcune, infatti, come la rabbia sono rappresentate in maniera più evidente attraverso la voce rispetto ad altre con un minore impatto, come ad esempio la tristezza. Sembra quindi una via da poter sicuramente esplorare soprattutto se messa in combinazione con le altre metodologie sviluppate, permettendo una maggiore confidenza nelle predizioni e un valore in più da poter analizzare. In particolare, molti sono gli studi dell'applicazione di questa metodologia utilizzando non le emozioni in maniera distinta e indipendente, ma all'interno del modello circomplesso, ossia nello spazio bidimensionale di attivazione e valenza emozionale. Questo ci permette di inserire questa tecnica all'interno del modello che abbiamo pensato e stiamo continuando a sviluppare, senza grandi cambiamenti nel modo in cui le emozioni vengono identificate dagli altri algoritmi, ma anzi permettendo una collaborazione proficua tra gli stessi. L'unico cambiamento necessario sarà lo sviluppo di nuovi esperimenti che coinvolgano la voce del partecipante come risposta a degli stimoli, questa può essere quindi un'indicazione fornita al gruppo di ricerca che nei prossimi mesi ha comunque intenzione di pensare a nuove tipologie di sperimentazione su cui testare le tecnologie che stiamo sviluppando.

- **Multi-Task Learning.** Oltre ad approfondire e tentare di utilizzare in maniera adeguata alcune delle tecniche descritte in letteratura e che hanno dimostrato un buon rendimento in una delle tre fasi identificate, si potrebbe anche pensare di cambiare totalmente approccio. L'idea

prende spunto dal paper [68], molto recente, che mostra come invece di dividere le varie fasi come abbiamo deciso di fare in questo lavoro, si può provare a produrre un modello che utilizzi il concetto, già introdotto nel Capitolo 4, di Multi-Task Learning. La funzione di Loss dell'unica CNN utilizzata, basata su YOLOv2, è formata da quattro componenti: la loss sulla posizione del bounding box, la loss sulla probabilità che il bounding box contenga effettivamente una faccia, la loss sulla predizione delle emozioni come categorie discrete e la loss sulla predizione di attivazione e valenza emotiva. Quindi questo modello non solo va ad unire la Face Detection con la Face Emotion Recognition, ma permette di andare ad identificare le emozioni con entrambi i metodi di individuazione riconosciuti dalla letteratura contemporaneamente. Si è visto come questo approccio permetta alla rete di generalizzare meglio e come la possibilità di lavorare a più task contemporaneamente permetta che tutte queste traggano beneficio reciproco. Queste osservazioni sono confermate non solo dai buoni risultati ottenuti in fase di test, ma anche da alcuni test cross-dataset che sono stati fatti per testare la generalizzazione della rete. La rete utilizzata era inoltre relativamente poco profonda, circa 10 layer, fattore che permette quindi l'utilizzo in real-time senza grandi problematiche. Un approccio ibrido con quello utilizzato in questo lavoro potrebbe, quindi, davvero migliorare notevolmente i risultati ottenuti e probabilmente anche le performance dei modelli nelle loro applicazioni finali.

- **Penalizzare le emozioni.** Come ampiamente descritto vi sono alcuni tipi di emozione che sono più difficilmente rilevabili a seconda della loro componente di attivazione e valenza emotiva. Si è infatti detto di come le emozioni negative siano di più difficile individuazione e mediamente presentino un'attivazione mediamente minore. Inoltre, proprio le emozioni con un basso livello di attivazione sono più difficilmente riconoscibili e individuabili. Basta osservare anche solo la distribuzione delle immagini all'interno del dataset AffectNet utilizzato, che presenta delle carenze

nelle zone dello spazio a bassa attivazione e valenza negativa e al contrario presenta il picco di popolazione nel quadrante caratterizzato da alta attivazione e valenza positiva. Per cominciare può essere utile investigare maggiormente in questa carenza di informazioni, per capire se sia quest'ultima a provocare la difficoltà dei classificatori in quelle zone dello spazio o se sia effettivamente complicato individuare delle emozioni con quelle caratteristiche e questo provochi una carenza effettiva di immagini, poiché ad esempio potrebbero essere rappresentate da caratteristiche espressive meno determinanti. Una volta chiarita la questione, si potrebbe comunque cercare di forzare il nostro modello a compiere uno sforzo nella direzione di questi stati emozionali, in particolare ad esempio penalizzando maggiormente nella funzione di Loss gli errori sulle immagini appartenenti a quella porzione di spazio. Questo potrebbe peggiorare notevolmente le prestazioni del classificatore, ma anche permettere una migliore individuazione in alcune tipologie di emozioni che al momento vengono a fatica identificate, ma che non pesano eccessivamente sui risultati numerici perché, appunto, caratterizzate da una popolazione molto bassa.

Bibliography

[1] McClure S.M, Li J., Tomlin D., Cypert K.S., Montague L.M. e Montague P.R.(2004), "Neural correlates of behavioral preference for culturally familiar drinks" *"Neuron", n.44, pp.379-387*

[2] The New York Times, 2009, "Lab Watches Web Surfers to See Which Ads Work"
http://www.nytimes.com/2009/07/27/technology/27disney.html

[3] Nicolas Hamelin , Park Thaichon , Christopher Abraham , Nicholas Driver , Joe Lipscombe , Jayarethanam Pillai, 2020 "Storytelling, the scale of persuasion and retention: A neuromarketing approach"

[4] Wikipedia, "Neuroriabilitazione"
https://it.wikipedia.org/wiki/Neuroriabilitazione

[5] SRLabs, "Che cos'è l'eye tracking"
https://www.srlabs.it/che-cose-leye-tracking/

[6] Medium, "How to choose a programming language for a project"
https://medium.com/better-programming/how-to-choose-a-programming-language-for-a-project-7c7a3e5a4de6

[7] Python.it, "About Python"
https://www.python.it/about/

[8] Wikipedia, "Python"
https://en.wikipedia.org/wiki/Python_(programming_language)

[9] K. Jarrod Millman, Michael Aivazis, "Python for Scientists and Engineers"

[10] O'Reilly, "Python Cookbook"

[11] Fernando Pérez, Brian Granger, John Hunter, "Python: An Ecosystem

for Scientific Computing"

[12] Abhinav Nagpal, Goldie Gabrani, "Python for Data Analytics, Scientific and Technical Applications"

[13] Vidya M. Ayer, Sheila Miguez, Brian H. Toby, "Why scientists should learn to program in Python"

[14] Neudeep Technology Blog
https://www.blog.neudeep.com/howto/advantages-and-disadvantages-of-tkinter-and-wxpython/231/

[15] Jonathan W. Peirce, "Generating stimuli for neuroscience using PsychoPy"

[16] David Bridges, Alain Pitiot, Michael R. MacAskill, Jonathan W. Peirce, "The timing mega-study: comparing a range of experiment generators, both lab-based and online"

[17] OpenCV
https://opencv.org/about/

[18] Pulli, Kari; Baksheev, Anatoly; Kornyakov, Kirill; Eruhimov, Victor, "Realtime Computer Vision with OpenCV"

[19] VideoLan Org, "LibVLC"
https://www.videolan.org/vlc/libvlc.html

[20] "LibVLC Python Bindings"
https://wiki.videolan.org/Python_bindings/

[21] Github CEF Python
https://github.com/cztomczak/cefpython

[22] Chromium Embedded Framework
https://bitbucket.org/chromiumembedded/cef/src/master/

[23] David Marr (1982), "Vision: A Computational Investigation into the Human Representation and Processing of Visual Information"

[24] P. Ekman, W.V. Friesen, (1971), "Constants across cultures in the face and emotion"

[25] Sarah-Jane Vick, Waller, Bridget M., Parr, Lisa A., Smith Pasqualini, Marcia C., Bard, Kim A. (15 December 2006). "A Cross-species Comparison of Facial Morphology and Movement in Humans and Chimpanzees

Using the Facial Action Coding System (FACS)"

[26] Mitchell, T, (1997), "Machine Learning" p. 2

[27] Judea Pearl, Dana Mackenzie, "The Book of Why: The New Science of Cause and Effect" (2018 ed.)

[28] "AN EMPIRICAL SCIENCE RESEARCH ON BIOINFORMATICS IN MACHINE LEARNING – Journal". Retrieved 28 October 2020

[29] Vivien Cabannes, Alessandro Rudi, Francis Bach, (2021), "Disambiguation of weak supervision with exponential convergence rates" arXiv:2102.02789.

[30] "What is semi-supervised machine learning?" https://bdtechtalks.com/2021/01/04/semi-supervised-machine-learning/

[31] Sebastian Ruder, An Overview of Multi-Task Learning in Deep Neural Networks, arXiv:1706.05098

[32] The Complete Guide to Artificial Neural Networks: Concepts and Models https://missinglink.ai/guides/neural-network-concepts/complete-guide-artificial-neural-networks/

[33] David E. Rumelhart, Geoffrey E. Hinton, Ronald J. Williams, "Learning representations by back-propagating errors"

[34] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks"

[35] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, “Deep Residual Learning for Image Recognition”

[36] Kunihiko Fukushima, (1980), "Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position" , doi:10.1007/BF00344251

[37] A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[38] Sergey Ioffe, Christian Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", arXiv:1502.03167

[39] T. Huang, "Computer Vision : Evolution And Promise" doi:10.5170/CERN-1996-008.21

[40] Jonathan Long, Evan Shelhamer, Trevor Darrell, "Fully Convolutional Networks for Semantic Segmentation", arXiv:1411.4038

[41] Paul Viola, Michael Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", DOI: 10.1109/CVPR.2001.990517

[42] Kaiming He, Georgia Gkioxari, Piotr Dollár, et al.,"Mask R-CNN", (2017), arXiv: 1703.06870

[43] Jonathan Posner, James A. Russell, Bradley S. Petersona, "The circumplex model of affect: An integrative approach to affective neuroscience, cognitive development, and psychopathology"

[44] Russell JA, "A circumplex model of affect" , Journal of Personality and Social Psychology. 1980

[45] Shan Li, Weihong Deng, "Deep Facial Expression Recognition: A Survey"

[46] Ali Mollahosseini, Behzad Hasani, Mohammad H. Mahoor, "AffectNet: A Database for Facial Expression, Valence, and Arousal Computing in the Wild"

[47] D. Kollias, et. al. "Recognition of affect in the wild using deep neural networks", CVPRW, 2017

[48] S. Zafeiriou, et. al. "Aff-Wild: Valence and Arousal in-the-wild Challenge", CVPRW, 2017

[49] D. Kollias, et. al.: "Deep Affect Prediction in-the-wild: Aff-Wild Database and Challenge, Deep Architectures, and Beyond". International Journal of Computer Vision (2019)

[50] Sebastian Handrich, Laslo Dinges, Ayoub Al-Hamadi, Philipp Werner, Zaher Al Aghbari, "Simultaneous Prediction of Valence/Arousal and Emotions on AffectNet, Aff-Wild and AFEW-VA"

[51] Viola and Jones, "Rapid object detection using a boosted cascade of

simple features", Computer Vision and Pattern Recognition, 2001

[52] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, "You Only Look Once: Unified, Real-Time Object Detection", 2015

[53] Redmon, Joseph and Farhadi, Ali, "YOLO9000: Better, Faster, Stronger" arXiv:1612.08242, 2016

[54] Redmon, Joseph and Farhadi, Ali, "YOLOv3: An Incremental Improvement", 2018

[55] Tsung-Yi Lin et al., "Feature Pyramid Networks for Object Detection", arXiv:1612.03144

[56] Object detection with SSD by Jonathan Hui https://jonathan-hui.medium.com/ssd-object-detection-single-shot-multibox-detector-for-real-time-processing-9bd8deac0e06

[57] Wei Liu et al., "SSD: Single Shot MultiBox Detector" , 2016

[58] Shan Li, Weihong Deng, "Deep Facial Expression Recognition: A Survey"

[59] C. Sagonas, E. Antonakos, G, Tzimiropoulos, S. Zafeiriou, M. Pantic. "300 faces In-the-wild challenge: Database and results"

[60] Vahid Kazemi, Josephine Sullivan, "One Millisecond Face Alignment with an Ensemble of Regression Trees"

[61] Chih-Fan Hsu et al., "A Detailed Look At CNN-based Approaches In Facial Landmark Detection"

[62] Kanade, T., Cohn, J. F., Tian, Y. (2000). "Comprehensive database for facial expression analysis"

[63] Lucey, P., Cohn, J. F., Kanade, T., Saragih, J., Ambadar, Z., Matthews, I. (2010). "The Extended Cohn-Kanade Dataset (CK+): A complete expression dataset for action unit and emotion-specified expression"

[64] Dimitrios Kollias et al. , "Deep Affect Prediction in-the-wild: Aff-Wild Database and Challenge,Deep Architectures, and Beyond", 2019

[65] https://dizionari.repubblica.it/Italiano/P/prosodia.html

[66] Anton Batliner et al., "The Recognition of Emotion", DOI: 10.1007/978-3-662-04230-4_9

[67] Williams, Ronald J.; Hinton, Geoffrey E.; Rumelhart, David E., 1986., "Learning representations by back-propagating errors"

[68] Handrich et al., "Simultaneous Prediction of Valence/Arousal and Emotions on AffectNet, Aff-Wild and AFEW-VA"