POLITECNICO DI MILANO
Corso di Laurea in Ingegneria Informatica
Dipartimento di Elettronica e Informazione

REALIZZAZIONE E VALIDAZIONE DI UN SISTEMA PER LA CLASSIFICAZIONE DI MOVIMENTI IMMAGINARI IN UNA INTERFACCIA CERVELLO-COMPUTER

AI & R Lab
Laboratorio di Intelligenza Artificiale
e Robotica del Politecnico di Milano

Relatore: Ing. Matteo Matteucci
Correlatore: Ing. Rossella Blatt

Tesi di Laurea di:
Zennaro Fabio Massimo, matricola 716847

Anno Accademico 2009-2010

Indice

1 Introduzione **7**
- 1.1 La ricerca nel campo delle interfacce cervello-computer 8
- 1.2 Obiettivo della tesi 8
- 1.3 Struttura della tesi 10

2 Cenni di biologia **12**
- 2.1 Cervello 12
- 2.2 Neuroni 14
- 2.3 Sinapsi 15
- 2.4 Corteccia motoria 18

3 Registrazione dell'attività neurale **21**
- 3.1 Tecniche di registrazione dell'attività neurale 21
  - 3.1.1 Confronto tra le tecniche di registrazione dell'attività neurale 22
- 3.2 Elettroencefalografia 26
  - 3.2.1 Strumentazione 26
  - 3.2.2 Montaggio 27
  - 3.2.3 Funzionamento 28
  - 3.2.4 Ritmi elettroencefalografici 30

4 Interfacce cervello-computer e stato dell'arte **33**
- 4.1 Definizione di interfacce cervello-computer 33
- 4.2 Le interfacce cervello-computer come skill 35
- 4.3 Interfacce cervello-computer e applicazioni di interfacce cervello-computer 36
- 4.4 Principali destinatari delle interfacce cervello-computer 36
- 4.5 Classificazione delle interfacce cervello-computer 38
  - 4.5.1 Classificazione in base alla posizione dei dispositivi di acquisizione 38
  - 4.5.2 Classificazione in base al controllo nervoso-muscolare dell'utente 39
  - 4.5.3 Classificazione in base alla modalità di attivazione 39
  - 4.5.4 Classificazione in base ai segnali utilizzati 40

4.5.5 Classificazione in base alle applicazioni supportate 42
4.6 Stato dell'arte delle interfacce cervello-computer 43
4.6.1 Interfacce cervello-computer non-invasive basate su ritmi sensomotori (SMRs) 43
4.6.2 Interfacce cervello-computer non-invasive basate su slow cortical potentials (SCPs) 44
4.6.3 Interfacce cervello-computer non-invasive basate su P300 45
4.6.4 Interfacce cervello-computer non-invasive basate su visual evoked potentials (VEPs) 45
4.6.5 Interfacce cervello-computer invasive 46
4.7 Ritmi sensomotori 47
4.8 Criteri di valutazione delle performance delle interfacce cervello-computer 48
4.8.1 Metodi elementari per la valutazione delle performance di un'applicazione di un'interfaccia cervello-computer 49
4.8.2 Metodi alternativi per la valutazione delle performance di un'interfaccia cervello-computer 51
Signal-to-Noise Ratio (SNR) 52
Determination Coefficient (r^2) 52

5 Analisi e classificazione del segnale per motor imagery 55
5.1 Elaborazione del segnale 57
5.1.1 Preprocessing del segnale 57
5.2 Analisi del segnale 58
5.2.1 Discriminazione degli artefatti 58
5.2.2 Normalizzazione 61
5.2.3 Filtraggio spaziale 61
Common Average Reference 61
Small Laplacian Filter 62
Large Laplacian Filter 63
Confronto tra i metodi di filtraggio spaziale 65
5.2.4 Selezione dei canali e delle frequenze 65
5.2.5 Analisi in frequenza del segnale 66
Trasformata di Fourier 66
Trasformata discreta di Fourier 66
Trasformata discreta di Fourier veloce 67
Stima spettrale autoregressiva 67
Entropia 67
Entropia relativa 69
Teorema della distribuzione della massima entropia 69
Stima spettrale 70
Metodo del periodogramma 70
Teorema della massima entropia di Burg 71
5.2.6 Calcolo delle feature 72
Calcolo manuale delle feature 74
Calcolo automatico delle feature 75

5.2.7 Riduzione della dimensionalità . . . 75
Feature selection . . . 76
Fisher Rank . . . 79
Wilcoxon test . . . 80
Sequential Selection . . . 81
Algoritmi genetici . . . 82
Feature projection . . . 85
Linear Discriminant Analysis . . . 86
Principal Component Analysis . . . 88
5.2.8 Classificazione . . . 89
Classificatore lineare standard . . . 92
Classificatore quadratico . . . 93
Classificatore KNN . . . 94
Classificatore ad albero . . . 95
Classificatore SVM . . . 96
5.2.9 Feedback . . . 98

6 Sistema sviluppato e sua validazione 100
6.1 Configurazione sperimentale . . . 101
6.1.1 Acquisizione del segnale . . . 101
6.1.2 Elaborazione del segnale . . . 102
6.2 Protocollo di laboratorio . . . 102
6.3 Sistema di analisi e classificazione dei dati . . . 104
6.4 Risultati . . . 108
6.4.1 Risultati delle performance delle sessioni di motor imagery senza feedback . . . 108
6.4.2 Risultati delle performance delle sessioni di motor imagery con feedback . . . 109
6.4.3 Risultati delle performance dei diversi classificatori . . . 111
6.4.4 Validazione statistica delle performance dei diversi classificatori . . . 116
Test di Friedman . . . 121
Test di Nemenyi . . . 122
Test di Holm . . . 123
6.4.5 Impatto della generazione e della selezione automatica delle feature . . . 123

7 Discussione 130
7.1 Discussione . . . 130
7.2 Sviluppi futuri . . . 132
7.2.1 Ulteriori sviluppi durante lo stage presso la NTT . . . 133

A Dimostrazioni 134
A.1 Determination Coefficient (r^2) . . . 134
A.2 Analisi in frequenza . . . 139
A.2.1 Teorema della diseguaglianza di Jensen . . . 139

A.2.2 Teorema della diseguaglianza dell'informazione 140
A.2.3 Teorema della distribuzione della massima entropia 142

Bibliografia **144**

Capitolo 1

Introduzione

Un'interfaccia cervello-computer è un dispositivo progettato per permettere una comunicazione diretta tra una macchina e un utente umano; nell'accezione più comune si tratta di un dispositivo in grado di monitorare l'attività celebrale di un utente e utilizzare determinati segnali per interpretarne e attuarne la volontà.

In maniera schematica, un'interfaccia cervello-computer può essere rappresentata come un sistema a blocchi: il segnale generato dal cervello dell'utente è registrato da un opportuno modulo di acquisizione del segnale; questo segnale è quindi trasmesso a un modulo di analisi del segnale, che operando e classificando i dati ricevuti è in grado di estrarre informazione utile dal segnale grezzo registrato dal cervello dell'utente; infine, l'informazione utile estratta durante l'analisi del segnale è utilizzata per produrre un output che è reinviato all'utente come feedback (vedi Figura 1.1).

Le interfacce cervello-computer rappresentano un paradigma originale di comunicazione tra uomo e macchina, privo di interazioni fisiche dirette e fondato esclusivamente sugli stati mentali dell'utente. In quanto tali, le interfacce cervello-computer possono costituire un mezzo di comunicazione inestimabile per tutti quei soggetti che, a causa di malattie o incidenti, hanno perso il controllo dei propri muscoli volontari; nei casi più gravi, un'interfaccia cervello-computer potrebbe effettivamente essere l'unico modo in cui pazienti affetti da forme estreme di paralisi potrebbero comunicare con il mondo esterno; la

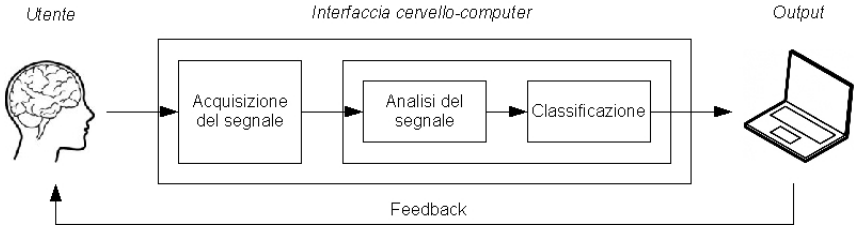

Figura 1.1: Schema a blocchi di un'interfaccia cervello-computer.

maggior parte dei dispositivi moderni per assistere persone affette da paralisi è infatti sviluppata assumendo l'ipotesi che questi pazienti abbiano ancora una minima forma di controllo sul movimento di alcuni muscoli (e.g., occhi, dita); qualora questa assunzione non sia vera, le interfacce cervello-computer costituirebbero l'unica soluzione per permettere a questi pazienti di esprimere la propria volontà.

1.1 La ricerca nel campo delle interfacce cervello-computer

Il campo di ricerca delle interfacce cervello-computer è un settore ancora giovane. La ricerca affonda le sue radici nelle indagini e negli studi neuroscientifici sul funzionamento del cervello.

Una delle ipotesi fondamentali su cui si basano molte ricerche neuroscientifiche è l'assunto che esista una relazione tra stati mentali e attività neurale [4]. In altre parole, sarebbe possibile definire un'esperienza mentale analizzando l'attività elettrica generata dai neuroni all'interno del cervello. In verità, questa ipotesi è tuttora molto dibattuta e la connessione tra stati mentali e attività neurale, ammesso che esista, è ben lungi dall'essere nota. Di fronte a questo problema, i ricercatori nel campo delle interfacce cervello-computer hanno seguito un approccio più pratico e ingegneristico. Consapevoli della difficoltà di dare una spiegazione teorica completa e coerente all'attività neurale, molti hanno preferito limitarsi a studiare in quale modo utilizzare e sfruttare i segnali prodotti dal cervello piuttosto che darne un'interpretazione [91]. In quest'ottica l'obiettivo principale della ricerca sulle interfacce cervello-computer diventa la capacità di elaborare (ed eventualmente indurre a generare) segnali cerebrali; conseguentemente, l'obiettivo di spiegare i fenomeni neurofisici viene spesso relegato in secondo piano [82].

Tutto questo significa che la ricerca sulle interfacce cervello-computer è principalmente volta all'analisi e alla risoluzione di problemi ingegneristici, non problemi neuro-scientifici [5]. La realizzazione, almeno a livello teorico, di un'interfaccia cervello-computer presenta indubbiamente delle difficoltà, ma dal momento che questi ostacoli sono soprattutto pratici e non concettuali, la fattibilità di interfacce cervello-computer con alte prestazioni sembra essere esclusivamente una questione di tempo.

1.2 Obiettivo della tesi

Come detto, il campo di ricerca delle interfacce cervello-computer è un settore giovane, in grado di offrire ai ricercatori specializzati in informatica numerose sfide di carattere ingegneristico. Sebbene esistano già diversi prototipi e modelli di interfacce cervello-computer, spesso i risultati e le performance di questi dispositivi sono ben lontani dai risultati ideali che si attenderebbero da un'interfaccia cervello-computer da utilizzare al di fuori di un laboratorio di ricerca.

Accuratezza, velocità, affidabilità e usabilità sono solo alcuni dei requisiti che un'interfaccia cervello-computer dovrebbe soddisfare; rispetto a ognuno di questi requisiti ancora molto lavoro può e deve essere svolto per colmare quel divario tra le attuali performance e il funzionamento ideale atteso.

Questa tesi in particolare si concentra sullo studio dell'accuratezza delle interfacce cervello-computer. L'accuratezza di un'interfaccia cervello-computer è determinata in maniera sostanziale dal modo in cui i segnali cerebrali acquisiti dall'utente sono trattati e manipolati; di conseguenza, nel corso dello sviluppo di questo progetto, sono stati analizzati e studiati quei processi e quei metodi che permettono di estrarre e ottenere informazione utile a partire da segnali grezzi; in particolare l'attenzione è stata focalizzata sugli algoritmi di calcolo delle feature, feature extraction, feature projection, feature selection e classificazione. Nel corso degli esperimenti sono stati sviluppati, testati e combinati tra loro diversi algoritmi; sono stati quindi confrontati e validati statisticamente i risultati ottenuti utilizzando algoritmi tradizionali presenti in letteratura e modifiche e combinazioni originali degli stessi. Più precisamente cinque differenti set di test sono stati condotti sui dati raccolti durante gli esperimenti:

- *Analisi delle performance delle sessioni di motor imagery senza feedback:* un primo insieme di test è stato eseguito utilizzando un'applicazione per un'interfaccia cervello-computer priva di feedback; durante questi esperimenti sono state valutate le prestazioni dell'interfaccia cervello-computer utilizzando sistemi di classificazione elementari; i dati così ottenuti sono stati utilizzati come metro di paragone nel corso dei successivi esperimenti;

- *Analisi delle performance delle sessioni di motor imagery con feedback:* un secondo insieme di test è stato condotto utilizzando un'applicazione per un'interfaccia cervello-computer con feedback; durante questi esperimenti sono state valutate le prestazioni dell'interfaccia cervello-computer utilizzando sistemi di classificazione elementari; i dati così ottenuti sono stati utilizzati per stimare il contributo del feedback sulle performance dell'interfaccia cervello-computer;

- *Analisi delle performance di diversi classificatori:* un terzo insieme di test è stato condotto implementando e testando diversi algoritmi per l'analisi e la classificazione del segnale; durante questi esperimenti sono stati utilizzati i dati raccolti durante i precedenti esperimenti; i risultati sono quindi stati raccolti e confrontati tra loro;

- *Validazione statistica delle performance dei diversi classificatori:* per valutare in maniera affidabile i risultati raccolti nel corso degli esperimenti precedenti è stato realizzato un insieme di test statistici volti a validare i risultati e le conclusioni raggiunte;

- *Impatto della generazione e selezione automatica delle feature:* con l'intento di automatizzare l'intero processo di generazione delle feature e di classificazione è stato inoltre sviluppato un algoritmo genetico originale per mezzo del quale un insieme ottimale di feature è selezionato in maniera

automatica anzichè in maniera manuale da un utente esperto; i risultati ottenuti sono stati poi confrontati con i risultati ottenuti nel corso dei precedenti esperimenti.

Da ultimo, i migliori algortimi di calcolo delle feature, feature extraction, feature projection, feature selection e classificazione sono stati selezionati per essere implementati nella nostra interfaccia cervello-computer realizzata all'interno del laboratorio di Intelligenza Artificiale e Robotica del Politecnico di Milano.

1.3 Struttura della tesi

La presente tesi è strutturata nel modo seguente:

Capitolo 1 - Introduzione Il presente capitolo offre al lettore una sintetica descrizione del settore di ricerca delle interfacce cervello-computer e spiega quale sia il ruolo, l'obiettivo e la motivazione di questa tesi.

Capitolo 2 - Cenni di biologia Il secondo capitolo presenta un insieme di nozioni elementari di biologia. Studiando un argomento che affonda le sue radici tanto nel campo dell'informatica quanto nel campo della biologia, sarebbe impossibile procedere nella ricerca senza cognizioni basilari di neurofisiologia. In questo capitolo si introduce dunque la struttura e il funzionamento del cervello, ponendo particolare attenzione a quegli elementi e a quelle funzioni che saranno poi utilizzate per lo sviluppo della nostra interfaccia-cervello computer.

Capitolo 3 - Registrazione dell'attività neurale Il terzo capitolo discute le moderne tecniche di registrazione dell'attività neurale. Dal momento che un'interfaccia cervello-computer funziona per mezzo del segnale generato nel cervello dall'utente, è di primaria importanza essere in grado di identificare e registrare questo segnale. In questo capitolo si descrivono dunque gli aspetti positivi e gli aspetti negativi delle moderne tecniche di registrazione dell'attività neurale, giustificando infine la decisione presa dal nostro laboratorio di sviluppare un'interfaccia cervello-computer basata su elettroencefalografo.

Capitolo 4 - Interfacce cervello-computer e stato dell'arte Il quarto capitolo si concentra sulle interfacce cervello-computer. Si cerca innanzitutto di dare una definizione di interfacce cervello-computer, evidenziandone le proprietà, i limiti, le applicazioni e le finalità. Si procede quindi delineando lo stato dell'arte delle interfacce cervello-computer; si descrivono perciò i più importanti modelli sviluppati, sottolineando le qualità e le limitazioni di ciascuno. L'ultima parte del capitolo è dedicata a un'analisi più approfondita di quella particolare famiglia di interfacce cervello-computer che è stata studiata nel nostro laboratorio, le interfacce cervello-computer sensomotorie; di queste ultime, oltre a citare i risultati ottenuti precedentemente in letteratura, si spiegano le basi biologiche del loro funzionamento

e si elenca un insieme di metodi comunemente utilizzati per valutarne le prestazioni.

Capitolo 5 - Analisi e classificazione del segnale per motor imagery Il quinto capitolo è dedicato alla teoria dell'analisi e della classificazione del segnale. Si descrivono le tecniche fondamentali di analisi e manipolazione del segnale (e.g., filtraggio, normalizzazione) e i principali metodi di generazione delle feature e di classificazione del segnale (e.g., feauture selection, feature projection) presenti in letteratura. Il problema di estrarre informazioni utili dal segnale acquisito da un utente costituisce il tema di interesse principale di questa tesi; in quanto tale, ampio spazio è stato riservato alla descrizione e alla spiegazione di quei metodi di generazione delle feature e di classificazione del segnale che sono stati poi implementati e testati con la nostra interfaccia cervello-computer.

Capitolo 6 -Sistema sviluppato e sua validazione Il sesto capitolo descrive il sistema per l'analisi e la classificazione del segnale sviluppato nel nostro laboratorio e gli esperimenti condotti per la validazione dello stesso. Sono presentati con precisione le configurazioni sperimentali adottate, i protocolli di laboratorio utilizzati durante gli esperimenti, l'ambiente software all'interno del quale sono state condotte le analisi dei dati raccolti durante gli esperimenti e, infine, i risultati ottenuti.

Capitolo 7 - Discussione Il settimo capitolo discute i risultati ottenuti nel corso degli esperimenti, ne suggerisce un'interpretazione e trae delle conclusioni. Infine sono indicate possibili proposte per uno sviluppo futuro di questo progetto.

Appendice A - Dimostrazioni Da ultimo, nell'appendice si raccolgono le spiegazioni e le dimostrazioni di alcuni teoremi utilizzati nel corso della tesi. L'appendice si configura come una sezione di consultazione a cui ricorrere per trovare l'enunciazione di alcuni dei teoremi citati, ma non spiegati esaustivamente, nei capitoli precedenti.

Capitolo 2

Cenni di biologia

Come affermato, il fine di una interfaccia cervello-computer è quello di costituire una via di comunicazione che dipenda esclusivamente dai segnali prodotti dall'attività neurale del cervello. E' opportuno dunque introdurre alcune nozioni elementari riguardo la natura e la struttura del cervello, delineando quegli aspetti che saranno di maggior interesse per le interfacce cervello-computer, e in particolare per le interfacce cervello-computer sensomotorie.

2.1 Cervello

Il cervello, o più correttamente l'*encefalo*[1], è il centro del sistema nervoso centrale umano, contenuto all'interno della scatola cranica. Esso è responsabile di tutte le attività del corpo umano, dalla regolazione del battito cardiaco alla gestione della memoria, dalla generazione di pensieri alla formazione di sogni e speranze [1]. La struttura del cervello ha una complessità seconda a nessuna degli altri organi del corpo umano e, tuttora, buona parte delle sue reali funzioni sono ancora oggetto di studio [1].

L'encefalo propriamente detto è costituito da (vedi Figura 2.1) [40, 71]:

Cervello (prosencefalo) il cervello costituisce la parte anteriore e superiore dell'encefalo; con la sua massa esso occupa quasi interamente la scatola cranica;

Tronco encefalico (mesencefalo) il tronco encefalico è situato nella parte inferiore dell'encefalo, tra il prosencefalo e il cervelletto; riceve input dai nervi cranici e trasferisce impulsi sensori e motori tra il cervello e la spina dorsale; contiene inoltre le strutture responsabili del controllo del battito cardiaco, della respirazione e della pressione sanguigna;

[1] Si noti che, tecnicamente, in campo medico, il termine inglese *brain*, utilizzato nella dicitura *brain-computer interface* (intefaccia cervello-computer), identifica l'*encefalo*; il corrispondente termine medico inglese per *cervello* è *cerebrum* [2].

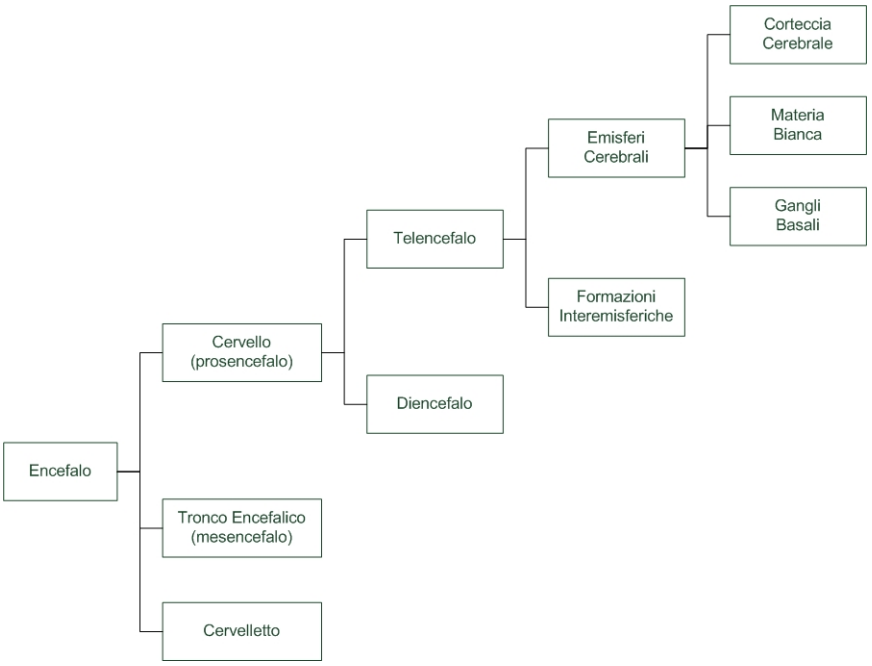

Figura 2.1: Struttura logica dell'encefalo

Cervelletto il cervelletto è situato nella parte posteriore e inferiore dell'encefalo; ha un ruolo fondamentale nell'apprendimento e nella memorizzazione dei movimenti, nonchè nella pianificazione e nel coordinamento.

Analizzando più a fondo il cervello, è possibile suddividerlo in [2, 40]:

Diencefalo collocato tra la parte superiore dell'encefalo e il mesencefalo, il diencefalo è costituito da strutture simmetriche specializzate, quali il *terzo ventricolo*, il *talamo*, il *subtalamo*, l'*epitalamo* e l'*ipotalamo*.

Telencefalo al di sopra del diencefalo trova posto il telencefalo che costituisce la parte esterna dell'encefalo; il telencefalo si compone di due *emisferi cerebrali*, bilaterali e simmetrici, separati tra loro dalla scissura interemisferica longitudinale, e di alcune *formazioni interemisferiche*.

I due emisferi cerebrali che formano il telencefalo constano di una superficie esterna di materia grigia, la *corteccia cerebrale* e di un nucleo interno di *materia bianca*; inoltre, in profondità, all'interno dei due emisferi sono presenti anche altre strutture di materia grigia, chiamate *gangli della base* [2, 40].

La corteccia cerebrale è filogeneticamente la parte più giovane dell'encefalo. La superficie esterna appare irregolare, caratterizzata da fessure (*solchi*) e rilievi (*giri*); i solchi più profondi sono detti *scissure* e permettono di suddividere la superficie della corteccia in regioni chiamate *lobi* (vedi Figura 2.2) [29].

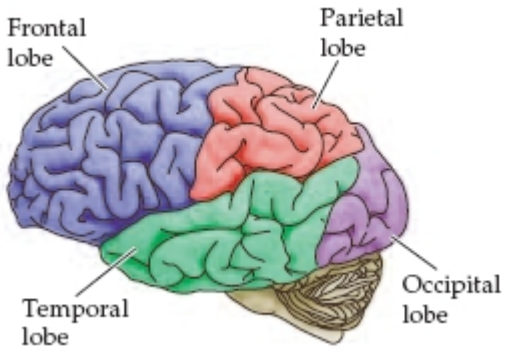

Figura 2.2: Suddivisione della corteccia in lobi [71]

La corteccia è preposta a un elevato numero di funzioni, tra cui funzioni sensorie, motorie, cognitive, affettive e mnemoniche. Più precisamente, separando l'area della corteccia cerebrale in lobi, è possibile identificare aree funzionali all'interno di ciascuno lobo (vedi Figura 2.3) [58]:

Lobo frontale l'area compresa tra la parte anteriore del cervello e il solco centrale contiene le aree funzionali preposte all'elaborazione delle informazioni motorie (*corteccia motoria*), visive, linguistiche e intellettive;

Lobo parietale l'area compresa tra il solco centrale e il solco pareto-occipitale contiene le aree funzionali preposte all'elaborazione delle informazioni sensorie (*corteccia somatosensoria*);

Lobo temporale l'area delimitata dal solco laterale contiene le aree funzionali preposte all'elaborazione delle informazioni uditorie (*corteccia uditiva*) e alla gestione della memoria;

Lobo occipitale l'area delimitata dal solco pareto occipitale contiene le aree funzionali preposte all'elaborazione delle informazioni visive (*corteccia visiva*).

2.2 Neuroni

La materia grigia che forma la corteccia cerebrale ha uno spessore variabile da 1.5 mm a 4.5 mm e viene generalmente suddivisa in sei *strati* secondo la densità e la disposizione delle cellule che la costituiscono [29]. Le cellule nervose si distinguono in [3, 29]:

Neuroni i neuroni sono le cellule nervose in grado di gestire l'informazione, elaborando e trasferendo segnali elettrici; si conta che nel cervello umano vi siano oltre 10^{10} neuroni [23].

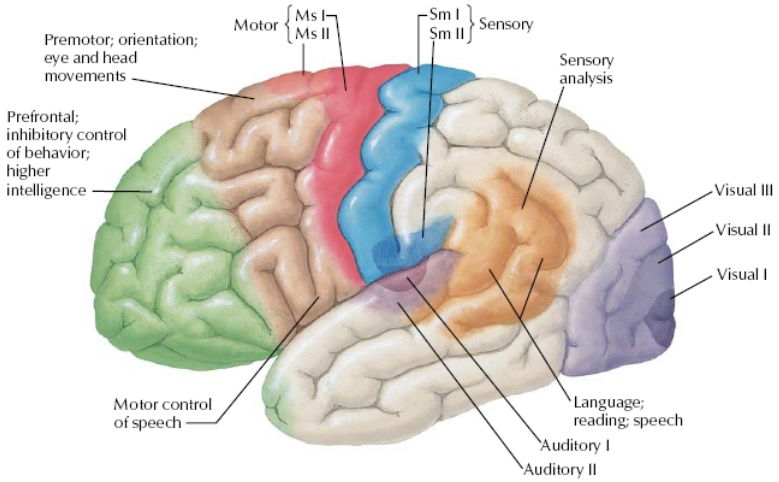

Figura 2.3: Aree funzionali della corteccia [58]

Cellule gliali le cellule gliali sono cellule nervose senza alcun ruolo nella comunicazione elettrica; sono più numerose dei neuroni e il loro unico compito è quello di supporto nutrizionale, ionico e meccanico ai neuroni .

Sebbene sia possibile distinguere ulteriormente i neuroni in diverse sottoclassi (e.g., cellule piramidali, cellule fusiformi, cellule orizzontali di Cajal, cellule stellate, cellule di Martinotti [29]), tutti i neuroni condividono una struttura simile. Ciascun neurone è costituito da un corpo cellulare (*soma* o *perikaryon*) da cui dipartono dei filamenti citoplasmatici (*dendriti* e *assone*). I dendriti si estendono dal soma del neurone e formano una struttura ad albero la cui funzione è quella di portare i segnali elettrici di altre cellule nervose al corpo cellulare del neurone. L'assone è, invece, un unico filamento citoplasmatico rivestito di mielina, da cui possono comunque generarsi rami secondari (*collaterali*), responsabili per la trasmissione dell'informazione dal soma del neurone ad altre cellule nervose; l'efficienza nella trasmissione del segnale elettrico lungo l'assone è determinata dal diametro dell'assone e dal livello di isolamento garantito dal rivestimento di mielina (vedi Figura 2.4) [29].

I punti di giunzione tra due neuroni, attraverso i quali è possibile trasferire informazione da un neurone all'altro, prendono il nome di *sinapsi*.

2.3 Sinapsi

Esistono due principali forme di comunicazione per mezzo delle quali i neuroni possono scambiarsi informazioni [66]:

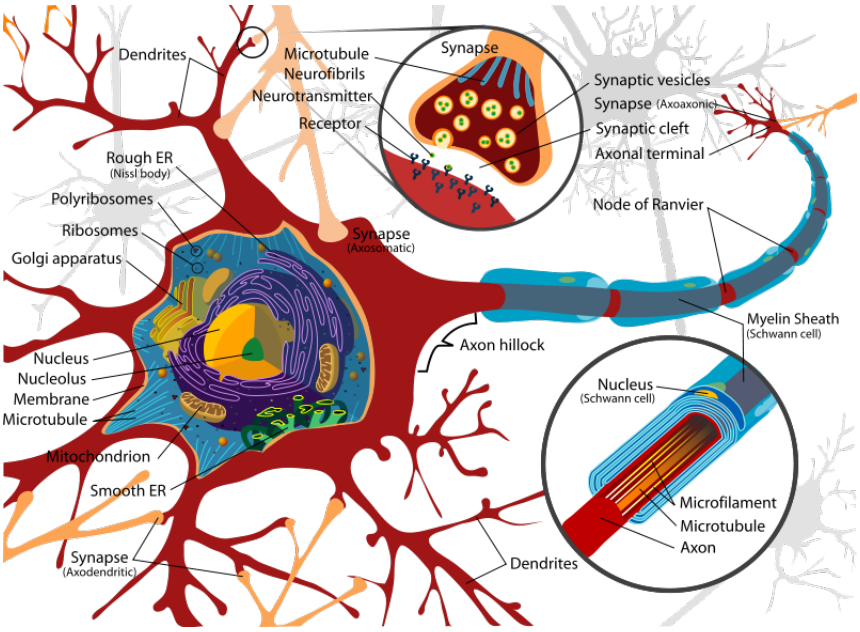

Figura 2.4: Struttrura di un neurone.

Comunicazione neurone-neurone si tratta di una forma di comunicazione punto-punto, resa possibile dalle sinapsi; figurativamente, è analoga a una comunicazione unicast in una rete cablata;

Comunicazione diffusa si tratta di una forma di comunicazione globale, non mediata dalle sinapsi, resa possibile dal rilascio di molecole messaggere in grado di diffondersi liberamente, come ad esempio il gas ossido nitrico; figurativamente, è analoga a una comunicazione broadcast in una rete wireless.

Considerando ora la comunicazione sinaptica tra neuroni, è possibile suddividerla a sua volta in due tipologie [66]:

Comunicazione sinaptica elettrica la comunicazione sinaptica elettrica ha luogo in presenza di *sinapsi elettriche*, ovvero quei punti in cui due cellule nervose entrano in contatto diretto; la comunicazione sinaptica elettrica è veloce e affidabile e permette il passagio diretto di impulsi elettrici in entrambe le direzioni.

Comunicazione sinaptica chimica la comunicazione sinaptica chimica si realizza in presenza di *sinapsi chimiche*, ovvero in quei punti in cui tra due cellule vi sia un piccolo spazio; gli impulsi elettrici sono veicolati tra le due cellule per mezzo di neurotrasmettitori chimici; la comunicazione sinaptica chimica è una forma di comunicazione unidirezionale, in cui la cellula

nervosa pre-sinaptica si specializza nella creazione e nel rilascio di neurotrasmettitori chimici in presenza di un impulso elettrico e la cellula nervosa post-sinaptica si specializza nel riconoscimento dei neurotrasmettitori e nella generazione di un impulso elettrico in presenza di neurotrasmettitori chimici.

Si conta che nel cervello umano vi siano più di 10^{13} sinapsi e oltre 100 diversi tipi di neurotrasmettitori [23].

L'informazione tra i neuroni è scambiata sotto forma di impulsi elettrici, chiamati *potenziali di azione*. Il flusso di corrente è reso possibile dal passaggio di ioni carichi positivamente o negativamente attraverso la membrana cellulare. Il potenziale di azione è una breve inversione del potenziale di membrana (i.e., la differenza di potenziale elettrico attraverso la membrana) di una cellula nervosa della durata di meno di 1 ms [42]; una diminuzione del potenziale di membrana viene detta depolarizzazione, mentre un aumento del potenziale è detto iperpolarizzazione.

Quando la concentrazione di anioni e cationi all'interno e all'esterno della membrana si equivalgono, allora il potenziale di membrana è nullo. Per modificare la differenza di potenziale tra l'interno e l'esterno del neurone intervengono i canali K^+ o Na^+ collocati sulla membrana cellulare; una volta aperti, questi canali consentono agli ioni K^+ o Na^+ di fluire attraverso la membrana cellulare e impediscono a ioni differenti o altre proteine di muoversi attraverso la membrana cellulare. In condizioni di riposo il potenziale di un neurone varia tra -50 e -75 mV [29].

Un potenziale d'azione è costituito da due fasi: una fase di depolarizzazione e una fase di repolarizzazione (vedi Figura 2.5). Durante la fase di depolarizzazione, i canali Na^+ iniziano ad aprirsi più rapidamente dei canali K^+; gli ioni Na^+ iniziano a entrare nella cellula e depolarizzano ulteriormente la membrana; questo fa sì che altri canali Na^+ vengano aperti. La soglia critica di questo fenomeno è il momento in cui il flusso di Na^+ entranti supera il flusso di K^+ uscenti; a quel punto si attiva un ciclo positivo che comporta un rapido aumento del potenziale d'azione e conseguentemente un picco della tensione; se il potenziale d'azione supera gli 0 mV, si parla di *overshoot* del potenziale d'azione. Durante la fase di repolarizzazione i canali Na^+ si disattivano, facendo ritornare il potenziale di azione al valore di riposo; analogamente alla fase di depolarizzazione, se il potenziale d'azione decresce al di sotto del potenziale di riposo, si parla di *undershoot* del potenziale di azione (vedi Figura 2.6) [29].

Il comportamento degli impulsi elettrici è quindi descritto da una legge *all-or-none*: se un impulso è sufficiente a depolarizzare un neurone oltre la soglia critica e a generare così un picco, il neurone si attiverà e trasmetterà a sua volta l'impulso elettrico; in caso contrario il neurone non si attiverà e nessun segnale verrà propagato [42].

Una volta generato, un potenziale di azione può percorrere, attraverso gli assoni, lunghe distanze all'interno del cervello senza alcuna perdita in ampiezza [3] raggiungendo velocità fino a 120 m/s [66]. Alla propagazione del potenziale di azione segue un breve periodo di refrattarietà totale o relativa durante il quale

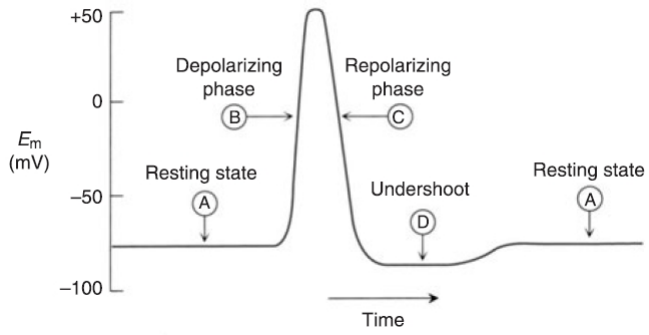

Figura 2.5: Potenziale d'azione [66]

i neuroni non hanno la possibilità di generare o trasmettere potenziali di azione [42].

2.4 Corteccia motoria

Di particolare interesse per lo sviluppo di interfacce cervello-computer sensomotorie è lo studio del funzionamento della corteccia motoria. Localizzata nel lobo frontale, la corteccia motoria presiede la pianifazione e l'esecuzione dei movimenti [53]. Sebbene il controllo dei movimenti si estenda a diverse regioni del cervello (tra cui i gangli basali, talamo e cervelletto [1]), l'area che maggiormente determina il movimento è la corteccia motoria.

La corteccia motoria è divisa in due aree [42]:

Corteccia motoria primaria (M1) l'attività della corteccia motoria primaria, sebbene non controlli in maniera assoluta i movimenti, dimostra una forte correlazione con i movimenti e le modalità dei movimenti (e.g., studiando le modalità di attivazione dei neuroni della corteccia motoria primaria è possibile determinare l'intensità del movimento);

Corteccia motoria secondaria (M2) divisa in *area motoria supplementare* (SMA) e *area premotoria* (PMA), la corteccia motoria secondaria è collocata anteriormente alla corteccia motoria primaria; è responsabile della pianificazione dei movimenti e la sua attività può precedere di centinaia di millisecondi la realizzazione del movimento; in particolare, l'area motoria supplementare è preposta a compiti complessi, mentre l'area premotoria pianifica quei movimenti che richiedono uno stimolo sensorio.

La corteccia motoria primaria è organizzata somatotopicamente come una mappa motoria controlaterale [29]: questo significa che la parte destra del corpo è mappata sull'emisfero sinistro, mentre la parte sinistra è mappata a destra. La rappresentazione delle diverse parti del corpo non è nè proporzionale nè ordinata: mani e testa ricoprono l'area maggiore della corteccia motoria; inoltre la

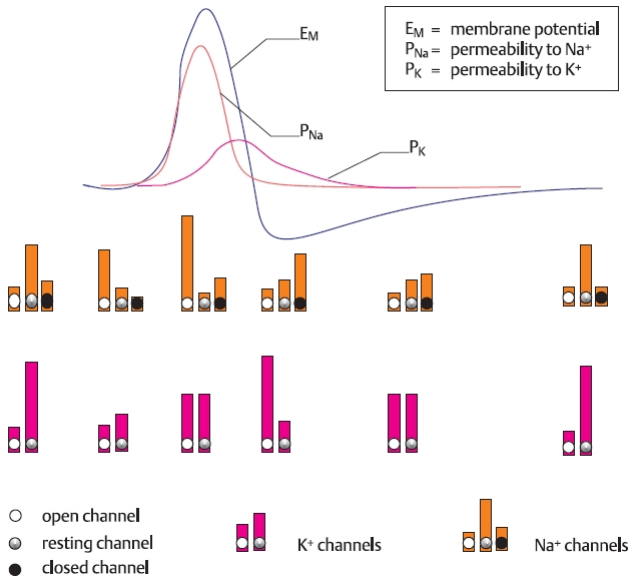

Figura 2.6: Potenziale d'azione e canali della membrana nuclerare [29]

rappresentazione di mani e piedi è collocata in prossimità della scissura interemisferica laterale, mentre la rappresentazione della testa si trova lateralmente rispetto alla scissura interemisferica laterale. La mappatura del corpo umano sulla corteccia motoria primaria disegna una figura umana deformata chiamata homunculus di Penfield (vedi Figura 2.7) [66].

All'interno della corteccia motoria il *tratto piramidale* (o *tratto corticospinale*) veicola gli impulsi che controllano l'esecuzione di precisi movimenti volontari. Il termine *piramidale* deriva dalle *cellule piramidali*, un particolare tipo di neuroni che devono il loro nome alla caratteristica forma del soma delle cellule nervose che costituiscono i tessuti di questa regione del cervello [29]. Come sarà spiegato nel prossimo capitolo, le cellule piramidali sono di fondamentale interesse per la realizzazione di interfacce cervello-computer sensomotorie non solo perchè correlate alla volontà di movimento di un soggetto, ma anche per la loro disposizione regolare che rende possibile una buona registrazione dei segnali generati [4].

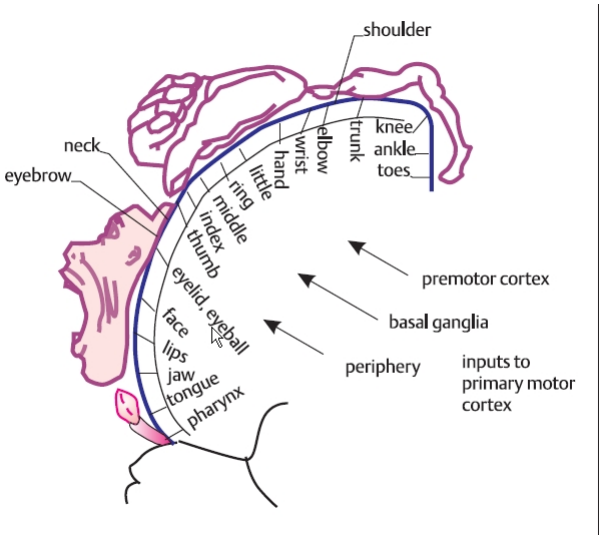

Figura 2.7: Homunculus di Penfield mappato sulla corteccia motoria primaria [29]

Capitolo 3

Registrazione dell'attività neurale

3.1 Tecniche di registrazione dell'attività neurale

Per esaminare e rilevare l'attività neurale del cervello esistono oggi svariati metodi basati su diverse tecnologie, ciascuno con i propri vantaggi e i relativi svantaggi [4, 88, 45, 66]:

Elettroencefalografia *(Electroencephalography, EEG)*: l'elettroencefalografia è una tecnica per misurare l'attività elettrica generata dall'attività neurale del cervello per mezzo di elettrodi posti sullo scalpo;

Elettrocorticografia *(Electrocorticography, ECoG)*: l'elettrocorticografia è una tecnica per misurare l'attività elettrica generata dall'attività neurale del cervello per mezzo di elettrodi posti chirurgicamente sulla corteccia cerebrale o per mezzo di griglie di microelettrodi inserite chirurgicamente nella corteccia cerebrale [72];

Magnetoencefalografia *(Magnetoencephalography, MEG)*: la magnetoencefalografia è una tecnica per misurare l'attività magnetica generata dall'attività neurale del cervello per mezzo di sensori cilindrici collocati in prossimità della testa [1];

Risonanza magnetica funzionale *(functional Magnetic Resonance Imaging, fMRI)*: la risonanza magnetica funzionale è una tecnica medica per la produzione di immagini tridimensionali; essa non registra direttamente l'attività elettrica dei neuroni, ma le conseguenze indirette della loro attività: generando un intenso campo magnetico esterno, è possibile valutare la risposta magnetica transitoria evocata all'interno del cervello e ottenerne così un'immagine tridimensionale [66];

Tipo di attività cerebrale:	Attività Elettrica	Attività Magnetica	Flusso Sanguigno	Riflessione della luce
Tecnica di acquisizione:	EEG, ECoG	MEG	fMRI, PET, SPECT	Optical Imaging

Tabella 3.1: Categorizzazione delle tecniche di registrazione dell'attività neurale.

Tomografia a emissione di positroni *(Positron Emission Tomography, PET)*: la tomografia a emissione di protoni è una tecnica medica per la produzione di immagini tridimensionali; essa collega le funzioni del cervello alle loro relative strutture: usando un isotopo radioattivo e osservando l'aumento di consumo di glucosio o il flusso del sangue, è possibile identificare quali aree del cervello sono attive in un dato intervallo di tempo [66];

Tomografia a emissione di singolo fotone *(Single Photon Emission Computed Tomography, SPECT)*: la tomografia a emissione di singolo fotone è una tecnica medica per la produzione di immagini tridimensionali; usando dei raggi gamma e un'apposita videocamera, è possibile rilevare e registrare l'attività del cervello.

Immagini ottiche *(Optical Imaging)*: il termine immagini ottiche si riferisce a una serie di tecniche mediche per la produzione di immagini tridimensionali, come la *diffusive optical imaging*, la *near infrared spectroscopy* o la *event related optical signal*; esse permettono di rilevare l'attività dei neuroni e di analizzare il funzionamento del cervello per mezzo della riflessione della luce emessa da una sorgente laser debole o infrarossa diretta verso il cervello [1];

A seconda dell'attività cerebrale che registrano, queste tecniche possono essere categorizzate come indicato in Tabella 3.1.

3.1.1 Confronto tra le tecniche di registrazione dell'attività neurale

Come detto, ciascuna delle tecniche illustrate nella sezione precendente presenta dei lati positivi e dei lati negativi.

La rilevazione dell'attività neurale per mezzo dell'elettroencefalografia (vedi Figura 3.1) è in grado di fornire con continuità un segnale con alta risoluzione temporale (dell'ordine di millisecondi) e si avvale di apparecchiature portatili e relativamente economiche; la preparazione della strumentazione e del soggetto sottoposto all'elettroencefalografia è rapida (dell'ordine di minuti) e assolutamente sicura; nel corso dell'acquisizione, inoltre, la strumentazione non vincola eccessivamente il soggetto, lasciandolo libero di muoversi, seppure in maniera limitata; infine, dal momento che l'elettroencefalografia è una tecnica di rilevazione dell'attività neurale molto utilizzata sia in ambito medico che nella ricerca, è disponibile una consistente documentazione e un ampio supporto per

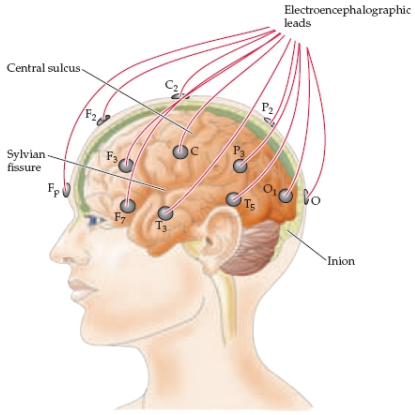

Figura 3.1: Montaggio degli elettrodi per elettroencefalografia [71].

l'uso. Tuttavia, l'elettroencefalografia ha inevitabilmente anche degli svantaggi: la registrazione, infatti, avviene per mezzo di elettrodi che registrano l'attività di numerosi neuroni e ricevono un segnale filtrato dai tessuti frapposti tra lo scalpo e il cervello; di conseguenza, la risoluzione spaziale di un'elettroencefalografia è estremamente limitata (dell'ordine di centimetri) [4].

L'elettrocorticografia, basata sullo stesso principio dell'elettroencefalografia, riesce a superare il principale limite della precedente tecnica spostando la posizione degli elettrodi dallo scalpo direttamente alla corteccia cerebrale, come mostrato in Figura 3.2. Essa riesce dunque a conciliare l'alta risoluzione temporale (dell'ordine di millisecondi) dell'elettroencefalografia con un'altrettanto alta risoluzione spaziale (dell'ordine di micrometri); il posizionamento degli elettrodi in prossimità della sorgente del segnale permette inoltre di registrare segnali con ampiezza maggiore (dell'ordine di 50-100 μv contro i 10-20 μv dell'elettroencefalografia), banda più ampia (0-200 Hz contro i 0-40 Hz dell'elettroencefalografia) e meno interferenze dovute a rumore esterno [72]. Escluso il primo posizionamento, l'elettrocorticografia conserva tempi di preparazione molto bassi, uguali o minori di quelli dell'elettroencefalografia. È però inevitabile che la decisione di impiantare gli elettrodi direttamente sulla corteccia cerebrale implichi degli svantaggi: in primo luogo, essendo necessaria un'operazione chirurgica per collocare gli elettrodi, questa tecnica risulta più invasiva e rischiosa dell'elettroencefalografia, in quanto presenta tutte le complessità connesse con un intervento chirurgico; in secondo luogo, la necessità dell'intervento di un équipe medica per l'impianto degli elettrodi non può che far lievitare il prezzo di questa tecnica; in terzo luogo, a parte gli inevitabili quesiti etici riguardo l'impianto di dispositivi artificiali in un essere umano, è necessario ottenere l'approvazione delle autorità mediche e legali competenti prima di procedere al collocamento degli elettrodi all'interno del cranio; da ultimo, ogniqualvolta si impiantino degli elettrodi per elettrocorticografia è necessario affrontare il problema della loro stabilità, che

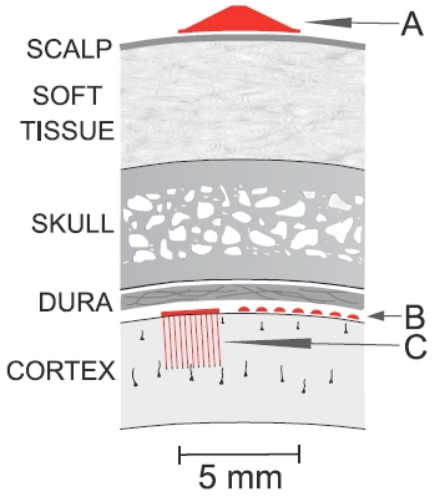

Figura 3.2: Posizionamento degli elettrodi nell'elettroencefalografia (A), nell'elettrocorticografia (B) o in altre tecniche invasive (C) [72].

pur essendo lunga (dell'ordine di anni), non è comunque illimitata [87] e richiede dunque un controllo periodico.

La magnetoencefalografia presenta caratteristiche molto simili all'elettroencefalografia; possiede cioè una risoluzione temporale e una risoluzione spaziale equivalenti a quelle dell'elettroencefalografia, nonchè un montaggio altrettanto sicuro. A differenza di quest'ultima, però, richiede un'apparecchiatura molto più sofisticata e meno portatile, poco adatta a essere utilizzata nella vita reale al di fuori di un laboratorio; inoltre, anche il costo risulta di gran lunga maggiore della semplice elettroencefalografia.

La risonanza magnetica funzionale, contrariamente all'elettroencefalografia o alla magnetoencefalografia, è in grado di fornire dati con una buona risoluzione spaziale (dell'ordine di millimetri), ma con una scarsa risoluzione temporale (dell'ordine di secondi); infatti, dovendo registrare le variazioni di consumo di ossigeno e di afflusso di sangue nel cervello, questa tecnica richiede inevitabilmente intervalli di tempo più lunghi per rilevare l'attività neurale. Trattandosi di una tecnica non invasiva che non richiede alcun intervento chirurgico, l'uso della risonanza magnetica funzionale è, sotto questo aspetto, sicuro. L'apparecchiatura per la risonanza magnetica funzionale, analogamente alla magnetoencefalografia, ha dimensioni consistenti, limitata portabilità e un prezzo maggiore della semplice elettroencefalografia [1].

La tomografia a emissione di positroni e la tomografia a emissione di singolo fotone sono tecniche estremamente utili in ambito medico e neuroscientifico per lo studio dell'attività cerebrale, ma di ridotto impiego per la realizzazione di interfacce cervello-computer a causa della loro limitata risoluzione sia a livello temporale (dell'ordine di minuti) sia a livello spaziale (dell'ordine di centimetri);

	Risoluzione Temporale	Risoluzione Spaziale	Costo Relativo	Dimensione	Portabilità	Sicurezza
EEG	$\sim$ms	$\sim$cm	\$	ridotta	portabile	alta
ECoG	$\sim$ms	$\sim\mu$m		ridotta	portabile	bassa
MEG	$\sim$ms	$\sim$cm	\$\$\$	larga	stazionaria	media
fMRI	$\sim$s	$\sim$mm	\$\$\$	larga	stazionaria	media
PET	$\sim$min	$\sim$cm	\$\$\$\$	larga	stazionaria	media
SPECT	$\sim$min	$\sim$cm		larga	stazionaria	media
Optical Imaging	$\sim$ms	$\sim$cm	\$	ridotta	portabile	alta

Tabella 3.3: Confronto delle tecniche di registrazione dell'attività neurale

in particolare, la bassa risoluzione temporale è dovuta, ancora una volta, come nel caso della risonanza magnetica funzionale, ai lunghi intervalli di tempo richiesti per monitorare il flusso di sangue e il consumo di energia all'interno del cervello. La tomografia a emissione di positroni e la tomografia a emissione di singolo fotone sono tecniche che non richiedono alcun intervento chirurgico, ma necessitano tuttavia di opportune precauzioni per l'introduzione nel corpo del paziente di isotopi traccianti radioattivi. Entrambe le tecniche utilizzano inoltre apparecchiature statiche, di grandi dimensioni e costose; in questa prospettiva, la principale differenza tra tomografia a emissione di positroni e tomografia a emissione di singolo fotone consiste nel prezzo, che pur essendo sempre elevato, risulta molto inferiore per la tomografia a emissione di singolo fotone grazie all'uso di isotopi radioattivi che richiedono un apparato tecnologico meno sofisticato [1].

I metodi di rilevazione dell'attività neurale basati su immagini ottiche, pur comprendendo tecniche differenti, sono in grado di raggiungere risoluzioni temporali (dell'ordine di millisecondi) e risoluzioni spaziali (dell'ordine di centimetri) analoghe all'elettroencefalografia grazie all'uso di laser; inoltre, poichè i laser utilizzati sono laser deboli, anche le tecniche basate su immagini ottiche sono assolutamente sicure. Le apparecchiatture per la realizzazione di immagini ottiche hanno generalmente dimensioni contenute che le rendono facilmente portatili; infine anche il loro costo non differisce molto dal costo richiesto per l'installazione e l'utilizzo di tecniche elettroencefalografiche.

Questo confronto tra le differenti tecniche di rilevazione dell'attività neurale si può sintetizzare in Tabella 3.3 [72, 37]. Dalla descrizione fatta e dalla

tabella sinottica presentata, è ora possibile motivare la decisione presa nel nostro laboratorio di adottare l'elettroencefalografia come tecnica per prelevare il segnale da usare nell'interfaccia cervello-computer; l'elettroencefalografia, infatti, si è dimostrata l'unica soluzione sensata e adottabile nel nostro laboratorio. L'uso dell'elettrocorticografia è stato scartato a priori in quanto la legislazione italiana pone dei vincoli riguardo l'uso di impianti artificiali nell'uomo per la ricerca e perchè, in ogni caso, il nostro laboratorio non possiede le strutture necessarie per eseguire operazioni chirurgiche. La magnetoencefalografia, la tomografia a emissione di positroni e la tomografia a emissione di singolo fotone sono state scartate in quanto, come evidente dall'analisi della tabella, queste alternative sono dominate dall'elettroencefalografia: l'elettroencefalografia è in grado di fornire risultati analoghi o migliori con apparecchiature più versatili ed economiche. La risonanza magnetica funzionale offre risultati migliori dell'elettroencefalografia in quanto a risoluzione spaziale, ma il costo proibitivo la rende poco attraente e, soprattutto, la bassa risoluzione temporale che richiede tempi molto lunghi per rilevare l'attività neurale fa sì che sia poco adatta per un'interfaccia cervello-computer che deve restituire un feedback in tempo reale. Le tecniche basate su immagini ottiche risultano le uniche vere concorrenti dell'elettroencefalografia: esse sono infatti in grado di fornire prestazioni comparabili a quelle offerte dall'elettroencefalografia; a questo punto, la disponibilità di una maggiore letteratura e la presenza di standard ormai approvati ha volto la nostra scelta sull'uso dell'elettroencefalografia in luogo delle tecniche basate su immagini ottiche. Da ultimo, a ulteriore supporto della nostra decisione, vi è la considerazione che l'economicità e la portatilità di un elettroencefalografo sono un vantaggio non solo per il nostro laboratorio, ma anche e soprattutto per i potenziali futuri fruitori dell'interfaccia cervello-computer; infatti, il costo ridotto e le dimensioni contenute permetterebbero a ogni potenziale utente di acquistare il proprio elettroenfalografo e utilizzare l'interfaccia cervello-computer presso la propria residenza.

3.2 Elettroencefalografia

Dal momento che si è deciso di adottare la tecnica dell'elettroencefalografia per registrare l'attività neurale, si procede ora a una descrizione più accurata di questa tecnica.

3.2.1 Strumentazione

Sviluppata negli anni '20 del 1900 da Hans Berger, l'elettroencefalografia è stata la prima tecnica di registrazione dell'attività cerebrale [87, 72, 45].

L'elettroencefalografia è in grado di registrare l'attività elettrica del cervello per mezzo di una serie di elettrodi. A seconda della loro dimensione si possono distinguere due tipologie di elettrodi [4]:

Macroelettrodi i macroelettrodi sono elettrodi a forma di disco o di cono, in metallo (oro, stagno, argento o cloruro di argento), del diamentro di 5-

10 mm; questi macroelettrodi possono essere collocati sia sullo scalpo nel corso di un'elettroencefalografia, sia impiantati sulla corteccia cerebrale per eseguire un'elettrocorticografia.

Microelettrodi i microelettrodi sono elettrodi di dimensioni minime, nell'ordine di micron, estremamente affidabili, realizzati in vetro con filamenti in oro o in tungsteno; questi microelettrodi sono progettati per registrare l'attività di singoli neuroni o piccoli gruppi di neuroni nel corso di un elettrocorticografia; esempi di microelettrodi sono lo Utah Intracranical Electrode Array o il Cone Electrode.

Essendo interessati solo all'elettroencefalografia, nel corso dei nostri esperimenti in laboratorio sono stati impiegati esclusivamente macroelettrodi; per questa ragione, d'ora in avanti, si userà indifferentemente il termine macroelettrodi ed elettrodi.

3.2.2 Montaggio

Per il montaggio degli elettrodi si possono usare indifferentemente singoli elettrodi oppure una cuffia di elettrodi che tipicamente integra 16, 32, 64, 128 o 256 elettrodi; il ricorso alla cuffia è senz'altro più comodo e più rapido nel caso si debba utilizzare un elevato numero di elettrodi.

Prima del collocamento degli elettrodi o prima della registrazione da cuffia, è necessario *preparare* il soggetto per garantire una migliore rilevazione del segnale da parte dell'elettroencefalografo. La *preparazione* consiste di due fasi distinte e successive [4]:

1. La prima fase consiste nell'abrasione della cute dello scalpo per mezzo di un opportuno gel abrasivo nei punti in cui saranno collocati gli elettrodi; lo scopo di questo processo, assolutamente indolore, è quello di rimuovere lo strato di cellule dermatiche morte che hanno una conduttività elettrica molto bassa e che disturbano il segnale a causa dell'attività elettrodermatica nota come Galvanic Skin Responde (GSR);

2. La seconda fase consiste nello spalmare sulla cute dello scalpo un opportuno gel conduttivo nei punti in cui saranno collocati gli elettrodi; lo scopo di questo processo è quello di aumentare la conduttività elettrica e diminuire la resistenza registrata sugli elettrodi sotto una soglia accettabile (generalmente 5-10 kΩ).

Una volta completate queste due fasi è possibile procedere al posizionamento vero e proprio degli elettrodi. Come detto in precedenza, l'elettroencefalografia è in grado di registrare una differenza di potenziale all'interno del cervello e, per questa ragione richiede l'utilizzo di almeno due elettrodi, di cui uno di riferimento; in base a questa considerazione sono possibili due differenti tipi di montaggio [4]:

Montaggio bipolare nel montaggio bipolare gli elettrodi sono raggruppati a coppie: ogni elettrodo possiede un corrispettivo rispetto al quale sono

valutate le differenze di potenziale; questo significa che per ottenere n segnali, sono necessari $2n$ elettrodi;

Montaggio monopolare nel montaggio monopolare ogni elettrodo è valutato rispetto a un riferimento comune; questo riferimento è generalmente collocato su un sito considerato neutro, ovvero non affetto dall'attività cerebrale (e.g., mastoide o lobo dell'orecchio); per rilevazioni più affidabili è anche possibile usare più di un riferimento e valutare il segnale rispetto alla media dei riferimenti (e.g., due riferimenti collocati simmetricamente sui mastoidi o sui lobi dell'orecchio); questo significa che per ottenere n segnali sono necessari $n+k$ elettrodi, dove k è il numero di riferimenti che si intendono usare.

La posizione sullo scalpo dei singoli elettrodi è regolata da uno standard internazionale definito dall'American EEG Society e chiamato Sistema 10-20 per il Posizionamento degli Elettrodi (10-20 System of Electrode Placement) [60]. Questo standard permette di collocare gli elettrodi in posizioni ben determinate e consente dunque di definire con precisione quali posizioni siano state usate nel corso di un esperimento e di ripetere lo stesso esperimento usando le stesse locazioni con una certa facilità [59, 92]. Il nome 10-20 deriva dalla distanza percentuale che intercorre tra gli elettrodi o tra i punti di misurazione per il collocamento degli elettrodi: prima di posizionare gli elettrodi, si traccia il segmento immaginario che connette il naso alla nuca (o, più precisamente, il nasion all'inion) e il segmento che unisce il lobo auricolare destro al lobo auricolare sinistro; si divide ciascuno in sei sotto-segmenti tali che il primo e l'ultimo misurino ognuno il 10% del segmento originale e i restanti quattro misurino ciascuno il 20% del segmento originale; i punti identificati dall'inizio e dalla fine di ciascuno dei sotto-segmenti calcolati costituiscono i riferimenti in base ai quali determinare la posizione degli elettrodi come in Figura 3.3 [15]. Anche la nomenclatura della posizione degli elettrodi è soggetta a uno standard: ciasuna locazione è identificata da una lettera e un numero; la lettera indica sopra quale regione della corteccia cerebrale l'elettrodo è collocato (F indica la zona frontale, T la zona temporale, C la zona centrale, P la zona parietale e O la zona occipitale); il numero identifica un determinato elettrodo tra quelli collocati al di sopra della stessa regione della corteccia cerebrale, con la convenzione che un numero pari identifica un elettrodo collocato al di sopra dell'emisfero destro e un numero dispari identifica un elettrodo al di sopra dell'emisfero sinistro; fanno eccezione gli elettrodi posizionati al di sopra della linea che divide l'emisfero destro dall'emisfero sinistro: il secondo elemento del loro identificativo non può infatti essere un numero, e per questo viene utilizzata una "z" minuscola [59, 60].

3.2.3 Funzionamento

Gli elettrodi posti sullo scalpo di un soggetto sono in grado di registrare differenze di potenziale sullo scalpo, ovvero di evidenziare una differenza elettrica tra una zona attiva del cervello e una zona neutra di riferimento.

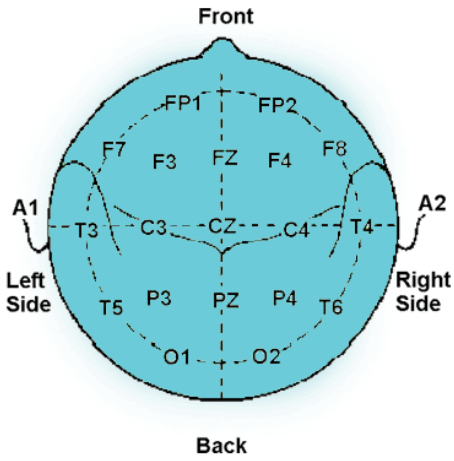

Figura 3.3: Il Sistema Internazionale 10-20 per il Posizionamento degli Elettrodi

Affinchè un segnale significativo possa essere registrato dall'elettroencefalografo è necessario che alcune condizioni sulla disposizione e sull'attivazione dei neuroni siano soddisfatte [4]:

1. I neuroni devono generare un segnale elettrico lungo uno specifico asse perpendicolare allo scalpo; infatti, segnali generati lungo un asse parallelo allo scalpo o con componenti simmetriche che si elidono (e.g., neuroni radialmente simmetrici che generano un segnale, anche significativo, lungo tutti i dendriti) non sono rilevati dall'elettroencefalografo;

2. I segnali generati dai neuroni devono essere paralleli ed equiversi; infatti, segnali controversi (e.g., segnali generati da un gruppo di neuroni disposti casualmente che generano un segnale orientato casualmente) tendono a elidersi e a non essere rilevati dall'elettroencefalografo;

3. I segnali generati dai neuroni devono attivarsi contemporaneamente o in un breve intervallo di tempo; infatti se gruppi di neuroni si attivassero a non breve distanza di tempo gli uni dagli altri, l'elettroencefalografo registrerebbe un potenziale pressochè costante, senza evidenziare nessuna variazione degna di rilievo;

4. I segnali generati dai neuroni devono avere segno concorde; infatti segnali equiversi, ma con segno opposto si elidono e non sono registrati dall'elettroencefalografo.

Fortunatamente, la configurazione dei neuroni nel tessuto della corteccia rispetta questi vincoli; le cellule piramidali, ad esempio, sono allineate e disposte perpendicolarmente allo scalpo; si ritiene invero che buona parte del segnale registrato dagli elettrodi sia proprio determinato dall'attività delle cellule piramidali [60].

Si ricordi inoltre che, data la bassa risoluzione spaziale dell'elettroencefalografia, è molto difficile relazionare il segnale registrato da un elettrodo con la precisa zona del cervello in cui esso si è originato e con la sua esatta funzione: un segnale può essere generato da popolazioni di neuroni estremamente disperse nel cervello, diversi gruppi di neuroni possono essere attivati per finalità differenti (multitasking), intere popolazioni di neuroni possono risultare invisibili all'elettroencefalografo [4]; a complicare ulteriormente lo studio, vi è poi anche il fatto che l'attivazione di gruppi di neuroni può variare, anche se entro certi limiti, da soggetto a soggetto.

Questa imprecisione o approssimazione del segnale registrato dall'elettroencefalografo è stata spesso indicata come il limite maggiore, e per alcuni insormontabile, delle interfacce cervello-computer basate sull'elettroencefalografia. L'uso dell'elettroencefalografo per rilevare l'attività generata da milioni di neuroni, più o meno sincrona e filtrata dai tessuti cranici, è stata paragonata al tentativo di ascoltare una folla radunata in uno stadio dal parcheggio prospicente lo stadio (in contrasto con l'elettrocorticografia, che sarebbe invece analoga ad ascoltare il discorso di un paio di tifosi nello stadio da alcuni posti di distanza) [85].

Il neuroscienziato J. Chapin, impegnato nella ricerca su interfacce cervello-computer invasive alla State University of New York Health Sciences Center, ha giustamente evidenziato che il limite delle interfacce cervello-computer basate sull'elettroencefalografia è che esse "do not extract the actual information in our brains - for example, our concept of a word"[1]. D'altra parte, pur riconoscendo questa innegabile limitazione, J.R. Wolpaw ha sottolineato come questo limite sia ininfluente per le interfacce cervello-computer attualmente sviluppate, in quanto "BCIs just need to convey intent" [85] [2]. In altre parole, per le interfacce cervello-computer non è necessaria una conoscenza precisa di come o cosa comunichino tra loro i neuroni, ma è sufficiente sapere che comunicano quando si intende raggiungere un determinato scopo. Per questa ragione, nonostante la ridotta risoluzione spaziale, l'elettroencefalografia è tuttora una valida tecnica per l'implementazione di un'interfaccia cervello-computer.

3.2.4 Ritmi elettroencefalografici

L'analisi e lo studio dei tracciati delle registrazioni elettroencefalografiche in soggetti adulti ha permesso di evidenziare la presenza di diverse oscillazioni regolari, che riflettono l'attività ritmica di gruppi di neuroni sincronizzati [4]. A seconda della frequenza dell'oscillazione e della locazione sullo scalpo su cui sono registrati, questi ritmi sono stati categorizzati in diverse classi [3]:

Ritmo Alfa il ritmo alfa è centrato intorno a 10 Hz, con estremi tra 8 e 13 Hz (vedi Figura 3.4); ha un'ampiezza compresa tra 15 e 65 μV ed è registrato principalmente nella parte occipitale e parietale della corteccia. Il ritmo

[1] "Non sono in grado di estarre la reale informazione contenuta nel nostro cervello - ad esempio, il nostro concetto di parola.". (trad. propria)

[2] "Le interfacce cervello-computer devono soltanto trasmettere un'intenzione.". (trad. propria)

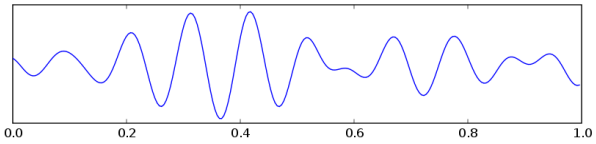

Figura 3.4: Ritmo Alfa

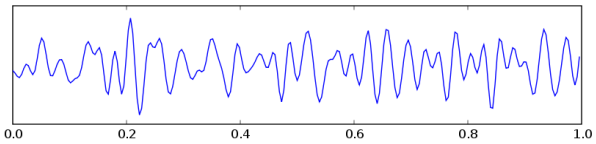

Figura 3.5: Ritmo Beta

alfa sembra associato al rilassamento e a una ridotta consapevolezza [60]: infatti risulta parzialmente o completamente assente nel momento in cui si intraprende un'attività mentale, mentre è accentuato nel momento in cui ci si rilassa o si chiudono gli occhi;

Ritmo Beta il ritmo beta occupa generalmente le frequenze comprese tra 18 e 25 Hz (vedi Figura 3.5); in rari casi può abbassarsi a frequenze tra 14 e 17 Hz e in casi ancora più rari occupare frequenze superiori a 25 Hz. Ha ampiezze ridotte, generalmente inferiori ai 25 μV ed è registrato principalmente nella parte frontale della corteccia. Il ritmo beta sembra associato all'attività mentale e all'attenzione [71]: risulta, infatti, sempre meno accentuato quanto più il tempo trascorre e il paziente si stanca;

Ritmo Mu il ritmo mu occupa le frequenze tra 8 e 10 Hz ed è registrato principalmente al di sopra della corteccia motoria (vedi Figura 3.6). Il ritmo mu risulta più evidente nei soggetti giovani e sembra associato al movimento. Tale ritmo ha un comportamento reattivo rispetto alla programmazione del movimento: il rilassamento accentua il ritmo mu, mentre la preparazione o l'immaginazione di un movimento bloccano il ritmo mu;

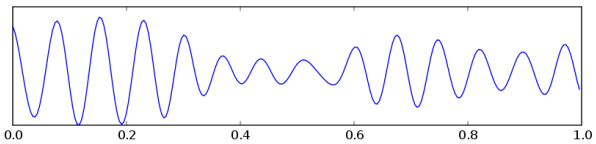

Figura 3.6: Ritmo Mu

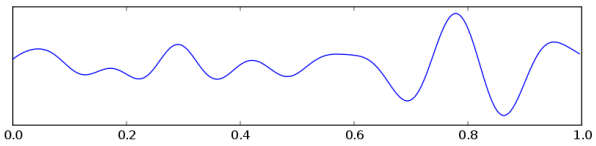

Figura 3.7: Ritmo Theta

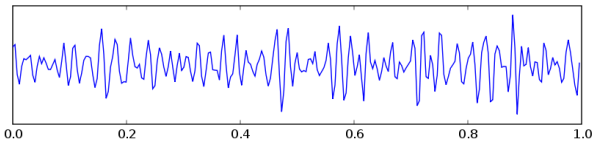

Figura 3.8: Ritmo Gamma

Ritmo Theta il ritmo theta occupa le frequenze tra 4 e 7 Hz ed è registrato principalmente nella parte frontale e centrale della corteccia (vedi Figura 3.7). Il ritmo theta sembra essere associato a stati di ispirazione creativa o di meditazione [60]; inoltre risulta accentuato durante il sonno o durante l'esecuzione di alcune attività mentali particolarmente impegnative;

Ritmo Lambda il ritmo lambda occupa le frequenze tra 4 e 6 Hz; è registrato principalmente nella parte occipitale della corteccia; ha ampiezze comprese tra i 20 e i 50 μV. Il ritmo lambda sembra associato a uno stato di attenzione e studio dell'ambiente circostante: risulta infatti evidente quando il paziente ha modo di osservarsi intorno, mentre risulta bloccato nel momento in cui si chiudono gli occhi;

Ritmo Delta il ritmo delta occupa le frequenze inferiori a 4 Hz; è registrato principalmente nella parte frontale e centrale della corteccia. Il ritmo delta sembra associato, nei soggetti sani, esclusivamente a uno stato di sonno [4].

Ritmo Gamma il ritmo gamma occupa le frequenze superiori a 35 Hz (vedi Figura 3.8). Il ritmo gamma sembra associato alla consapevolezza, alla capacità di unire differenti funzioni modulari cerebrali per formare un'immagine coerente [60].

Gli studi neurofisiologici sulla classificazione dei differenti ritmi elettroencefalografici e sull'associazione di ciascuno di questi con particolari stati o condizioni ha profonde implicazioni nella ricerca sulle interfacce cervello-computer; infatti, grazie ai risultati conseguiti in questo ambito, nello sviluppo di una particolare interfaccia cervello-computer, sarà possibile concentrare la propria attenzione solo su un limitato gruppo di ritmi, ovvero all'analisi di uno specifico insieme di segnali in un dato intervallo di frequenze registrate su una specifica regione dello scalpo; ciò consente sia di ridurre la mole di dati da dover analizzare, sia di eliminare segnali privi di informazione utile per l'interfaccia cervello-computer.

Capitolo 4

Interfacce cervello-computer e stato dell'arte

4.1 Definizione di interfacce cervello-computer

Esistono varie definizioni di quella che dovrebbe essere una interfaccia cervello-computer [28]. Una delle prime è offerta da J.J. Vidal in [82]:

> "The Brain Computer Interface project [...] was meant to [...] utilize the brain signals in a man-computer dialogue [...]."[1].

È interessante notare che, nonostante l'idea di un dialogo cervello-computer fosse nel 1973 tutt'altro che semplice, l'obiettivo ultimo che J.J. Vidal prospetta per la ricerca sulle interfacce cervello-computer è ancora più ambizioso e visionario:

> "To provide a direct link between the inductive mental processes used in solving problems and the symbol-manipulating, deductive capabilities of the computer, is, in a sense, the ultimate goal in man-machine communication. It would indeed elevate the computer to a genuine prosthetic extension of the brain."[2].

Una definizione più recente è fornita da J.R. Wolpaw e dai suoi colleghi del Wadsworth Center in [88] o [80], secondo cui un'interfaccia cervello-computer è un dispositivo creato:

[1] "L'obiettivo di una interfaccia cervello-computer è di utilizzare i segnali cerebrali in un dialogo uomo-computer" (trad. propria).

[2] "Fornire un collegamento diretto tra i processi mentali induttivi usati dall'uomo nella risoluzione di problemi e la manipolazione di simboli e le capacità deduttive di un computer è, in un certo senso, il fine ultimo della comunicazione uomo-macchina. Significherebbe, invero, elevare i computer a una pura protesi del cervello" (trad. propria).

> "to provide the brain with a new, non-muscular communication and control channel, [...] for conveying messages and commands to the external world."[3].

A differenza della definizione più generale di J.J. Vidal, J.R. Wolpaw definisce le interfacce cervello-computer in relazione al loro utilizzo in ambito medico; il fine delle interfacce cervello-computer [90] è infatti ridefinito come:

> "provide communication and control to people who are totally paralized."[4].

Analoga a quest'ultima è la definizione presentata in [34] da J.P. Donoghue e da altri ricercatori impegnati nello studio di protesi neuromotorie (neuromotor prostheses, NMPs), una tipologia di interfacce cervello-computer. Essi affermano che una protesi neuromotoria è un dispositivo che:

> "aims to replace or restore lost motor functions in paralysed humans by routeing movement-related signals from the brain, around damaged parts of the nervous system, to external effectors."[5].

Ancora una volta, dunque, l'attenzione è posta sul possibile utilizzo di tali interfacce cervello-computer in ambito medico.

Infine, un'ultima definizione che si considera è quella adottata nel corso del First International Meeting per la ricerca sulle interfacce cervello-computer tenutasi nel 2000, di cui il Wadsworth Center era un organizzatore [87]; in questo convegno si è ribadito che

> "A brain-computer interface is a communication system that does not depend on the brain's normal output pathways of peripheral nerves and muscles."[6].

È evidente che le definizioni presentate in questa sede sono tra loro sostanzialmente omogenee e differiscono per lo più nelle finalità; ciò che le accomuna è la definizione di un'interfaccia cervello-computer come un dispositivo che consente una *comunicazione uomo-macchina basata solo sui segnali generati dall'attività neurale del cervello e indipendente da qualsiasi altro segnale.*

A questo proposito, è stato provato che l'obiettivo di una comunicazione uomo-macchina che dipenda solo dall'attività del cervello e sia indipendente da qualsiasi altro segnale, in particolare segnali prodotti in preparazione o in seguito al movimento di un muscolo, non è un'utopia, ma è praticamente realizzabile; di fronte al dubbio che i risultati di un'interfaccia cervello-computer potessero

[3] "Per fornire al cervello un nuovo canale di comunicazione e controllo non-muscolare, per portare messaggi e comandi al mondo esterno" (trad. propria).

[4] "Fornire comunicazione e controllo a persone che sono totalmente paralizzate" (trad. propria).

[5] "Mira a rimpiazzare o a riparare le funzioni motorie perdute da umani paralizzati inviando segnali relativi al movimento dal cervello, evitando le parti danneggiate del sistema nervoso, fino a effettori esterni" (trad. propria).

[6] "Un'interfaccia cervello-computer è un sistema di comunicazione che non dipende dalle normali vie di output del cervello, costituite da nervi o muscoli periferici" (trad. propria).

essere determinati da contrazioni muscolari, magari impercettibili e inconsce, si è dimostrato [81] che, misurando l'attività dei muscoli cranici e dei muscoli degli arti, l'output di un'interfaccia cervello-computer è determinato dall'attività cerebrale ed è indipendente, o minimamente influenzato, dall'attività muscolare. Questo risultato, ottenuto dallo studio dell'utilizzo di un'interfaccia cervello-computer da parte di utenti sani, è stato poi ulteriormente convalidato dall'utilizzo di interfacce cervello-computer da parte di utenti completamente paralizzati.

4.2 Le interfacce cervello-computer come skill

La definizione data nel paragrafo precedente offre solo una visione parziale di ciò che in realtà è un'interfaccia cervello-computer; tale definizione, infatti, spiega quale sia il ruolo e la finalità del computer all'interno di un'interfaccia cervello-computer, ma trascura il fatto che un'interfaccia cervello-computer è costituita dall'unione di una macchina e di un soggetto umano. Prima di poter utilizzare efficientemente un'interfaccia cervello-computer è spesso necessaria una fase di addestramento; in molti casi, soprattutto nel caso delle interfacce cervello-computer sensomotorie non-invasive oggetto di questa tesi, è necessario un periodo di addestramento durante il quale l'utente impari a modulare i propri segnali elettrofisiologici per poter controllare con successo un'interfaccia cervello-computer.

In altre parole, l'analogia, spesso addottata in maniera inappropriata, di un'interfaccia cervello-computer come di una macchina in grado di leggere ("mind-reading") o ascoltare ("wire-tapping") [88] l'attività del cervello umano e trasformare la volontà dell'utente in azione è poco verosimile; è più realistico concepire un'interfaccia cervello-computer come un sistema basato sull'interazione di due dispositivi adattativi: il cervello umano e il computer; grazie alla versatilità del software e alla plasticità del cervello umano è possibile far sì che segnali appositamente generati dal cervello dell'utente siano riconosciuti e convertiti in un output sensato da parte di una macchina. Dunque l'utilizzo di un'interfaccia cervello-computer richiede che un utente sviluppi una nuova abilità ("skill") [87], un'abilità che non consiste nel controllo di uno o più muscoli, ma nel controllo della propria attività neurale. Ciò è possibile grazie all'estrema plasticità del cervello umano, capace di dedicare parte delle sue risorse a nuove abilità; P. Kennedy, scienziato impegnato nella ricerca sulle interfacce cervello-computer presso la Neural Signals Inc., parlando dei suoi pazienti che hanno appreso a muovere un cursore per mezzo di un'interfaccia cervello-computer, riconosce che essi sono stati in grado di imparare una nuova abilità evolvendo parte del loro cervello in quella che egli ha definito "cursor-related cortex", ovvero, corteccia cerebrale dedicata al controllo del cursore [5].

4.3 Interfacce cervello-computer e applicazioni di interfacce cervello-computer

Parlando di interfacce cervello-computer, è inoltre opportuno precisare la differenza che intercorre tra interfacce cervello-computer e applicazioni di interfacce cervello-computer [87]. Con il primo termine, interfacce cervello-computer, si intendono gli strumenti che permettono l'acquisizione di un segnale (i.e., l'insieme di elettroencefalografo, scheda di acquisizione digitale e software di elaborazione del segnale); un'interfaccia cervello-computer è dunque definita secondo la sua funzione e può essere giudicata in base alla sua semplicità d'uso ed efficienza. Con il secondo termine, applicazioni di un'interfaccia cervello-computer, si fa riferimento a un sistema che utilizza uno strumento, ovvero un'interfaccia cervello-computer, per un fine pratico (e.g., la selezione di una lettera su uno schermo); a un'interfaccia cervello-computer possono dunque corrispondere più applicazioni; un'applicazione di un'interfaccia cervello-computer è quindi definita dal suo scopo e può essere valutata considerando quanto bene riesce a raggiungere il suo obiettivo.

4.4 Principali destinatari delle interfacce cervello-computer

Come detto, la finalità delle interfacce cervello-computer è quella di fornire una via di comunicazione che sia indipendente dai normali canali di output del cervello, ovvero dai muscoli. Questa nuova possibilità di comunicazione e interazione con il mondo esterno è sviluppata prima di tutto a favore di quei soggetti vittime di gravi ed estese forme di paralisi, tra le quali la sindrome *locked-in*; i pazienti affetti da questa sindrome, pur mantenendo inalterate le proprie facoltà cognitive, perdono però il controllo dei propri muscoli; nei casi più gravi, quando il paziente non è più in grado di controllare nessun semplice movimento (e.g., il movimento degli occhi), un'interfaccia cervello-computer può rappresentare l'unica via di comunicazione possibile con il mondo esterno [91, 89, 47, 54, 81, 48, 94, 5, 67, 87, 92, 60, 88, 4, 68, 85, 50, 51, 93, 75, 90, 25, 49, 52, 37, 34, 38, 80, 73, 45, 15, 72][7]. Tra i principali beneficiari delle interfacce cervello-computer vi sono quei pazienti che sono stati colpiti da [88]:

Sclerosi Amiotrofica Laterale *(Amyotrophic lateral sclerosis, ALS)*: la sclerosi amiotrofica laterale, o morbo di Lou Gehrig, è la più comune forma di disordine dei neuroni motori; si tratta di una malattia degenerativa che colpisce i neuroni motori del cervello e della spina dorsale compromettendo il controllo volontario dei muscoli; si manifesta inizialmente con la paralisi di mani o piedi per poi estendersi progressivamente fino a giungere alla completa paralisi di ogni muscolo, compreso l'apparato respiratorio [1, 3];

[7] Si citano così tante fonti non perchè ognuna di esse sia rilevante e originale rispetto alle altre, ma esclusivamente per mostrare come questa finalità sia estremamente diffusa e sentita tra gli studiosi nel campo della ricerca sulle interfacce cervello-computer.

Ictus del tronco encefalico *(Brainstem stroke)*: un ictus si verifica quando un'arteria che trasporta ossigeno ed elementi nutritivi al cervello è otturata da un coagulo di sangue oppure esplode; in seguito a questo evento i neuroni nella parte del cervello privata di sangue muoiono nel giro di pochi minuti. Una tipologia di ictus particolarmente grave è quella che colpisce il tronco encefalico; nel caso il paziente sopravviva all'ictus del tronco encefalico, è possibile che sia colpito da paralisi di una metà del corpo o addirittura dell'intero corpo [1, 16];

Danni al cervello o alla colonna vertebrale *(Brain or spinal injury)*: "danni al cervello o alla colonna vertebrale" è un termine generico per indicare danni al sistema nervoso in seguito a incidente, ad abuso di alcool o droghe, a tumore, ad avvelenamento, ad annegamento, a emorragia, a ictus, ad AIDS o ad altre malattie e disordini; in seguito a questi eventi il sistema nervoso del paziente può risultare danneggiato e quindi non essere più in grado di controllare l'intero corpo o parte di esso [63, 64];

Paralisi cerebrale infantile *(Cerebral palsy)*: "paralisi cerebrale infantile" è un termine usato per indicare un insieme di disordini neurologici causati da uno sviluppo abnormale di quella parte del cervello che controlla il movimento muscolare; la paralisi cerebrale infantile si manifesta durante l'infanzia, non degenera con il passare del tempo, ma compromette in maniera più o meno grave la coordinazione muscolare e la capacià di muoversi [62];

Distrofie muscolari *(Muscolar dystrophies)*: "distrofia muscolare", o "miopatia ereditaria", è un termine usato per indicare un gruppo di malattie neuromuscolari degenerative che causano una progressiva debolezza muscolare e la perdita di tessuto muscolare fino a limitare o impedire il movimento volontario [36];

Sclerosi multipla *(Multiple sclerosis)*: la sclerosi multipla, o sclerosi a placche, è una malattia del sistema nervoso centrale molto diffusa le cui cause non sono ancora completamente note; la sclerosi multipla fa sì che le difese autoimmunitarie del corpo umano agiscano contro la mielina e contro il sistema nervoso, come se si trattasse di corpi estranei; quando la mielina, che riveste le fibre nervose e permette una trasmissione efficiente degli impulsi nervosi, viene rimossa si formano delle placche; questa lesione ha lo stesso effetto che avrebbe rimuovere il materiale isolante attorno a un cavo elettrico; i segnali nervosi non possono più essere trasmessi come in precedenza e questo causa in molti pazienti una paralisi più o meno estesa [1].

In questi casi, come nel caso di molte altre malattie che colpiscono e compromettono il sistema nervoso, il paziente potrebbe essere del tutto privo di controllo sui suoi muscoli ed essere dunque incapace di utilizzare quelli che sono gli attuali dispositivi creati per permettere alle persone paralizzate di comunicare; tutti questi dispositivi richiedono infatti che il paziente abbia almeno il controllo di un

muscolo, ad esempio il muscolo oculare. Pertanto le interfacce cervello-computer possono rappresentare un notevole miglioramento della qualità della vita se non addirittura una ragione per vivere per questi pazienti altrimenti impossibilitati a comunicare [85, 5].

Inoltre, gli sviluppi della tecnologia per le interfacce cervello-computer possono interessare anche utenti sani; sebbene ancora agli albori, le interacce cervello-computer costituiscono una nuova via di comunicazione tra l'uomo e la macchina e potrebbero un giorno diventare il paradigma di comunicazione principale tra l'utente e il computer [4] o, addirittura, come profetizzato da J.J. Vidal [82], essere il punto di unione tra la mente umana e l'intelligenza artificiale.

4.5 Classificazione delle interfacce cervello-computer

Abbiamo definito un'interfaccia cervello-computer come un dispositivo che consente una comunicazione uomo-macchina basata solo su segnali generati dall'attività neurale del cervello e indipendente da qualsiasi altro segnale; di seguito si analizza in quali modi tali dispositivi possano essere realizzati; in particolare si evidenzia quali siano le maggiori differenze che intercorrono tra varie tipologie di interfacce cervello-computer e come, in base a tali differenze, sia possibile definire una classificazione delle interfacce cervello-computer.

4.5.1 Classificazione in base alla posizione dei dispositivi di acquisizione

Anzitutto, è possibile distinguere le interfacce cervello-computer in merito al posizionamento dei dispositivi che vengono utilizzati per acquisire il segnale di un utente; vi sono due classi principali di interfacce cervello-computer [88]:

Non invasive si tratta di interfacce cervello-computer che utilizzano dispositivi la cui collocazione è esterna all'organismo del paziente e non richiedono dunque alcun tipo di intervento chirurgico. Ad esempio, sono interfacce cervello-computer non invasive quelle basate sull'elettroencefalografia;

Invasive si tratta di interfacce cervello-computer che utilizzano dispositivi impiantati sul cranio o all'interno di questo per mezzo di un'operazione chirurgica. Ad esempio, sono interfacce cervello-computer invasive quelle intracorticali che utilizzano microelettrodi collocati direttamente sulla corteccia cerebrale.

La distinzione tra interfacce cervello-computer non-invasive e invasive è di particolare importanza in quanto la decisione di adottare una classe di interfacce piuttosto che un'altra comporta rischi, costi e l'utilizzo di tecnologie estremamente differenti.

4.5.2 Classificazione in base al controllo nervoso-muscolare dell'utente

Una seconda classificazione è basata sulla relazione tra l'interfaccia cervello-computer e il sistema nervoso e muscolare del paziente; sebbene in base alla definizione data il contenuto della comunicazione in un'interfaccia cervello-computer dipenda esclusivamente da segnali generati dall'attività neurale del cervello e sia del tutto indipendente da altri segnali, è possibile che l'interfaccia cervello-computer richieda all'utente di avere un certo grado di controllo nervoso o muscolare per poter funzionare correttamente; si possono distinguere le interfacce cervello-computer tra [88]:

Dipendenti si tratta di interfacce cervello-computer che per poter funzionare richiedono che l'utente abbia un minimo controllo delle sue normali vie di input o di output. Ad esempio, per utilizzare interfacce cervello-computer basate su Visual Evoked Potentials (VEP) è richiesto che il paziente sia in grado di fissare il proprio sguardo su un target lampeggiante [88]; tuttavia, per fare questo, il paziente deve essere in grado di controllare i propri muscoli oculari per volgere opportunamente il proprio sguardo dove necessario[8].

Indipendenti si tratta di interfacce cervello-computer che per poter funzionare non richiedono all'utente nessun controllo delle sue normali vie di input o di output. Ad esempio, le interfacce cervello-computer basate sui ritmi sensomotori non necessitano di alcun controllo muscolare da parte dell'utente.

La distinzione tra interfacce cervello-computer dipendenti e indipendenti ha un'importanza fondamentale in quanto determina la popolazione che potrà adottare tali interfacce; è chiaro infatti che i possibili utilizzatori di interfacce cervello-computer indipendenti costituiscono un sottoinsieme di coloro che possono invece usufruire di interfacce cervello-computer dipendenti; in particolare i pazienti locked-in, ovvero i pazienti totalmente paralizzati, che potrebbero trarre i maggiori vantaggi da queste tecnologie, risulterebbero esclusi dai benefici di un'interfaccia cervello-computer dipendente.

4.5.3 Classificazione in base alla modalità di attivazione

Un'ulteriore dimensione secondo la quale è possibile distinguere differenti tipi di interfacce cervello-computer è quella relativa alla modalità e alla tempistica con cui un'interfaccia cervello-computer si attiva e opera; a seconda del ruolo della macchina, si possono distinguere interfacce cervello-computer [35]:

[8] Si noti che un'interfaccia cervello-computer dipendente, anche se a prima vista non sembra basarsi esclusivamente su segnali generati dall'attività neurale del cervello e pare dunque violare la definizione stessa di interfaccia cervello-computer, in realtà rispetta tale definizione; il contenuto della comunicazione dall'uomo alla macchina è infatti determinato esclusivamente sulla base dell'attività neurale del paziente e dunque tali interfacce cervello-computer dipendenti possono considerarsi a piena ragione delle vere interfacce cervello-computer.

Sincrone si tratta di interfacce cervello-computer in cui i tempi di comunicazione sono dettati dalla macchina; spetta cioè al computer determinare gli istanti in cui deve esserci una comunicazione e tocca all'utente rispettare questi tempi. Ad esempio, le interfacce cervello-computer basate su P300, nelle quali un utente deve prestare attenzione a una serie di stimoli visivi, sono sincrone in quanto è la macchina a determinare quando presentare all'utente tali stimoli visivi;

Asincrone si tratta di interfacce cervello-computer in cui i tempi di comunicazione sono dettati dall'uomo; è l'utente che decide di iniziare una comunicazione con il computer e quest'ultimo, riconosciuta la volontà del paziente, elabora il segnale che riceve in input.

La distinzione tra interfacce cervello-computer sincrone e asincrone è importante per poter valutare l'usabilità di tali dispositivi nel mondo reale; è infatti chiaro che un'interfaccia cervello-computer di entrambi i tipi può funzionare senza problemi in un ambiente di test come un laboratorio, ma è altrettanto evidente che nella vita quotidiana risulta molto più sensato avere un'interfaccia cervello-computer asincrona che autonomamente riconosca le intenzioni di un paziente in qualsiasi istante questo le voglia esprimere.

4.5.4 Classificazione in base ai segnali utilizzati

Si possono catalogare le interfacce cervello-computer anche in base al tipo di attività neurale che utilizzano per la comunicazione uomo-macchina; esistono due macro tipologie di segnali che possono essere utilizzati [88]:

Evocati (o event-related) si tratta di segnali generati dall'attività neurale del cervello in seguito a un dato evento, solitamente la generazione di uno stimolo sensorio stereotipato da parte dell'interfaccia cervello-computer. Ad esempio, le interfacce cervello-computer basate su P300 inducono l'utente a generare un pattern elettroencefalografico noto quando viene presentato un preciso stimolo visivo;

Spontanei (o user-generated) si tratta di segnali generati dall'attività neurale del cervello indipendentemente da stimoli esterni. Ad esempio, in un'interfaccia cervello-computer basata sui ritmi sensomotori, l'utente può imparare a regolare i ritmi registrati sulla corteccia sensomotoria indipendentemente da eventuali stimoli esterni.

La distinzione tra interfacce cervello-computer che usano segnali evocati o segnali spontanei ha una certa rilevanza in quanto contribuisce a determinare le tecnologie da adottare nell'interfaccia cervello-computer e la quantità di addestramento necessaria per l'utente prima di poter raggiungere un buon controllo dell'interfaccia cervello-computer.

Più precisamente, considerando queste due classi di segnali, è possibile identificare diversi sottotipi di segnali che rientrano in ciascuna di queste due categorie. Per quanto riguarda i segnali evocati si hanno:

P300 P300 è un picco positivo registrato dall'elettroencefalografo che appare circa 300 millisecondi dopo che all'utente viene presentato uno stimolo raro e rilevante rispetto all'attività che sta eseguendo [35]. P300 è un segnale generato dal cervello che sembra riflettere un processo cognitivo di alto livello che coinvolge la valutazione dello stimolo ricevuto e un aggiornamento della propria visione del mondo [4]; tuttavia il significato funzionale di P300 non è ancora completamente noto così come la sua origine e la sua natura: non tutti i segnali P300 sono uguali tra loro e attualmente sono noti almeno due sottotipi di P300, P3a e P3b [4];

VEP (Visual Evoked Potentials) VEPs sono segnali registrati sullo scalpo al di sopra della corteccia visiva in seguito a brevi stimoli visivi; per mezzo dell'analisi dei segnali VEPs è possibile determinare la direzione nella quale un utente sta volgendo il proprio sguardo [88]. Esistono oggi degli standard per la generazione e il riconoscimento di differenti tipi di VEPs, tra cui Steady State VEPs (SSVEPs), Sweep VEPs, Motor VEPs, Chromatic Color VEPs e diversi altri, di cui è disponibile un elenco in [61];

SSVEPs (Steady State Visual Evoked Potentials) SSVEPs, chiamati anche SSVERs (Steady State Visual Evoked Responses), denotano dei segnali generati dall'attività neurale del cervello in seguito alla presentazione di uno stimolo visivo lampeggiante a una determinata frequenza; infatti è noto che di fronte a un input che lampeggia a una data frequenza, gruppi di neuroni nelle aree visive si attivano alla stessa frequenza [4];

RPs (Readiness Potentials) RPs sono segnali generati dall'attività neurale del cervello prima di un determinato evento; a differenza dunque dei segnali precedenti che sono generati in risposta a un evento, gli RPs sono generati in anticipo rispetto all'evento stesso: rappresentano la prontezza da parte dell'utente a rispondere a un preciso evento, l'intenzione di compiere un'azione prima che questa sia realmente eseguita [4]. Riuscire a identificare l'attività cerebrale che precede un pensiero e determinarne la tempistica sono obiettivi molto utili in quanto fornirebbero informazione utile a un'interfaccia cervello-computer, ma sono ardui da raggiungere [4]; anche gli RPs, come altri segnali generati, sono infatti costituiti da diverse componenti, ciascuna delle quali ha una propria funzione e una propria utilità [4].

Per quanto riguarda invece i segnali spontanei, i più utilizzati per la realizzazione di interfacce cervello-computer sono:

SCP (Slow Cortical Potentials) SCPs sono segnali registrati dall'elettroencefalografo che rispecchiano dei cambiamenti lenti di voltaggio sulla corteccia cerebrale; tali cambiamenti avvengono nell'arco di 0.5-10 secondi; possono essere SCP negativi, associati generalmente con il movimento o altre funzioni che richiedono l'attivazione di aree della corteccia, o SCP positivi, che indicano invece una riduzione dell'attivazione della corteccia cerebrale [88]. All'interno dei segnali SCP è possibile distinguere

diversi componenti, tra cui CNV (Contingent Negative Variation), SPN (Stimulus Preceding Negativity), PINV (Post Imperative Negative Variation), RP (Readiness Potentials), ciascuno dei quali identificabile tramite elettroencefalogramma [4].

SMR (Sensory-Motor Rhythms) i ritmi sensomotori sono dei ritmi di tipo oscillatorio generati sulla corteccia sensomotoria dall'attività neurale del cervello; studiando le variazioni di questi ritmi è possibile determinare cambiamenti nello stato dei pazienti [35]. I ritmi sensomotori più utilizzati sono il ritmo mu e il ritmo beta; entrambi questi ritmi mostrano delle variazioni nel momento in cui l'utente decide di compiere un'azione motoria o, semplicemente, immagina di compiere un'azione motoria [88, 4, 35].

Neuronal Ensemble Activity la neuronal ensemble activity è un segnale generato da un ristretto numero di neuroni per comunicare informazioni ad altri neuroni; per mezzo di tecniche con alta risoluzione spaziale (tipicamente tecniche invasive) è possibile registrare i potenziali di azione, l'unità comunicativa base all'interno del cervello, e determinare così la volontà di un utente con grande precisione [35].

4.5.5 Classificazione in base alle applicazioni supportate

Un'ultima distinzione che è possibile considerare e in base alla quale classificare le interfacce cervello-computer è quella che riguarda le applicazioni supportate dall'interfaccia cervello-computer; le applicazioni più comuni sviluppate sono [35, 28]:

Spelling applications si tratta di applicazioni create per permettere a un utente di comporre frasi e parole attraverso la selezione di lettere (e.g., l'applicazione di spelling descritta da Birbaumer in [6]);

Environmental control applications si tratta di applicazioni create per permettere a un utente di interagire con un ambiente o con dei dispositivi controllabili elettronicamente (e.g., applicazioni per il contollo di elettrodomestici casalinghi, quali televisione o telefono);

Controllo di neuroprotesi si tratta di applicazioni create per permettere a un utente di gestire e controllare una protesi artificiale (e.g., applicazioni per il controllo di braccia o mani robotiche);

Controllo di una carrozzina per disabili si tratta di applicazioni create per permettere a un utente disabile di muovere la propria carrozzina;

Gaming applications si tratta di applicazioni create per permettere a un utente di eseguire dei semplici videogiochi;

Virtual reality applications si tratta di applicazioni create per permettere a un utente di interagire con una realtà virtuale tridimensionale e immersiva.

Determinare il tipo di applicazione supportato da un'interfaccia cervello-computer ha una notevole importanza in quanto essa influisce sull'utilità e sui possibili fruitori dell'interfaccia cervello-computer.

4.6 Stato dell'arte delle interfacce cervello-computer

La ricerca sulle interfacce cervello-computer ha prodotto diversi esemplari di interfacce funzionanti.

I primi progetti di interfacce cervello-computer risalgono al 1973 quando J.J. Vidal delineava in [82] un progetto per un'interfaccia cervello-computer non-invasiva che per mezzo di un elettroencefalografo riconoscesse variazioni dell'attività cerebrale indotte da stimoli esterni.

Da allora fino a oggi gli studi nel campo delle interfacce cervello-computer sono proseguiti e progrediti grazie all'apporto di diversi gruppi di ricerca in tutto il mondo che hanno contribuito allo sviluppo di questa tecnologia, creando le proprie implementazioni di interfacce cervello-computer.

4.6.1 Interfacce cervello-computer non-invasive basate su ritmi sensomotori (SMRs)

Tra i gruppi più attivi nella ricerca sulle interfacce cervello-computer non-invasive basate su ritmi sensomotori vi è senza dubbio il Wadsworth Center, costituito da J.R. Wolpaw, D.J. McFarland e i loro colleghi; impegnati nella ricerca dal 1986 [4], hanno realizzato un'interfaccia cervello-computer, chiamata Wadsworth BCI (Brain-Computer Interface), che si fonda sui ritmi sensomotori mu e beta registrati da un elettroencefalografo; diversi utenti, sani e disabili, hanno utilizzato questa interfaccia imparando, dopo un addestramento di alcune settimane, a controllare un cursore su uno schermo [88]; il controllo a una dimensione fu progettato nel 1990 [57] e implementato l'anno seguente [4], mentre il controllo a due dimensioni venne raggiunto per la prima volta nel 1994 [4].

Un'interfaccia cervello-computer analoga a quella del Wadsworth Center è stata sviluppata e presentata nel corso del 2000 dal gruppo di ricerca di Kostov e Polak e dal gruppo di ricerca di Penny, che hanno esplorato metodi di elaborazione dei segnali alternativi a quelli proposti da J.R. Wolpaw e dai suoi colleghi [88, 4].

Di grande interesse è anche l'interfaccia cervello-computer sviluppata dal team di ricerca di Pineda nel 2003; sempre basata sull'uso di ritmi sensomotori, questa interfaccia cervello-computer si distingue dalle precedenti in quanto è stata una delle prime a supportare un'applicazione di 3D gaming, ovvero uno sparatutto tridimensionale all'interno di un ambiente virtuale [4].

Risultati molto incoraggianti sono stati ottenuti presso l'Università di Graz da un team condotto da G. Pfurtscheller; la Graz BCI, realizzata tra il 1993 e il 1996 [4], è un'interfaccia cervello-computer non-invasiva anch'essa fondata sull'identificazione di sincronizzazioni e desincronizzazioni dei ritmi sensomotori

mu e beta per mezzo di un elettroencefalografo [88]; dal momento della sua realizzazione fino al 2003, la Graz BCI è stata utilizzata con svariate applicazioni, tra cui un'applicazione di spelling chiamata Virtual Keyboard, un'applicazione per il controllo di una mano meccanica e l'addestramento remoto di pazienti [68].

Buoni risultati sono stati ottenuti anche, parallelamente a questa tesi, da T. D'Albis [19] presso il Laboratorio di Intelligenza Artificiale e Robotica del Politecnico di Milano, nello sviluppo di un'interfaccia cervello-computer non-invasiva. Tale interfaccia cervello-computer è stata progettata per supportare applicazioni di spelling. A differenza di altre interfacce cervello-computer che cercano di aumentare le prestazioni e migliorare l'usabilità concentrandosi esclusivamente su tecniche per l'estrazione e l'elaborazione del segnale, questa interfaccia cervello-computer cerca anche di ottimizzare il modo in cui l'informazione acquisita è utilizzata per la comunicazione verbale. Per fare questo, i tradizionali moduli di elaborazione dei segnali dell'interfaccia cervello-computer sono stati integrati con un modulo di elaborazione del linguaggio naturale in grado di predire dal contesto i possibili sviluppi futuri della comunicazione. Sebbene i risultati ottenuti varino da soggetto a soggetto, l'accuratezza e la velocità di comunicazione raggiunta in seguito all'addestramento si sono dimostrati al livello dei risultati presenti in letteratura.

In generale, le interfacce cervello-computer non-invasive basate su ritmi sensomotori si sono dimostrate in grado di fornire buone performance, con indici di errori piuttosto contenuti; il loro svantaggio principale è la neccessità di un lungo periodo di addestramento, che può durare anche mesi, e, alla fine del quale, non è garantito che l'utente acquisisca davvero un buon controllo [60].

4.6.2 Interfacce cervello-computer non-invasive basate su slow cortical potentials (SCPs)

Sempre tra le interfacce cervello-computer non-invasive, un riconoscimento importante meritano i risultati ottenuti dai ricercatori dell'università di Tubingen, guidati da N. Birbaumer, per la loro interfaccia cervello-computer che utilizza gli slow cortical potentials. Gli slow cortical potentials sono stati registrati per mezzo di un elettroencefalografo e usati per gestire diverse possibili applicazioni, tra cui quelle di spelling e quelle per fornire risposte positive o negative a una domanda [5]. Il progetto più noto è stato il Thought Translation Device (TTD), realizzato nel 1999 [4], un'applicazione di spelling specificamente rivolta a pazienti disabili.

Anche le interfacce cervello-computer non-invasive basate su slow cortical potentials, così come quelle basate su ritmi sensomotori, richiedono un lungo periodo di addestramento al termine del quale non è assicurato che l'utente acquisisca un buon controllo. Inoltre, queste interfacce cervello-computer sembrano avere una velocità inferiore (spesso dovuta all'uso solo di selezioni binarie) e una percentuale di errore superiore rispetto alle applicazioni che utilizzano interfacce cervello-computer basate su ritmi sensomotori [60].

4.6.3 Interfacce cervello-computer non-invasive basate su P300

La prima interfaccia cervello-computer non-invasiva basata su P300 risale al 1988; il gruppo di ricerca diretto da L. Farwell della University of Illinois e E. Donchin della University of South California di Tampa realizzò un'interfaccia cervello-computer per applicazioni di spelling [85] che presentava all'utente una griglia di lettere; su questa griglia, a intervalli di tempo definiti, ciascuna colonna e ciascuna riga lampeggiava; osservando i picchi di P300 registrati dall'elettroencefalografo era possibile determinare quale lettera i pazienti volessero selezionare; in questo modo era possibile comporre parole che potevano poi essere lette da un sintetizzatore vocale [4, 60].

Nel 1995 il gruppo guidato da Polikoff creò un'interfaccia cervello-computer non-invasiva basata su P300 per mezzo della quale era possibile controllare il movimento di un cursore su uno schermo; il paziente doveva concentrarsi su un'icona corrispondente alla direzione in cui desiderava muoversi e contare quante volte tale icona lampeggiasse; registrando l'attività cerebrale in corrispondenza all'attivazione dell'icona desiderata era possibile determinare in che direzione l'utente intendesse muoversi [4].

Altre interfacce cervello-computer non-invasive basate su P300 sono state sviluppate in seguito da diversi istituti di ricerca; tra questi si ricorda l'interfaccia cervello-computer ottimizzata per l'environmental control realizzata da J. Bayliss nel 2001 alla University of Rochester [60], quella sviluppata da Hilit nel 2003 [4] e quella dell'Ecole Polytechnique de Lausanne per environmental control, sviluppata da U. Hoffmann e presentata nel 2007 [35].

Rispetto alle precedenti, le interfacce cervello-computer basate su P300 hanno il vantaggio di richiedere un addestramento minimo, nell'ordine dei minuti, giacchè la risposta P300 è una risposta naturale del cervello a uno stimolo non frequente; a questo si aggiunge una bassa percentuale di errore, anche se la velocità di tali dispositivi resta comunque limitata [60].

4.6.4 Interfacce cervello-computer non-invasive basate su visual evoked potentials (VEPs)

La prima interfaccia cervello-computer non-invasiva basata su visual evoked potentials può essere considerata proprio quella proposta da J.J. Vidal nel 1973 [82]; realizzata nel 1977 dallo stesso J.J. Vidal, questa interfaccia cervello-computer utilizzava quattro elettrodi per registrare l'attività elettroencefalografica di un utente e determinarne l'intenzione [4].

Tra le successive interfacce cervello-computer basate su visual evoked potentials si ricorda l'interfaccia cervello-computer presentata nel 1992 dal gruppo di ricerca di E. Sutter dello Smith-Kettlewell Institute of San Francisco; denominata Brain Response Interface (BRI), essa presentava all'utente una griglia di lettere o parole lampeggianti ed era in grado di determinare la direzione dello sguardo dell'utente e definire così l'elemento che egli voleva selezionare [4, 60].

Nel 2000 Middendorf e i suoi colleghi realizzarono un'interfaccia cervello-computer basata su steady state visual evoked potentials; la loro interfaccia cervello-computer presentava agli utenti dei rettangoli lampeggianti a una determinata frequenza; gli utilizzatori, concentrandosi su un rettangolo piuttosto che un altro, potevano rapidamente controllare i propri steady state visual evoked potentials [60].

Un miglioramento nell'uso dei steady state visual evoked potentials fu realizzato nel 2002 da Gao; la sua interfaccia cervello-computer era infatti in grado di distinguere tra quasi quaranta stimoli diversi presentati all'utente e selezionare, con una bassa percentuale di errore, lo stimolo scelto dall'utente [4].

In sintesi, le interfacce cervello-computer non-invasive basate su visual evoked potentials hanno dimostrato di poter raggiungere buone performance con basse percentuali di errori e con un periodo di addestramento ridotto, nell'ordine delle ore; tuttavia, dal momento che le interfacce cervello-computer basate su visual evoked potentials sono, per definizione, dipendenti e quindi rivolte a una classe di utenti più ristretta, esse hanno riscosso un interesse minore nel campo della ricerca rispetto ad altre tipologie di interfacce cervello-computer [4, 60].

4.6.5 Interfacce cervello-computer invasive

Le ricerche su interfacce cervello-computer invasive sono state condotte e descritte per la prima volta nel 1997 dal team di Huggins; utilizzando degli elettrodi già impiantati in pazienti affetti da gravi forme di epilessia, Huggins e i suoi colleghi hanno realizzato un'interfaccia cervello-computer offline in grado di discriminare con grande accuratezza fino a sei diversi tipi di azioni immaginate dall'utente [4]. Una versione online di questa interfaccia cervello-computer venne realizzata due anni più tardi da un gruppo di ricercatori condotto da Rohde [4].

Dal 1998, P. Kennedy, R. Bakay e P. Ehirim della società Neural Signals di Atlanta hanno realizzato delle interfacce cervello-computer invasive grazie alla collaborazione di pazienti malati di sclerosi laterale amiotrofica o colpiti da ictus; questi pazienti hanno accettato di prendere parte alla ricerca e si sono sottoposti a un intervento chirurgico per l'impianto degli elettrodi. In breve tempo, gli utenti hanno sviluppato un buon controllo dell'interfaccia cervello-computer; durante le prime sessioni, i pazienti usavano la propria immaginazione motoria per controllare l'interfaccia cervello-computer, ma con il passare del tempo hanno appreso a controllarla con la semplice volontà [4, 85].

Diversi studi sulle interfacce cervello-computer sono stati condotti anche su cavie non umane; questi esperimenti hanno permesso di studiare l'implementazione e le prestazioni di potenziali interfacce cervello-computer evitando il rischio di un'operazione chirurgica su un essere umano. Nel 1999, il team di J. Chapin della Pennsylvania-Hanemann School of Medicine e il team di M. Nicolelis della Duke University hanno dimostrato come un ratto, dotato di un'interfaccia cervello-computer invasiva, possa imparare a controllare un braccio robotico per procurarsi dell'acqua [4, 85]. L'anno seguente, il team di Wessberg ha condotto un esperimento analogo, con risultati altrettanto promettenti, usando però co-

me cavia una scimmia al posto di un topo [4]. Nel 2002, Taylor e A. Schwartz del La Jolla Neurosciences Institute hanno insegnato a una scimmia a controllare un braccio robotico in tempo reale in tre dimensioni per mezzo di elettrodi impiantati all'interno della corteccia cerebrale [4, 85, 5]. Alla Brown University, J. Donoghue e M. Serruya hanno realizzato interfacce cervello-computer simili, denominate Neuromotor Prostheses (NMP) in grado di monitorare l'attività di gruppi di neuroni e controllare un cursore all'interno di applicazioni informatiche [34].

Escluse le difficoltà tecniche, mediche e legali per eseguire l'impianto degli elettrodi, le interfacce cervello-computer invasive hanno dimostrato di poter raggiungere alti livelli di performance e precisione; la maggior fedeltà dei segnali registrati e i successi nello studio di queste interfacce sugli animali sembrano prospettare buoni sviluppi per le interfacce cervello-computer invasive [60].

4.7 Ritmi sensomotori

Dopo aver presentato le diverse tipologie di interfacce cervello-computer esistenti, è possibile spiegare la decisione presa nel nostro laboratorio di studiare un'interfaccia cervello-computer non-invasiva basata su ritmi sensomotori. Anzitutto la precedente scelta di utilizzare un elettroencefalografo come mezzo per registrare l'attività encefalografica ha precluso la possibilità di realizzare un'interfaccia cervello-computer invasiva. In secondo luogo, dal momento che si intendeva sviluppare un'interfaccia cervello-computer che potesse soddisfare quanto più possibile le reali esigenze dei suoi destinatari ideali, ovvero i pazienti affetti da sindrome *locked-in*, sono state escluse le interfacce cervello-computer basate su evoked visual potentials, in quanto interfacce cervello-computer dipendenti. Inoltre, desiderando evitare l'utilizzo di stimoli esterni, si sono scartate anche le interfacce cervello-computer basate su P300, in quanto, per funzionare, richiedono che sia presente un input sensoriale in grado di generare un picco di P300. Infine, tra le interfacce cervello-computer basate su slow cortical potentials e le interfacce cervello-computer basate su ritmi sensomotori si è deciso di propendere per le ultime, dal momento che le performance e i risultati pubblicati in letteratura appaiono più promettenti.

Nello sviluppo di interfacce cervello computer sensomotorie si ricorre generalmente a ritmi elettroencefalografici noti: i ritmi mu e beta. Tali ritmi, già presentati nel Paragrafo 3.2.4, risultano particolarmente adatti al controllo di un'interfaccia cervello-computer sensomotoria per diverse ragioni [91]. Innanzitutto entrambi i ritmi sono prodotti dalla corteccia sensomotoria, ovvero in quell'area del cervello responsabile del movimento; in secondo luogo, l'esperienza e gli studi precedenti hanno dimostrato che gli utenti di un'interfaccia cervello-computer, siano essi sani o disabili, imparano con il tempo a modificare l'ampiezza del proprio ritmo mu per controllare al meglio l'interfaccia cervello-computer [92]. Inoltre, nonostante la variabilità del ritmo mu da soggetto a soggetto, ulteriori ricerche in campo neurofisiologico hanno confermato che il ritmo mu è presente in quasi tutti i soggetti adulti [91]. L'interesse per l'area

sensomotoria del cervello è supportato dalla letteratura medica che ha dimostrato un forte parallelismo tra movimento e immaginazione del movimento: compiere un movimento o semplicemente immaginarlo richiedono lo stesso tempo, generano la stessa risposta vegetativa (i.e., incremento del ritmo cardiaco o della pressione arteriosa), sono soggetti alle stesse limitazioni fisiche (e.g., il trade-off tra velocità e precisione di un movimento) e, non da ultimo, attivano le stesse regioni del cervello [45]. Infine, il ritmo mu risulta anche essere un ritmo facile da riconoscere [89]: l'immaginazione del movimento provoca infatti un'attenuazione dell'ampiezza del ritmo mu nell'emisfero controlaterale rispetto alla parte del corpo che si è immaginato di muovere.

4.8 Criteri di valutazione delle performance delle interfacce cervello-computer

Esistono diversi criteri e metriche per valutare le performance di un'interfaccia cervello-computer o di un'applicazione di una interfaccia cervello-computer [46]. Usando una classificazione presentata nel corso del BCI Meeting 2005 possiamo distinguere questi metodi nei gruppi seguenti:

1. *Metodi elementari per la valutazione delle performance di un'applicazione di un'interfaccia cervello-computer*: si tratta di metodi semplici che prendono in considerazione esclusivamente i risultati degli esperimenti eseguiti con un'applicazione di un'interfaccia cervello-computer; tra questi la percentuale di errore (error rate), l'accuratezza (accuracy), la velocità (rate o rapidity), il tempo per completare un compito (time to complete a task) e l'informazione trasferita (bit rate);

2. *Metodi alternativi per la valutazione delle performance di un'interfaccia cervello-computer*: si tratta di metodi che permettono di valutare quantitativamente le performance di una interfaccia cervello-computer analizzando il segnale registrato o il controllo raggiunto dall'utente; tra questi il rapporto segnale-rumore (signal-to-noise ratio, SNR), il coefficiente di determinazione (determination coefficient, r^2) e la cappa di Kohen (Kohen's kappa);

3. *Metodi basati sulla teoria di signal detection*: si tratta di metodi che si fondano sulla teoria del signal detection; tra questi la curva delle caratteristiche operative del ricevitore (receiver operating characteristics curve, ROC).

Per completezza, è giusto precisare che esistono anche altri criteri per la valutazione e il confronto di interfacce cervello-computer [87]; alcuni di questi sono parametri generici o, comunque, di minore interesse rispetto a quelli sopra elencati; essi consentono infatti un confronto tra diverse tipologie di interfacce cervello-computer, ma solo a livello qualitativo, senza che sia possibile istituire un confronto quantitativo tra le prestazioni delle applicazioni di interfacce cervello-computer; tra questi criteri:

- *Response time*: tempo necessario all'interfaccia cervello-computer per elaborare un segnale e, eventualmente, restituire un output in un'applicazione;
- *Training*: tempo e modalità di training (se presente) necessari ad acquisire il controllo dell'interfaccia cervello-computer;
- *User population*: utenti a cui è rivolta l'interfaccia cervello-computer[9];
- *Supported applications*: applicazioni che possono essere supportate dall'interfaccia cervello-computer;
- *On-off method*: modalità di attivazione e disattivazione dell'interfaccia cervello-computer;
- *Constraints on input and output*: vincoli imposti all'utente durante l'utilizzo dell'interfaccia cervello computer (e.g., il vincolo di rimanere fermo oppure il vincolo di non parlare).

4.8.1 Metodi elementari per la valutazione delle performance di un'applicazione di un'interfaccia cervello-computer

Si esaminano ora con maggior dettaglio i metodi elementari per la valutazione delle performance di un'applicazione di un'interfaccia cervello-computer elencati nel precedente paragrafo.

Si consideri un'applicazione di un'interfaccia cervello-computer in cui l'esito di ciascuna prova (trial) sia un successo (hit) o un fallimento (miss) rispetto a un obiettivo (target[10]) predeterminato (e.g., muovere un cursore per colpire una parte dello schermo evidenziata).
Si può, allora, definire l'error rate come:

$$Error\,rate = \frac{misses}{misses + hits}.$$

L'error rate è dunque il rapporto tra il numero di fallimenti e il numero totale di prove. In generale, la misura fornita dall'error rate risulta di limitata utilità dal momento che essa non prende in considerazione la presenza di falsi positivi o falsi negativi (e.g., si consideri un utente del tutto privo di controllo

[9] Si tiene a precisare che la user population rappresenta un criterio di valutazione di minore importanza solo da un punto di vista ingegneristico e numerico; non si dimentichi che la finalità principale delle interfacce cervello-computer non è quella di garantire un sistema massimamente efficiente fine a sè stesso, ma quella di offrire una via di comunicazione a pazienti che diversamente non avrebbero altro modo di relazionarsi con il mondo esterno. Perciò, quando si valuta un'interfaccia cervello-computer per le sue reali possibilità applicative, la user population dovrebbe essere invece un criterio di massima rilevanza.

[10] La terminologia trial, hit, miss e target deriva da quella adottata da J.R. Wolpaw e dai suoi colleghi nell'eseguire i primi esperimenti di controllo di un cursore per mezzo di un'interfaccia cervello-computer presso il Wadsworth Center. La stessa terminologia è stata adottata all'interno del nostro laboratorio di interfacce cervello-computer durante gli esperimenti.

sull'applicazione cervello-computer che raggiunga involontariamente sempre lo stesso target, indipendentemente da quale sia il target corretto; nel caso l'applicazione dell'interfaccia cervello-computer proponga sempre il medesimo target raggiunto dall'utente, l'error rate sarebbe pari a 0; ma è evidente che questo risultato è privo di significato, poichè, per ipotesi, l'utente era privo di controllo e avrebbe dovuto ottenere un error rate maggiore di 0).

Complementare alla misura dell'error rate è l'accuracy [57], definita come:

$$Accuracy = \frac{hits}{misses + hits}.$$

L'accuracy è cioè il rapporto tra il numero di successi e il numero totale di prove. Si noti che l'accuracy è sottoposta alle stesse limitazioni dell'error rate.

In molte applicazioni di interfacce cervello-computer, un parametro di fondamentale rilevanza è non solo la precisione, ma anche il tempo impiegato per raggiungere un determinato risultato. Si consideri un'applicazione di un'interfaccia cervello-computer in cui all'utente è richiesto di completare un generico compito (task) (e.g., selezione di una lettera o composizione di una parola). Il time to complete a task (TCT) si definisce come:

$$TCT = \frac{tasks}{total\,time}.$$

Il time to complete a task misura dunque il rapporto tra il numero di compiti portati a termine e il tempo totale impiegato.

Nel caso un task corrisponda a un hit, il time to complete a task si può ridefinire come rate, ovvero rapporto tra gli hits e il tempo totale impiegato per ottenere gli hits.

Si noti che il time to complete a task può essere utilizzato solo per comparare interfacce cervello-computer che presentano applicazioni uniformi; per confrontare interfacce cervello-computer che supportano applicazioni non uniformi è necessario ricorrere ad altre metriche (e.g., bit rate).

I criteri finora esposti si adattano bene a specifiche applicazioni, in quanto valutano le prestazioni evidenziando singoli caratteri a discapito di altri. Occorrono invece nuove metriche capaci di tener conto dei diversi aspetti che contribuiscono a definire le performance di un'applicazione di un'interfaccia cervello-computer. Una possibile soluzione è offerta dal bit rate, la cui formulazione più generale è la seguente:

$$Bit\,rate = \frac{bits}{total\,time}.$$

Il bit rate rappresenta dunque la misura dell'informazione trasferita, ovvero del numero di bits trasferiti per unità di tempo (generalmente, in un minuto).

Tale metrica risulta però troppo semplificata per descrivere le reali performance di un'applicazione di un'interfaccia cervello-computer. Si consideri,

infatti, un'applicazione di un'interfaccia cervello-computer in cui l'esito di ciascuna prova (trial) sia un successo (hit) o un fallimento (miss) rispetto a un obiettivo (target) predeterminato (e.g., muovere un cursore per colpire una parte dello schermo evidenziata). In questo caso è evidente che non è sufficiente considerare il numero totale di bit trasferiti, ma anche la presenza di fallimenti o successi. Una misura più fondata, basata sulla teoria di Shannon, descritta da Pierce (1980) e riproposta da D.J. McFarland [50], ridefinisce il bit rate come segue:

$$bit\,rate\,per\,trial = \log_2 N + P \log_2 P + (1-P) \log_2 \frac{1-P}{N-1},$$

dove:

$$N = numero\,di\,target\,possibili,$$
$$P = probabilita'\,che\,il\,target\,corretto\,sia\,colpito\,(hit).$$

Il bit rate per trial è perciò la misura dell'informazione trasferita in ogni singolo trial; data la durata media di un trial è possibile calcolare il bit rate propriamente detto, ovvero il numero di bit trasferiti per unità di tempo (generalmente in un minuto). Si noti che questo nuovo valore del bit rate dipende da:

1. il tempo impiegato per un trial (maggiore è il tempo, minore è la quantità di informazione trasferita per unità di tempo);
2. il numero di target possibili (maggiore è il numero dei target, maggiore è il numero di bit necessari per identificarli, maggiore è l'informazione comunicata quando un target è selezionato);
3. la probabilità di ottenere un successo (maggiore è la probabilità di colpire il target corretto, maggiore è l'informazione utile trasmessa).

Questa metrica ha dunque il grande vantaggio di unire in un'unica valutazione il numero di target possibili, il tempo e l'accuracy evidenziando come il bit rate ottimale sia in realtà il risultato di un tradeoff tra questi parametri: aumentare a dismisura il numero di target e quindi l'informazione contenuta in ciascun target rischia di peggiorare l'accuracy a un livello tale da pregiudicare la comunicazione di qualsiasi informazione utile; viceversa, massimizzare l'accuracy diminuendo il numero di target o aumentando la durata dei trials, diminuisce drasticamente la quantità totale di informazioni trasmesse.

4.8.2 Metodi alternativi per la valutazione delle performance di un'interfaccia cervello-computer

Di seguito si propongono due approcci alternativi per la valutazione delle performance di un'interfaccia cervello-computer: il primo è un metodo basato sull'analisi del segnale e del rumore; il secondo è un metodo statistico mirato a valutare il grado di controllo raggiunto dall'utente nell'uso dell'interfaccia cervello-computer.

Signal-to-Noise Ratio (SNR)

Si consideri una qualsiasi interfaccia cervello-computer; il signal-to-noise ratio è definito come:

$$SNR = \frac{P_{signal}}{P_{noise}}.$$

Il signal to noise ratio misura dunque il rapporto tra la potenza del segnale analizzato e la potenza del rumore di fondo, ovvero quanto il segnale registrato è disturbato dal rumore [70]. Si noti che questa misura è indipendente dall'applicazione dell'interfaccia cervello-computer e dipende dalla strumentazione utilizzata dall'interfaccia cervello computer.

Determination Coefficient (r^2)

Si consideri un'applicazione di un'interfaccia cervello-computer in cui l'esito di ciascuna prova (trial) sia un successo (hit) o un fallimento (miss) rispetto a un obiettivo (target) predeterminato (e.g., muovere un cursore per colpire una parte dello schermo evidenziata); si supponga inoltre che il movimento sia determinato dall'ampiezza di un segnale su uno specifico canale (o dalla combinazione di più segnali). Allora, r^2 si definisce come [55]:

$$r^2 = \frac{SS_{exp}}{SS_{tot}}, \tag{4.1}$$

dove:

$$SS_{exp} = \sum_i (f_i - \bar{y})^2, \tag{4.2}$$

$$SS_{tot} = \sum_i (y_i - \bar{y})^2, \tag{4.3}$$

dove:

$$\begin{aligned} SS_{exp} &= \textit{varianza spiegata}, \\ SS_{tot} &= \textit{varianza totale}, \\ y_i &= \textit{valore osservato dell}'\, i - \textit{esimo elemento del dataset}, \\ f_i &= \textit{valore stimato dell}'\, i - \textit{esimo elemento del dataset}, \\ \bar{y} &= \textit{media dei dati osservati}. \end{aligned}$$

Dunque r^2 è il rapporto tra la varianza spiegata (SS_{exp}) e la varianza totale (SS_{tot}), ovvero il rapporto tra la varianza del modello predittivo e la varianza totale del processo osservato. Il valore di r^2 è un'informazione statistica che misura il grado di variabilità nei dati spiegato dal modello [55]; è un indice della bontà di adattamento di un modello stimato dai dati osservati: nel caso di

regressione lineare, un valore di r^2 pari a 1 indica un perfetto adattamento tra la regressione lineare e il fenomeno osservato, ovvero che tutta la varianza del processo è spiegata dalla regressione lineare; un valore di r^2 pari a 0 indica invece un adattamento praticamente nullo, ovvero che nessuna parte della varianza del processo è spiegata dalla regressione lineare [26]. Tuttavia, si noti che r^2 misura esclusivamente il grado di adattamento, ma non fornisce informazioni sul fatto che il modello regressivo sia realmente adatto a modellare i dati, sul valore della pendenza nella retta di regressione o sulla validità delle eventuali previsioni fornite dal modello regressivo [55].

Nella valutazione di un'applicazione di un'interfaccia cervello-computer, la varianza spiegata è data dal valore atteso del segnale corrispondente al target presentato (e.g., in un'applicazione di controllo monodimensionale verticale di un cursore mediante ampiezza del ritmo mu, si potrebbe attendere un valore dell'ampiezza del ritmo mu inferiore a 2 µv in presenza del target superiore e un valore superiore a 2 µv in presenza del target inferiore[11]), mentre la varianza totale è data dai valori registrati sperimentalmente; r^2 riflette perciò il livello di controllo dell'utente sul segnale da lui generato: un valore di r^2 pari a 1 indica un perfetto controllo del segnale, un valore di r^2 pari a 0 indica un controllo nullo [72].

Dall'analisi di r^2 è inoltre possibile determinare quale segnale (e.g., banda di frequenza) e da quale canale (e.g., posizione dell'elettrodo sullo scalpo) l'utente riesca a ottenere il controllo massimo sull'interfaccia cervello-computer [92].

La piattaforma BCI2000 offre, tra i vari moduli integrati, un tool per l'analisi offline; una delle funzionalità presenti è quella per il calcolo di r^2; all'interno del file Matlab *rsqu.m,* scritto da G. Schalk, il valore di r^2 è calcolato come segue:

$$r^2 = \frac{\frac{\left(\sum_{i=1}^{N_q} q_i\right)^2}{N_q} + \frac{\left(\sum_{j=1}^{N_r} r_j\right)^2}{N_r} - G}{\sum_{i=1}^{N_q}(q_i)^2 + \sum_{j=1}^{N_r}(r_j)^2 - G}, \tag{4.4}$$

dove:

$$G = \frac{\left(\sum_{i=1}^{N_q} q_i + \sum_{j=1}^{N_r} r_j\right)^2}{N_q + N_r}, \tag{4.5}$$

dove:

$$\begin{aligned} q,\, r &= \textit{vettori di dati}, \\ N_q,\, N_r &= \textit{numero di elementi dei vettori } q \textit{ e } r. \end{aligned}$$

A prima vista, la formula utilizzata in BCI2000 può apparire differente dalla definizione di r^2 data nelle righe precedenti; analizzando con maggior attenzione

[11] Questi valori sono puramente esemplificativi; la determinazione dell'output dell'applicazione in relazione all'ampiezza del ritmo considerato è molto più complessa di come presentata in questo esempio; vedi [91].

il codice prodotto da G. Schalk, si può facilmente dimostrare l'equivalenza con la definizione data (vedi Appendice).

Si noti inoltre che usando valori divisi in due classi e prendendo come valore atteso la media delle due classi, la misura di r^2 permette di valutare anche la varianza intra-classe (interna alle singole classi) e la varianza inter-classe (tra le due classi); un valore di r^2 pari a 1 indica una bassa varianza intra-classe e un'alta varianza inter-classe (grazie a cui è facile classificare il segnale dell'utente e controllare opportunamente l'applicazione), mentre un valore di r^2 pari a 0 corrisponde a un'alta varianza intra-classe e una bassa varianza inter-classe (per cui sarà difficile classificare il segnale dell'utente e controllare opportunamente l'applicazione) [72].

Capitolo 5

Analisi e classificazione del segnale per motor imagery

In questo capitolo si spiegano e analizzano i principali algoritmi di analisi e classificazione del segnale utilizzati in letteratura nello sviluppo di interfacce cervello-computer basate su motor imagery. Questi algoritmi sono stati poi implementati all'interno del nostro sistema di analisi e classificazione del segnale sviluppato presso il Laboratorio di Intelligenza Artificiale e Robotica, come spiegato nel capitolo successivo.

Lo schema generale di un'interfaccia cervello-computer può essere sinteticamente rappresentato come un modello a blocchi costituito da due elementi [88] (vedi Figura 5.1):

1. *Blocco di elaborazione del segnale:* il blocco di elaborazione del segnale costituisce il primo componente di ogni interfaccia cervello-computer; esso riceve in ingresso un segnale analogico (i.e., il segnale elettrico del cervello) e genera come output un segnale digitale (i.e., un segnale utilizzabile dal computer). Questo blocco è costituito a livello hardware dal dispositivo in grado di registrare un segnale e connettere il dispositivo a un computer (e.g., un elettroencefalografo e il cavo di connessione a un laptop), a livello software dai componenti necessari per inviare questo segnale a un computer (e.g., driver per interfacciare l'elettroencefalografo al computer). Oltre alla semplice digitalizzazione, il blocco di elaborazione del segnale può offrire anche altre funzionalità, tra cui un'amplificazione o un pre-filtraggio del segnale, che permettono di selezionare la quantità di dati trasferiti e comunicare solo informazioni utili. L'insieme di amplificazione, filtraggio e digitalizzazione costituisce quello che è spesso chiamato *preprocessing* (Paragrafo 5.1.1).

2. *Blocco di analisi del segnale*: il blocco di analisi del segnale costituisce il componente all'interno del quale il segnale precedentemente acquisito viene elaborato per produrre un risultato significativo; esso riceve in input

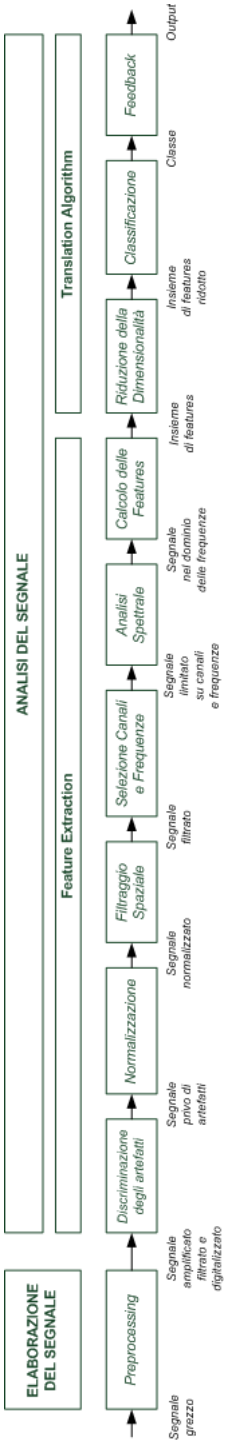

Figura 5.1: Schema a blocchi dell'analisi e della classificazione del segnale per motor imagery.

il segnale digitale prodotto dal blocco a monte (i.e., un segnale utilizzabile dal computer) e genera in output un comando o un feedback (e.g., la selezione di una lettera su uno schermo o il movimento di un arto robotico). A un'analisi più attenta è possibile distinguere due sottoblocchi che costituiscono questo blocco:

Feature extraction il blocco di feature extraction è costituito da tutte le procedure necessarie per estrarre ed evidenziare le caratteristiche proprie di un segnale; in questo blocco rientrano la discriminazione degli artefatti (Paragrafo 5.2.1), la normalizzazione (Paragrafo 5.2.2), il filtraggio spaziale (Paragrafo 5.2.3), la selezione dei canali e delle frequenze (Paragrafo 5.2.4), l'analisi in frequenza (Paragrafo 5.2.5) e il calcolo vero e proprio delle feature (Paragrafo 5.2.6).

Translation algorithm il blocco di translation algorithm è costituito da tutte le funzioni necessarie per tradurre il segnale generato dall'utente in un output corrispondente alla volontà dell'utente; tale traduzione si avvale, ovviamente, delle caratteristiche del segnale messe in rilievo nel sottoblocco precedente; in questo blocco rientrano le operazioni di riduzione della dimensionalità (Paragrafo 5.2.7), di classificazione (Paragrafo 5.2.8) e di feedback (Paragrafo 5.2.9).

Si noti che l'ordine con cui si presentano le fasi di elaborazione e classificazione del segnale, sebbene sensato e consequenziale, non corrisponde a una rigida successione: in particolari applicazioni di interfacce cervello-computer, alcune fasi potrebbero essere anticipate, posticipate, modificate o addirittura rimosse. Quella proposta nei paragrafi seguenti è dunque una possibilità, che esemplifica come il segnale prodotto da un utente può giungere a determinare l'output di un'interfaccia cervello-computer, in particolare della nostra interfaccia cervello-computer sensomotoria.

5.1 Elaborazione del segnale

5.1.1 Preprocessing del segnale

Prima di essere trasmesso a un computer per l'analisi, il segnale acquisito da un elettroencefalografo è generalmente sottoposto a tre elaborazioni [4]:

Amplificazione: il segnale acquisito è generalmente molto piccolo, nell'ordine di 100 microvolt; per favorirne l'analisi il segnale è generalmente amplificato di un fattore 5000, 10000 o anche 20000 [54, 49, 73];

Filtraggio: il segnale acquisito generalmente contiene molte frequenze, la maggior parte delle quali non sono di interesse all'interno di un'applicazione di un'interfaccia cervello-computer e possono introdurre rumore; per rimuovere queste componenti, il segnale può essere fatto passare attraverso un opportuno filtro; sinteticamente i filtri che possono essere implementati in un elettroencefalografo sono:

- Filtro passa-basso (lowpass filter): filtro che taglia tutte le frequenze superiori a una data soglia;
- Filtro passa-alto (highpass filter): filtro che taglia tutte le frequenze inferiori a una data soglia;
- Filtro passa-banda (bandpass filter): filtro che taglia tutte le frequenze esterne a un dato intervallo (questo filtro può essere considerato come una combinazione di un filtro passa-basso e di un filtro passa-alto);
- Filtro elimina-banda (notch filter): filtro che taglia tutte le frequenze interne a un dato intervallo.

Tipicamente, in un'applicazione di un'interfaccia cervello-computer, che generalmente utilizza frequenze comprese tra 3 Hz e 40 Hz, il filtraggio avviene usando un filtro passa-alto intorno a 0.1 Hz, un filtro passa-basso a 40-60 Hz [54, 49, 73] fino a 100 Hz [4] e infine un filtro elimina-banda a 50 o 60 Hz, per tagliare il rumore dovuto alla frequenza della corrente elettrica.

Digitalizzazione: per potere essere inviato e analizzato da un computer digitale, il segnale analogico deve essere convertito in un segnale digitale attraverso un apposito convertitore analogico-digitale (analog to digital converter, ADC); il segnale da analizzare è generalmente campionato a 128 Hz, 196 Hz [54, 49, 73] o 256 Hz. La frequenza a cui effettuare la digitalizzazione deve essere scelta tenendo in considerazione il teorema di Shannon: la frequenza di campionamento deve essere maggiore del doppio della massima frequenza di interesse; ad esempio, se l'intervallo di frequenze da studiare è 0.1-100 Hz, è opportuno un campionamento a 256 Hz (256 Hz è infatti ben al di sopra del doppio di 100 Hz). È comunque bene valutare con attenzione il valore del campionamento perchè, come un campionamento troppo basso rischia di perdere certe frequenze, così un campionamento troppo alto finirebbe solo per produrre una massa di dati superflui, molti dei quali ricavabili per interpolazione da altri dati.
Il numero di bit per rappresentare il segnale può variare da 8 bit (256 valori possibili), a 12 bit (4.096 valori possibili), a 16 bit (65.536 valori possibili) fino a rappresentazioni estremamente precise a 24 bit (16.777.216 valori possibili).

5.2 Analisi del segnale

5.2.1 Discriminazione degli artefatti

Generalmente, il segnale registrato durante un'elettroencefalografia è spesso disturbato da rumori esterni; nel corso di un'elettroencefalografia è infatti inevitabile che un soggetto compia diversi movimenti, molti dei quali involontari (e.g., sbattere le palpebre), che generano degli artefatti, ovvero del rumore elettrico

tale da mascherare il segnale che si intende rilevare. A seconda della loro origine, tali artefatti possono distinguersi in [47]:

Artefatti non-EEG gli artefatti non-EEG sono artefatti generati al di fuori dell'attività elettrica del cervello; in questa categoria rientrano lo sbattere delle palpebre, il movimento degli occhi, l'attività dei muscoli e il battito del cuore;

Artefatti EEG gli artefatti EEG sono artefatti generati dall'attività elettrica del cervello; in realtà essi sono parte imprescindibile del normale funzionamento del cervello e vengono definiti artefatti esclusivamente perchè hanno una frequenza prossima o uguale alla frequenza del segnale di interesse; nel caso di interfacce cervello-computer sensomotorie che usano la frequenza mu, la frequenza visuale alfa può costituire un artefatto.

Le sorgenti di rumore e il loro impatto sull'elettroencefalografia variano profondamente da soggetto a soggetto; questo rende particolarmente complessa la determinazione di cosa sia un artefatto e quando l'influenza di un artefatto sia tale da richiedere un intervento [4]; in quest'ultimo caso, ovvero quando si sia determinato che il rumore dell'artefatto è eccessivo e rischia di compromettere la registrazione del segnale di interesse, si possono adottatare due soluzioni differenti [60]:

Rimozione del segnale la scelta più semplice, rapida ed efficace consiste nell'utilizzo di metodi per l'esclusione della parte di registrazione elettroencefalografica contaminata dagli artefatti; è possibile, ad esempio, eliminare un segnale ogniqualvolta esso superi una determinata soglia; il riconoscimento della parte di elettroencefalogramma da eliminare può avvenire manualmente, da parte di un operatore umano, o in maniera automatica, da parte di un software; in entrambi i casi la validità di questi metodi si basa sulla qualità del riconoscimento degli artefatti. Lo svantaggio di questi metodi consiste nella perdita di dati, che, se eccessiva, rischia di compromettere il funzionamento dell'interfaccia cervello-computer; di conseguenza essi risultano inapplicabili in presenza di pazienti che generano un alto numero di artefatti (e.g., bambini o pazienti che a causa delle loro patologie hanno frequenti contrazioni muscolari involontarie) o nel caso l'interfaccia cervello-computer debba essere utilizzata in un ambiente diverso dall'ambiente protetto di un laboratorio in cui si può chiedere a un soggetto di minimizzare tutti i suoi movimenti.

Filtraggio del segnale qualora la rimozione del segnale non sia applicabile, è possibile ricorrere a più elaborati metodi di filtraggio del segnale; questi metodi si fondano sull'assunto che si possa considerare il segnale registrato come una combinazione lineare del segnale pulito, prodotto dall'elettroencefalografo, e del rumore. Le tecniche più semplici mirano a *pulire* il segnale registrato sottraendo a esso il valore del rumore moltiplicato per un opportuno peso; approcci più moderni comprendono l'uso di tecniche basate su indipendent component analysis (ICA), reti neurali e mappe di

Kohonen. La maggior parte di queste tecniche può essere usata sia offline, lavorando a posteriori sui dati registrati nel corso di una sessione di utilizzo di un'interfaccia cervello-computer, sia online, mentre si utilizza l'interfaccia cervello-computer; l'obiettivo primario è ancora quello di garantire un filtraggio tale da rendere possibile l'uso di interfacce cervello-computer anche al di fuori degli ambienti schermati dei laboratori.

Per applicare tecniche di filtraggio del rumore è necessario essere in grado di monitorare le sorgenti degli artefatti; per questa ragione, nel corso dell'utilizzo di un'interfaccia cervello-computer, oltre all'elettroencefalografo può essere necessario montare un elettrooculografo (EOG) per rilevare l'attività oculare e, più raramente, un elettromiografo (EMG) per registrare l'attività muscolare o un elettrocardiografo (ECG) per rilevare l'attività cardiaca.

Di tutti gli artefatti non-EEG presenti, quelli che influenzano maggiormente il segnale dell'elettroencefalografia sono quelli prodotti dall'attività oculare, sia per la loro intensità, sia per la loro prossimità ai punti di registrazione da parte dell'elettroencefalografo. Il bulbo oculare è infatti polarizzato, con la cornea caricata positivamente e la retina negativamente: il movimento degli occhi genera un segnale elettrico dovuto allo spostamento di questo dipolo costituito dal sistema cornea-retina; quando si sbattono le palpebre, queste calano sulla cornea carica positivamente, permettendo alla carica di fluire lungo la fronte e di influenzare così la registrazione dell'elettroencefalografo [32]. Dal momento che l'intensità di questi artefatti rispetto ai segnali dell'elettroencefalografia è rilevante, si usa spesso eseguire, insieme all'elettroencefalografia, anche un'elettrooculografia per mezzo di due o più elettrodi posizionati nei pressi dell'occhio in configurazione bipolare; il montaggio più semplice e tipico è detto EOG verticale (vEOG) e richiede soltanto un elettrodo al di sopra dell'occhio e un elettrodo al di sotto dell'occhio; altri possibili montaggi comprendono l'EOG orizzontale (hEOG) e l'EOG radiale (rEOG) [32].

Per quanto riguarda l'attività muscolare, il suo contributo al rumore risulta più limitato, soprattutto all'interno di un laboratorio dove il paziente ha la possibilità di rimanere pressochè immobile. Gli artefatti più influenti sono generati soprattutto dai muscoli facciali, in particolare i muscoli frontali e temporali. I muscoli scheletrici del volto sono infatti in grado di generare segnali tra 0 e 200 Hz, registrabili, in maniera più o meno intensa a seconda dell'origine, da tutti gli elettrodi sullo scalpo; in particolare i muscoli frontali presentano un picco intorno a 30 Hz, alti valori tra 45 e 60 Hz con una progressiva riduzione dopo i 60 Hz, mentre i muscoli temporali generano picchi a 20 Hz e la loro ampiezza maggiore è tra 35 e 80 Hz [27]; tali segnali, dunque, se non limitati od opportunamente filtrati possono nascondere anche i segnali generati nella corteccia sensomotoria e limitare il funzionamento di un'interfaccia cervello-computer. È dimostrato, inoltre, che l'attività muscolare registrata per mezzo di EMG è più accentuata soprattutto nel corso delle prime sessioni di addestramento e in quei soggetti che non sviluppano un controllo completo di un'interfaccia cervello-computer [49]; in questi casi l'attività muscolare può condizionare negativamente l'utilizzo di un'interfaccia cervello-computer; tuttavia

con il progredire dell'addestramento è possibile che il soggetto perfezioni il suo controllo sull'interfaccia cervello-computer e riduca il rumore dovuto all'attività muscolare.

5.2.2 Normalizzazione

Il segnale registrato dall'elettroencefalografo è non-stazionario: questo significa che è possibile che vi siano rilevanti variazioni spontanee nell'ampiezza del segnale nel corso di sessioni condotte in giorni differenti o persino in sessioni condotte durante la stessa giornata. Per limitare questo effetto è possibile utilizzare un filtro a media mobile che usando i dati precedenti cerchi di prevedere la media e la deviazione standard delle sessioni successive. Usando questi dati è quindi possibile calcolare il nuovo valore del segnale normalizzato [25]:

$$y(\tau) = \frac{x(\tau) - \mu}{\sigma},$$

dove:

$$\begin{aligned} y(\tau) &= \textit{segnale normalizzato al tempo } \tau, \\ x(\tau) &= \textit{segnale non normalizzato al tempo } \tau, \\ \mu &= \textit{media del segnale calcolata al tempo } \tau - 1, \\ \sigma &= \textit{deviazione standard del segnale calcolata al tempo } \tau - 1. \end{aligned}$$

5.2.3 Filtraggio spaziale

Il filtraggio spaziale, così come il filtraggio temporale, il signal averaging o il single-trial recognition, è una delle principali tecniche di elaborazione del segnale acquisito da un'elettroencefalografo [88]. Valutando il segnale acquisito da ciascun canale in combinazione con il segnale acquisito da altri canali, il filtraggio spaziale permette di ottenere una misura del segnale registrato meno disturbata da artefatti. Inoltre, riducendo il rumore o rafforzando il segnale, il filtraggio spaziale permette di migliorare il rapporto segnale-rumore, ovvero di limitare l'impatto del rumore sul segnale che si intende invece monitorare [50].

A seconda del numero e della posizione degli elettrodi utilizzati per il filtraggio, si distinguono diversi tipi di filtri spaziali. Nel campo delle interfacce cervello-computer i più utilizzati sono il common average reference, lo small Laplacian filter e il large Laplacian filter. L'applicazione di un filtro piuttosto che un altro, o la decisione di non utilizzare alcun filtro, può influenzare sensibilmente il segnale registrato dall'elettroencefalografo.

Common Average Reference

Il common average reference (CAR) valuta il segnale registrato da ciascun elettrodo sottraendo la media del segnale registrato da tutti gli altri elettrodi (vedi

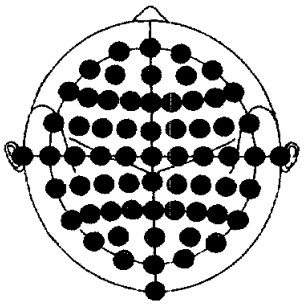

Figura 5.2: Per eseguire un filtraggio spaziale common average reference sono utilizzati tutti gli elettrodi. [47].

Figura 5.2). Per poter applicare il common average reference, la condizione ideale sarebbe quella di avere una copertura uniforme dello scalpo con gli elettrodi. Il common average reference opera come un filtro spaziale passa-alto: esso può rilevare quelle componenti che sono registrate dalla maggioranza degli elettrodi e ridurne l'influenza sul segnale filtrato; come effetto collaterale, però, se una componente è registrata dalla maggioranza degli elettrodi, ma risulta minima o assente nel segnale dell'elettrodo di interesse, è possibile che essa compaia come "potenziale fantasma" in seguito al filtraggio [47]. Matematicamente, il segnale filtrato dal common average reference può essere calcolato come:

$$V_i^{CAR} = V_i - \frac{1}{n}\sum_{j=1}^{n} V_j,$$

dove:

$$\begin{aligned} V_i &= segnale\, registrato\, dall'i-esimo\, elettrodo, \\ V_i^{CAR} &= segnale\, dall'i-esimo\, elettrodo\, filtrato\, con\, CAR, \\ n &= numero\, di\, elettrodi\, montati. \end{aligned}$$

Small Laplacian Filter

Lo small Laplacian filter valuta il segnale registrato da ciascun elettrodo combinandolo con il segnale registrato dai quattro elettrodi adiacenti che circondano l'elettrodo di interesse (vedi Figura 5.3). Lo small Laplacian filter può essere considerato come la derivata seconda della distribuzione spaziale di voltaggio, che permette di evidenziare l'attività di sorgenti radiali al di sotto del punto di registrazione sullo scalpo [88]. Anche lo small Laplacian filter opera come un filtro spaziale passa-alto che permette dunque di mettere in rilievo le attività

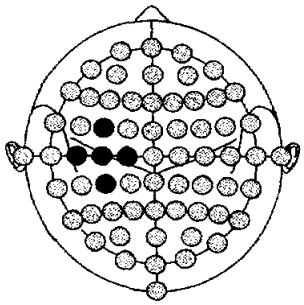

Figura 5.3: In nero, gli elettrodi utilizzati per eseguire un filtraggio spaziale small Laplacian [47].

del cervello maggiormente localizzate a scapito delle attività più diffuse. Formalmente, il segnale che si ottiente attraverso uno small Laplacian filter si può esprimere come [33]:

$$V_i^{SL} = V_i - \frac{1}{4}\sum_{j \in S} V_j,$$

oppure, nel caso più generale, come [47]:

$$V_i^{SL} = V_i - \sum_{j \in S_i} g_{ij} V_{ij},$$

dove:

$$g_{ij} = \frac{1}{d_{ij}} \sum_{j \in S_i} \frac{1}{d_{ij}},$$

dove:

$$\begin{aligned} V_i &= segnale\, registrato\, dall'i-esimo\, elettrodo, \\ V_i^{SL} &= segnale\, dall'i-esimo\, elettrodo\, filtrato\, con\, small\, Laplacian\, filter, \\ S_i &= insieme\, degli\, elettrodi\, adiacenti\, all'i-esimo\, elettrodo, \\ d_{ij} &= distanza\, tra\, l'i-esimo\, e\, il\, j-esimo\, elettrodo. \end{aligned}$$

Large Laplacian Filter

Il large Laplacian filter valuta il segnale registrato da ciascun elettrodo combinandolo con il segnale registrato dai quattro elettrodi prossimi agli elettrodi adiacenti che circondano l'elettrodo di interesse (vedi Figura 5.4). Anche il large Laplacian filter si può considerare come la derivata seconda della distribuzione spaziale di voltaggio, che opera come un filtro spaziale passa-alto e permette

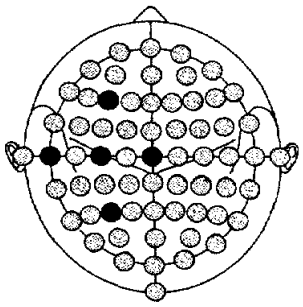

Figura 5.4: In nero, gli elettrodi utilizzati per eseguire un filtraggio spaziale large Laplacian [47].

di evidenziare le attività del cervello maggiormente localizzate a scapito delle attività più diffuse. Si noti tuttavia che con l'aumentare della distanza dall'elettrodo di interesse, il large Laplacian filter diventa meno sensibile alle sorgenti con frequenze spaziali più alte, ovvero le sorgenti più localizzate, e più sensibile alle sorgenti con frequenze spaziali più basse, ovvero le sorgenti più distribuite [88]. Di conseguenza, mentre lo small Laplacian filter è in grado di offrire i risultati migliori in presenza di un segnale altamente localizzato e con una configurazione degli elettrodi precisa e stabile, il large Laplacian filter dovrebbe risultare più efficiente in presenza di segnali meno localizzati e con configurazioni meno precise [47].

La formulazione matematica che esprime il segnale filtrato attraverso un large Laplacian filter è identica a quella dello small Laplacian filter; l'unica differenza è data dal valore che sarà assunto da S e da d_{ij}:

$$V_i^{LL} = V_i - \sum_{j \in S_i} g_{ij} V_{ij},$$

dove:

$$g_{ij} = \frac{1}{d_{ij}} \sum_{j \in S_i} \frac{1}{d_{ij}},$$

dove:

$$\begin{aligned} V_i &= segnale\, registrato\, dall'i-esimo\, elettrodo \\ V_i^{LL} &= segnale\, dall'i-esimo\, elettrodo\, filtrato\, con\, large\, Laplacian\, filter \\ S_i &= insieme\, degli\, elettrodi\, adiacenti\, all'i-esimo\, elettrodo \\ d_{ij} &= distanza\, tra\, l'i-esimo\, e\, il\, j-esimo\, elettrodo \end{aligned}$$

Confronto tra i metodi di filtraggio spaziale

Il common average reference, lo small Laplacian filter e il large Laplacian filter rappresentano ottimi filtri spaziali, in grado di restituire un segnale filtrato con un rapporto segnale-rumore migliore di quello ottenuto da una semplice registrazione monopolare o bipolare senza alcun filtro. Questi filtri, infatti, comportandosi come filtri spaziali passa-alto sono in grado di evidenziare l'attività delle sorgenti locali e limitare il contributo di sorgenti distribuite e lontane, che generalmente sono costituite da movimenti muscolari, oculari o dal ritmo visuale alfa. Nel caso delle interfacce cervello-computer sensomotorie, basate su ritmo beta o mu, i risultati di gran lunga migliori si sono ottenuti con il common average reference o il large Laplacian filter [47]; ciò è dovuto principalmente alle frequenze spaziali filtrate, alla dimensione topografica e alla locazione del segnale di interesse. Gli studi sulla corteccia sensomotoria e i dati sperimentali ottenuti da D. J. McFarland [47] confermano che i risultati migliori offerti dal common average reference e dal large Laplacian filter sono dovuti alla loro capacità di evidenziare il segnale di interesse con maggiore fedeltà; inoltre, a differenza dello small Laplacian filter, questi due filtri hanno anche una ridotta sensibilità all'inevitabile variabilità nel posizionamento degli elettrodi ($\pm 0.5\,cm$) tra una sessione e l'altra.

5.2.4 Selezione dei canali e delle frequenze

Per mezzo dell'elettroencefalografia e di un opportuno filtraggio è dunque possibile ottenere i segnali necessari per il controllo di un'interfaccia cervello-computer. Tuttavia, non tutti i segnali così ottenuti sono utili per il fine prestabilito: generalmente è sufficiente il segnale registrato da un sottoinsieme limitato di canali e in una determinata banda di frequenze.

Per determinare i canali e le bande da utilizzare è necessario eseguire delle sessioni iniziali durante le quali monitorare l'attività su tutti gli elettrodi e su tutte le frequenze per poi determinare, attraverso opportuni criteri di valutazione, i canali e le frequenze che l'utente dimostra di poter controllare maggiormente. La selezione di canali e frequenze può avvenire manualmente da parte di un operatore [57] oppure può essere computerizzata in modo da selezionare automaticamente i canali e le frequenze che offrono i risultati migliori. Una volta trovati i canali di interesse è inoltre possibile registrare il segnale prodotto dagli elettrodi di interesse a una frequenza maggiore e campionare tutti gli altri segnali, per future analisi offline, a una frequenza inferiore [54, 50].

Nel campo delle interfacce cervello-computer sensomotorie i canali selezionati più frequentemente sono, ovviamente, quelli corrispondenti agli elettrodi al di sopra della corteccia sensomotoria, in particolare C3, C4, CP3 o CP4 [90, 49]. Le frequenze più comuni sono quelle associate al movimento e all'immaginazione del movimento e prodotte proprio in corrispondenza della corteccia sensomotoria, ovvero le frequenze mu e beta [25, 52].

5.2.5 Analisi in frequenza del segnale

Durante gli esperimenti condotti in laboratorio, l'analisi del segnale avviene nel dominio delle frequenze. Questo significa che i dati nel dominio del tempo devono essere convertiti nel dominio delle frequenze; inoltre, dal momento che il feedback di un'applicazione di un'interfaccia cervello-computer deve essere in tempo reale (e.g., muovere una carrozzella non appena l'utente manifesta l'intenzione di compiere una tale azione), è fondamentale che i metodi e gli algoritmi di trasformazione del segnale siano veloci, non computazionalmente pesanti e restituiscano un risultato valido anche con un input di dimensioni limitate.

Diversi algoritmi sono stati implementati per convertire i dati ottenuti nel dominio del tempo in dati nel dominio delle frequenze, tra cui la *trasformata discreta di Fourier veloce* (discrete fast Fourier transform, DFFT) [89] e la *stima spettrale autoregressiva* (AR spectral estimation) [38].

Nel caso di analisi offline, ovvero nel caso di analisi a posteriori dell'acquisizione, senza necessità di alcun feedback e senza requisiti sul tempo di risposta, è anche possibile utilizzare la *trasformata di Fourier breve* (short-time Fourier transform, STFT) [60] o la *rappresentazione quadratica tempo-frequenze* (quadratic time-frequency representation) [60].

Nel seguito, si presentano sommariamente i concetti di *trasformata di Fourier* (Fourier transform) e di *trasformata discreta di Fourier* (discrete Fourier transform), per poi introdurre gli algoritmi usati nel caso di applicazioni di interfacce cervello-computer.

Trasformata di Fourier

La trasformata di Fourier è un'operazione che permette di trasformare un segnale reale nel dominio del tempo in un segnale complesso nel dominio delle frequenze [70]. Matematicamente, essa è definita come:

$$X(f) = \int_{-\infty}^{+\infty} x(t) \exp^{-j2\pi ft}\, dt.$$

Dato un segnale $x(t)$, la trasformata di Fourier permette dunque di rappresentare il segnale come una serie infinita di esponenziali e di scomporre il segnale in una serie infinita di sinusoidi; in questo modo può essere messo in evidenza il contributo di ciascuna frequenza al segnale finale [65].

Trasformata discreta di Fourier

Dal momento che non è possibile calcolare una somma infinita di termini e poichè quasi tutta la densità spettrale di un segnale tempo-continuo è contenuta in una banda limitata, la trasfromata di Fourier di un segnale può essere più agevolmente calcolata attraverso la trasformata discreta di Fourier:

$$X_k = \sum_{n=0}^{N-1} x_n \exp^{\frac{-j2\pi kt}{N}},$$

dove n indica i campioni del segnale tempo continuo disponibile. Secondo il teorema del campionamento, affinché il segnale calcolato in frequenza non contenga errori di equivocazione (aliasing) in frequenza, è necessario che la frequenza di campionamento sia almeno maggiore del doppio della banda del segnale da trasformare [70].

Trasformata discreta di Fourier veloce

Data la definizione di trasformata discreta di Fourier di cui sopra, la trasformata discreta di Fourier veloce è un algoritmo in grado di implementare la trasformata discreta di Fourier e ridurne la complessità computazionale da $2N^2$ a $2N \log_2(N)$. Esistono diversi algoritmi in grado di implementare una trasformata discreta di Fourier veloce, tra cui l'algoritmo di Cooley-Tukey (basato sulla decimazione in tempo) o l'algoritmo di Sande-Tukey (basato sulla decimazione in frequenza) [84].

Stima spettrale autoregressiva

La stima spettrale autoregressiva è una delle tecniche più diffuse per determinare lo spettro di un segnale durante l'utilizzo di un'applicazione di un'interfaccia cervello-computer; questa tecnica ha infatti una risoluzione superiore rispetto a quella che potrebbe essere ottenuta utilizzando un metodo basato sulla trasformata di Fourier [38].

Per la stima dei coefficienti del modello autoregressivo, uno degli algoritmi più noti è l'algoritmo di Burg, basato sull'applicazione del metodo della massimizzazione dell'entropia (Maximum Entropy Method, MEM) all'analisi spettrale.

Per poter comprendere con precisione come funzioni la stima spettrale è necessario introdurre alcuni concetti e teoremi.

Entropia Uno dei concetti fondamentali per la teoria dell'informazione e per il metodo della massimizzazione dell'entropia è il concetto di entropia [18, 79].

Sia data una variabile aleatoria X con funzione di probabilità di massa o densità $p_X(x)$ tale che:

$$P(X = n) = p_X(n),$$
$$\textstyle\sum_n p_X(n) = 1.$$

La conoscenza della densità $p_X(x)$ della variabile aleatoria X può fornire informazione riguardo al sistema modellato dalla variabile aleatoria X. Se a priori tutti i valori di $p_X(x)$ fossero equivalenti, questo significherebbe che nessuna particolare informazione riguardo al sistema sarebbe disponibile (e.g., se la variabile aleatoria X fosse l'esito del lancio di un dado a sei facce equilibrato, allora tutti

i risultati sarebbero equamente probabili e nessuna informazione sul sistema sarebbe disponibile); d'altra parte, se a priori fosse possibile determinare il valore di un determinato $p_X(x)$, questo significherebbe che una certa quantità di informazione riguardo al sistema sarebbe disponibile (e.g., se la variabile aleatoria X fosse l'esito del lancio di un dado a sei facce truccato in modo da far apparire la faccia n, allora la probabilità di ottenere n sarebbe diversa e maggiore della probabilità di ottenere le altre facce; si avrebbe, cioè, dell'informazione riguardo al sistema).

In definitiva, esiste un legame tra probabilità di avvenimento di un evento e informazione; tale relazione è esprimibile matematicamente come:

$$I(n) = k \ln\left(\frac{1}{p_X(n)}\right), \tag{5.1}$$

o:

$$I(n) = \log_2\left(\frac{1}{p_X(n)}\right), \tag{5.2}$$

ovvero con k uguale a 1 quando il logaritmo è in base 2. Se dunque la formula (5.1) o (5.2) esprime l'informazione riguardo all'evento $X = n$, è possibile conseguentemente definire l'informazione totale di un sistema. Infatti, si consideri un tempo T sufficientemente grande per cui, per ogni evento m il numero di ripetizioni di $X = m$ è uguale a $p_X(m)T$; allora l'informazione totale si può esprimere come:

$$I_{tot} = k\left(p_X(1)T\ln\frac{1}{p_X(1)} + p_X(2)T\ln\frac{1}{p_X(2)} + \ldots + p_X(m)T\ln\frac{1}{p_X(m)} + \ldots\right). \tag{5.3}$$

La quantità di informazione totale per unità di tempo è definita entropia o entropia di Shannon:

$$H = \frac{I_{tot}}{T} = \frac{k}{T}\sum_i p_X(i)T\ln\frac{1}{p_X(i)} = -k\sum_i p_X(i)\ln p_X(i). \tag{5.4}$$

Concettualmente, dunque, l'entropia è la misura dell'incertezza media descritta da una variabile aleatoria: un'entropia pari a 0 indica un sistema perfettamente determinato privo di incertezze (valore che si ottiene solo se tutti i valori di $p_X(x)$ sono uguali a 0 eccetto un valore $p_X(k)$ che sia uguale a 1); un'entropia maggiore di 0 indica la presenza di incertezza all'interno del sistema. Ancora, l'entropia può essere intesa come misura del disordine di un sistema, come misura della nostra ignoranza rispetto alla reale struttura di un sistema [79] oppure come misura della sorpresa di fronte al risultato prodotto dal sistema [76]. Inoltre, usando il logaritmo in base 2 e ottenendo:

$$H = -\sum_i p_X(i)\log_2 p_X(i), \tag{5.5}$$

l'entropia fornisce anche il numero medio di bit necessari per descrivere una variabile aleatoria [18].

Entropia relativa Mentre l'entropia calcola l'informazione media necessaria per descrivere una variabile aleatoria, l'entropia relativa o distanza di Kullback Leibler [18] è una misura della distanza tra due distribuzioni.

Data una variabile aleatoria X e due distribuzioni $p_X(x)$ e $q_X(x)$, l'entropia relativa è definita come:

$$D(p\|q) = \sum_{x\in\Re} p_X(x) \log_2 \frac{p_X(x)}{q_X(x)}. \tag{5.6}$$

Come detto, l'entropia relativa misura la distanza tra due distribuzioni: ovvero, data una variabile aleatoria con distribuzione $p_X(x)$, l'entropia relativa quantifica l'inefficienza che deriva utilizzando la distribuzione $q_X(x)$ quando invece la vera distribuzione è $p_X(x)$; al lato pratico, usando la vera distribuzione $p_X(x)$, sarebbe necessario usare $h(p)$ bit in media per descrivere la variabile aleatoria, mentre usando la distribuzione $q_X(x)$ sarebbero necessari $h(p) + D(p\|q)$ bit in media per descrivere la stessa variabile aleatoria [18].

Teorema della distribuzione della massima entropia Sia data una variabile aleatoria continua X definita su uno spazio campionario S con funzione di densità $f_X(x)$, tale che la sua entropia sia $h(f)$. Si cerchi ora di massimizzare $h(f)$ rispettando i seguenti vincoli:

$$f_X(x) \geq 0 \ \ \forall x \in \Re, \tag{5.7}$$

$$\int_S f_X(x)dx = 1, \tag{5.8}$$

$$\int_s f_X(x) r_i(x) dx = \alpha_i \ \ \forall 1 \leq i \leq m, \tag{5.9}$$

dove le formule (5.7) e (5.8) sono vincoli impliciti nella definizione stessa di funzione di densità, mentre la formula (5.9) richiede di soddisfare precisi vincoli sui momenti [18].

Si prenda allora una $g(x)$ che soddisfi (5.7), (5.8) e (5.9). Si prenda anche una $f*(x)$, della forma[1]:

$$f(x) = \exp^{\lambda_0 - 1 + \sum_{i=1}^{m} \lambda_i r_i(x)}. \tag{5.10}$$

È inoltre legittimo assumere, per il teorema della diseguaglianza dell'entropia (vedi Appendice) che:

$$D(g\|f*) \geq 0, \tag{5.11}$$

[1] Questa formula può essere derivata analiticamente cercando di soddisfare i vincoli (5.7), (5.8) e (5.9). Per questo, vedi [18]. In questo contesto la si prende per data, e si dimostrerà che è la formula in grado di soddisfare i vincoli imposti e massimizzare l'entropia $h(f)$.

e quindi:

$$-h(g) + h(f*) \geq 0, \tag{5.12}$$

ovvero:

$$h(g) \leq h(f*). \tag{5.13}$$

Il teorema della distribuzione della massima entropia [18] afferma che data una $f*(x)$ della forma (5.10), dove $\lambda_1, \lambda_2 \ldots \lambda_m$ sono scelti per soddisfare i vincoli (5.7), (5.8) e (5.9), allora $f*$ è l'unica a massimizzare $h(f)$ tra tutte le funzioni di densità f che soddisfano i vincoli (5.7), (5.8) e (5.9).

Stima spettrale Si consideri un processo casuale X_i stazionario a media nulla, ovvero tale che la media sia costante a ogni istante di tempo e pari a 0 e l'autocorrelazione del processo non dipenda dai singoli istanti temporali considerati, ma solo dal loro intervallo:

$$\begin{gathered} E[X_i] = \mu_X = 0, \\ \forall x_1 \wedge x_2,\ x_2 - x_1 = \tau \quad R_X(x_1, x_2) = R_X(\tau) = E[x_i x_{i+\tau}]. \end{gathered} \tag{5.14}$$

La trasformata di Fourier dell'autocorrelazione è la densità spettrale di potenza:

$$S_X(\lambda) = \int_{-\infty}^{+\infty} R_X(\tau) \exp^{-j2\pi\lambda\tau}\, d\tau. \tag{5.15}$$

Eseguire una stima spettrale significa dunque determinare la struttura di un processo a partire dal valore ricavato da campioni dello stesso processo [18]. La densità spettrale indica infatti come viene ripartita la potenza del processo nelle varie bande spettrali che compongono il processo stesso [70].

Metodo del periodogramma Uno dei metodi più semplici e intuitivi [18] per ottenere l'autocorrelazione $R_X(\tau)$ e da questa la densità spettrale di energia $S_X(\lambda)$ attraverso la formula (5.15), consiste nello stimare i valori dell'autocorrelazione a partire dai valori ottenuti da un campione di lunghezza n:

$$\hat{R}_X(\tau) = \frac{1}{n-\tau} \sum_{i=1}^{n-\tau} X_i X_{i+\tau}. \tag{5.16}$$

Questo metodo, però, per grandi valori di n, non converge correttamente allo spettro reale del processo osservato; questo risultato è dovuto al fatto che i valori dell'autocorrelazione calcolati con la formula (5.16) hanno una buona accuratezza per bassi valori di τ, quando molti campioni rientrano nella sommatoria, ma un'accuratezza molto limitata per alti valori di τ, quando invece ben pochi campioni sono usati nella stima. Alcune soluzioni a questo metodo sono state proposte per migliorarne l'accuratezza; è possibile, ad esempio,

computare l'autocorrelazione usando solo valori bassi di τ e porre tutti gli altri valori a 0, anche se questo introdurrebbe degli artefatti a causa della repentina transizione a zero; per contrastare quest'ultimo effetto, si potrebbero usare delle finestre per rendere graduale la transizione, ma l'uso di finestre potrebbe ridurre la risoluzione spaziale e generare stime spettrali negative.

Teorema della massima entropia di Burg Il metodo di Burg [18, 79] si propone come una possibile soluzione ai limiti individuati nel metodo del periodogramma; anziché porre i valori alti di τ a 0, Burg propone di scegliere un valore basato sulle minime assunzioni possibili riguardo ai dati, ovvero di scegliere quei valori che massimizzerebbero l'entropia del processo.
Il teorema della massima entropia di Burg [18] afferma che l'entropia massima di un processo casuale X_i che soddisfi il vincolo:

$$E\left[X_i X_{i+k}\right] = \alpha_k \;\; 0 \leq k \leq p \;\; \forall i, \tag{5.17}$$

è il processo di Gauss-Markov di ordine p della forma:

$$X_i = -\sum_{k=1}^{p} a_k X_{i-k} + Z_i, \tag{5.18}$$

dove:

$$Z \sim N\left(0, \sigma^2\right), \tag{5.19}$$

e $a_1, a_2 \ldots a_p$ e σ^2 sono scelti per soddisfare il vincolo espresso in (5.17).

Per determinare il valore di $a_1, a_2 \ldots a_p$ e σ^2, si consideri di nuovo l'espressione in (5.18). Valutandola in 0, e ricordando che $R(k) = R(-k)$, si ha:

$$R(0) = -\sum_{k=1}^{p} a_k R(-k) + \sigma^2. \tag{5.20}$$

Usando poi l'operatore di valore atteso e moltiplicando per X_{i-1} si ottiene:

$$R(l) = -\sum_{k=1}^{p} a_k R(l-k) \;\; l = 1, 2 \ldots \tag{5.21}$$

Quest'ultima formula è un'espressione delle equazioni di Yule-Walker. Si hanno così $p+1$ equazioni in $p+1$ incognite ($a_1, a_2 \ldots a_p$ e σ^2) nella forma [9]:

$$\begin{bmatrix} a_1 \\ a_2 \\ \cdots \\ a_p \end{bmatrix} = \begin{bmatrix} R_0 & R_1 & \cdots & R_{p-1} \\ R_1 & R_0 & \cdots & R_{p-2} \\ \cdots & \cdots & \cdots & \cdots \\ R_{p-1} & R_{p-2} & \cdots & R_0 \end{bmatrix}^{-1} \begin{bmatrix} R_1 \\ R_2 \\ \cdots \\ R_p \end{bmatrix}, \tag{5.22}$$

da cui ricavare $a_1, a_2 \ldots a_p$, più l'equazione di $R(0)$, definita in (5.17), da cui ottenere σ^2.

Le equazioni di Yule-Walker forniscono un set di equazioni lineari che possono essere rapidamente computate usando algoritmi di risoluzione noti, come l'algoritmo di Durbin-Levinson [9]. Dalle equazioni di Yule-Walker è inoltre possibile stimare il valore dello spettro anche per valori di autocorrelazione maggiori di p (estensioni di Yule-Walker per le autocorrelazioni [18]).

Lo spettro del processo a massima entropia, soggetto ai vincoli:

$$R(0), R(1) \dots R(p),$$

è dunque:

$$S(l) = \frac{\sigma^2}{\mid 1 + \sum_{k=1}^{p} a_k \exp^{-ikl} \mid^2}. \tag{5.23}$$

Date le formule per calcolare lo spettro attraverso le equazioni di Yule-Walker, l'unico elemento ancora da determinare è l'ordine del modello p. Nel contesto delle interfacce cervello-computer questo problema è stato affrontato sperimentalmente dal Wadsworth Center [38]. Una prima ipotesi, subito rivelatasi inappropriata, è stata quella di usare un numero di poli, e quindi un ordine, pari al numero di picchi generati dalla combinazione di onde delta, theta, alfa, beta e gamma. La letteratura scientifica preesistente propone, nell'uso dell'elettroencefalografo, un'ordine pari a 11-12; tuttavia, sebbene sia stato provato che questi valori siano affidabili per una normale elettroencefalografia, essi risultano meno adatti quando per mezzo di una elettroencefalografia si cerca di monitorare un ritmo sensomotorio governato dall'utente. Usando come criterio di valutazione il coefficiente di correlazione e calcolando lo spettro con tutti gli ordini pari tra 2 e 32, il gruppo di ricerca di McFarland ha studiato il valore di r^2 in corrispondenza del ritmo beta, del ritmo mu e della loro combinazione [38]. Mentre per il ritmo beta si possono ottenere r^2 anche con ordini molto bassi come 2, in generale i tre ritmi ottengono i risultati migliori con un ordine maggiore di 10; in particolare il ritmo mu e la combinazione di mu e beta hanno un valore significativo quando l'ordine è maggiore di 10 e hanno un comportamento di tipo asintotico quando l'ordine è maggiore di 26 [38].

Il valore ottimale dell'ordine della stima autoregressiva può variare in maniera limitata tra gli utenti; generalmente il valore dell'ordine della stima autoregressiva è scelto tra 16 [38, 39] e 25 [74].

5.2.6 Calcolo delle feature

Una volta ottenuto un segnale utile, è necessario costruire o estrarre da questo delle feature utili a identificare la volontà dell'utente dell'interfaccia cervello-computer. Questo procedimento prende il nome di *calcolo delle feature* (o *feature extraction* o *feature construction* o *feature generation* o *feature choice*).

Con il termine *feature* si intende il valore simbolico di un attributo o il valore numerico di un variabile di input che possa essere utilizzato per discriminare un segnale (e.g., in un'interfaccia cervello-computer, la volontà dell'utente). Il fine di questo processo è dunque quello di generare un insieme di feature utili a

partire da un insieme di segnali grezzi [31]; si cercano cioè quelle feature che siano invarianti rispetto a trasformazioni irrilevanti degli input e permettano così di evidenziare le variazioni significative dei segnali di input [22].

La finalità principale del procedimento di calcolo delle feature è quella, dunque, di ridurre la quantità di informazione acquisita; infatti, sebbene teoricamente sia possibile operare utilizzando i segnali acquisiti senza nessuna operazione di feature extraction preventiva, questo risulta all'atto pratico molto oneroso, se non addirittura impossibile. Utilizzando conoscenze a priori, se disponibili, e algoritmi di feature extraction è possibile dunque calcolare un insieme limitato di descrittori che sintetizzino tutta l'informazione contenuta nei segnali originali.

Le operazioni necessarie a generare un insieme di feature, partendo da un segnale grezzo, possono essere determinate dall'analisi dei segnali da parte di operatori umani esperti oppure possono essere eseguite in maniera computerizzata per mezzo di algoritmi di feature extraction.

Spesso il confine tra operazioni di feature extraction e signal processing è veramente labile; in molti casi operazioni di analisi dei segnali vengono considerate parte del processo di feature extraction [78]. Inoltre, anche il legame tra la feature extraction e la classificazione è molto stretto: la feature extraction dovrebbe generare un insieme di feature in funzione del classificatore, ovvero trovare quell'insieme di feature che consenta a un classificatore di ottenere le migliori prestazioni possibili [22].

Dato un segnale grezzo esistono diverse operazioni che possono essere eseguite per creare un insieme di feature [31]:

Operazioni di signal processing alcune operazioni di signal processing elencate nei paragrafi precedenti possono essere eseguite o rieseguite come parte del processo di feature extraction, tra cui standardizzazione, normalizzazione o incremento del rapporto segnale-rumore;

Operazioni di discretizzazione del segnale nel caso di un segnale grezzo continuo è possibile generare feature campionando il segnale continuo nel tempo e/o nelle ampiezze;

Operazioni su feature locali nel caso di dati strutturati è possibile utilizzare tecniche specifiche per l'estrazione di feature locali, che, basandosi sulla conoscenza del problema e del dominio, consentono di generare un insieme di feature locali utili;

Operazioni di eliminazione degli outlier nel caso vi siano alcuni dati molto lontani dalla media è possibile rimuoverli; la soglia in base alla quale considerare un dato outlier è generalmente calcolata in base alla varianza dei dati; i dati outlier sono spesso dovuti al rumore e possono avere un effetto negativo sulla successiva classificazione [77].

Una buona procedura di feature extraction è in grado di generare dai segnali originari un numero ragionevole di feature utili senza perdere informazione; esiste comunque un trade-off tra numero di feature e perdita di informazione: da una parte è necessario avere un numero di feature non troppo elevato per

evitare problemi connessi all'eccessiva quantità di dati; dall'altra è importante non trascurare o eliminare quelle feature che potrebbero contenere informazione utile.

Al termine dell'operazione di feature construction è possibile organizzare i dati raccolti in una matrice $n \times m$, dove le righe denotano gli istanti temporali o i singoli campioni acquisiti (detti *esempi*, *instances* o *osservazioni*) e le colonne i valori osservati (detti *feature*, *values* o *attributi*); un generico valore (i, j) di questa matrice rappresenta quindi il valore che la feature j ha assunto al tempo i o nella misurazione i. Se si dispone poi dei risultati attesi per ogni istante temporale, è conveniente organizzare questi ultimi in un vettore $n \times 1$, dove le righe denotano sempre gli istanti temporali e la colonna contiene il risultato atteso (detto *target*, *label* o *etichetta*); un generico valore i di questo vettore rappresenta dunque l'output atteso al tempo i o nella misurazione i. Se si considera ad esempio un'applicazione di un'interfaccia cervello-computer sensomotoria in cui al soggetto è richiesto di selezionare un determinato target, allora si potrà costruire una matrice delle feature come la seguente:

$$\begin{array}{c} \\ t_1 \\ t_2 \\ \dots \\ t_N \end{array} \quad \begin{array}{c} Feature \\ \left[\begin{array}{cccc} x_{11} & x_{12} & \dots & x_{1M} \\ x_{21} & x_{22} & \dots & x_{2M} \\ \dots & \dots & \dots & \dots \\ x_{N1} & x_{N2} & \dots & x_{NM} \end{array}\right] \end{array} \quad \begin{array}{c} Valore\,atteso \\ \left[\begin{array}{c} trg_1 \\ trg_2 \\ \dots \\ trg_N \end{array}\right] \end{array},$$

dove gli N istanti temporali sono gli istanti in cui viene campionato il segnale dell'elettroencefalografo (e.g., il segnale registrato ogni secondo), le M feature sono le feature selezionate per discriminare il segnale dell'utente (e.g., la frequenza mu dal canale C3 o la frequenza mu dal canale C4) e i target sono i risultati attesi (e.g., target nella parte superiore dello schermo o target nella parte inferiore dello schermo). Per mezzo di questa matrice è quindi possibile analizzare il comportamento delle differenti feature in diversi istanti temporali in relazione al target presentato; dalla matrice delle feature si potrà determinare il potere predittivo delle feature, ovvero se il valore assunto dalle singole feature o combinazioni di esse è indicativo del target e può essere utilizzato proficuamente per la successiva classificazione.

Calcolo manuale delle feature

Un primo possibile metodo per la generazione di feature è il calcolo manuale delle feature. Il calcolo manuale delle feature si basa sull'identificazione da parte di un operatore di un insieme di feature utili. La scelta di tali feature può essere basata sull'analisi manuale dei dati disponibili, sullo studio di casi precedenti presenti in letteratura, sulla conoscenza del dominio da parte di esperti del settore o su esperimenti di tipo *trial and error*. Tale procedimento per il calcolo delle feature è spesso equiparato a un'arte piuttosto che a una scienza [43]. Lo svantaggio principale dei metodi di calcolo manuale delle feature è che sono generalmente lunghi, in quanto la quantità di dati da studiare è consistente e la capacità di analisi di un essere umano è inevitabilmente limitata. Inoltre, specialmente nel

caso dell'analisi dei dati di interfacce cervello-computer in cui la variabilità tra i soggetti è molto alta, i risultati ottenuti hanno un potere di generalizzazione molto basso; è infatti comune che un insieme di feature che garantisce ottime prestazioni per un utente, fornisca soltanto delle prestazioni mediocri per un altro utente. D'altra parte, sebbene i risultati ottenuti attraverso metodi di calcolo manuale delle feature dipendano inevitabilmente dall'esperienza e dalla competenza di chi le studia, le performance che si ottengono con questi metodi sono generalmente alte in quanto le feature ottenute sono state appositamente generate per il caso specifico.

Calcolo automatico delle feature

Alternativamente al metodo di generazione manuale delle feature è possibile ricorrere a metodi di calcolo automatici per la generazione delle feature. Tali metodi puntano a generare, partendo dai dati grezzi, un insieme di feature che abbiano la *proprietà* di sintetizzare e comprimere quanta più informazione possibile [77]. La generazione automatica delle feature può essere eseguita sfruttando la conoscenza a priori del problema, utilizzando trasformazioni lineari dei dati, sfruttando proprietà locali dei dati [77] o semplicemente tentando di selezionare quei dati che permettano di massimizzare i futuri risultati della classificazione. Generalmente, potendo sfruttare il potere computazionale dei moderni calcolatori, i metodi di calcolo automatico delle feature risultano molto più veloci dei metodi manuali. I metodi automatici hanno inoltre maggiore versatilità e possono essere applicati, senza modifiche, per risolvere diversi problemi. Inevitabilmente connesso a questa maggiore versatilità, vi è però il fatto che i sistemi di calcolo automatico delle feature, non essendo progettati per uno specifico problema o per uno specifico set di dati, tendono a raggiungere dei risultati subottimali; ciò è specialmente evidente nel caso delle interfacce cervello-computer, in cui l'alta variabilità dei dati da soggetto a soggetto potrebbe richiedere uno studio accurato di ciascun singolo caso.

5.2.7 Riduzione della dimensionalità

La quantità di dati contenuti nell'insieme di feature generato per mezzo di procedimenti di feature extraction è sicuramente inferiore alla quantità di dati contenuti nei segnali acquisiti, ma spesso ancora troppo elevata per poter essere utilizzata in modo efficiente.

L'eccessiva quantità di dati può essere causa di diversi problemi tra cui:

Curse of dimensionality uno dei problemi più gravi che affligge tutte le applicazioni che utilizzano un'ingente mole di dati è quello del *curse of dimensionality,* ovvero dell'eccessiva dimensionalità dei dati. Una quantità di dati troppo grande può infatti rapidamente diventare ingestibile e rendere impossibile un uso pratico ed efficiente dei dati; spesso, avere troppi dati è praticamente equivalente a non avere nessun dato [41]. Inoltre è noto che esiste un numero limite di dati o feature superato il quale le performance di un sistema non migliorano, ma addirittura peggiorano [7];

Costo computazionale la quantità di dati disponibili determina i requisiti di spazio e di tempo che un sistema deve soddisfare per gestire e analizzare questi dati. Un numero eccessivo di dati può rapidamente portare all'esaurimento della memoria del sistema o a tempi di elaborazione eccessivamente lunghi;

Problemi numerici la sovrabbondanza di dati può essere causa anche di problemi numerici; ad esempio la ridondanza dei dati può generare matrici non invertibili di difficile gestione.

Per risolvere questi problemi è opportuno ricorrere a un ulteriore riduzione della quantità di informazione presente nell'insieme di feature calcolato. Esistono due tecniche per ridurre la dimensionalità delle feature: *feature selection* e *feature projection*.

Feature selection

Una volta generato un insieme di feature utili dai segnali grezzi, può essere opportuno eseguire delle operazioni su questo insieme per ridurre il numero di feature che verranno usate nella successiva classificazione. Un primo modo per diminuire il numero di feature consiste nell'eliminare tutte quelle che risultino prive di potere predittivo. Questo procedimento prende il nome di *feature selection* (o *feature reduction*). Il fine principale della feature selection è quello di scartare tutte le feature che non hanno contenuto informativo e conservare così solo quelle rilevanti per la successiva classificazione. Durante questo processo si punta perciò a eliminare tutte quelle feature che sono irrilevanti (i.e., non influenzano la classificazione), ridondanti (i.e., contengono informazione già contenuta in altre feature) e/o introducono rumore (i.e., influenzano negativamente la qualità della discriminazione) [41].

Riducendo opportunamente il numero delle feature disponibili è possibile ottenere diversi vantaggi: anzitutto si possono avere miglioramenti nella performance della classificazione; in secondo luogo si limita il tempo, lo spazio e le risorse computazionali necessarie per l'elaborazione e l'archiviazione dei dati; in terzo luogo, la selezione di un numero contenuto di feature permette di comprendere e analizzare più facilmente il processo oggetto di studio e i legami tra le feature e i target; infine, riducendo la dimensionalità dei dati è anche possibile rappresentare le feature graficamente in maniera chiara e intellegibile [31].

A seconda della disponibilità dei target, ovvero dei valori attesi, possiamo classificare le tecniche di feature selection in [41]:

Tecniche supervisionate le tecniche supervisionate sfruttano l'informazione offerta dal target per valutare quanto bene le feature discriminino i target;

Tecniche non supervisionate le tecniche non supervisionate sono prive dell'informazione offerta dal target e valutano l'utilità delle feature analizzando la ridondanza, la separabilità delle feature e le proprietà intrinseche delle feature.

Considerando il funzionamento degli algoritmi di feature selection si possono poi distinguere due aspetti comuni a tutti gli algoritmi: il modo in cui è valutata la bontà di una singola feature o di un insieme di feature e il modo in cui una feature o un insieme di feature è scelto prima di essere valutato.

A seconda del metodo utilizzato per valutare la bontà delle feature le tecniche di feature selection si dividono in:

Metodi Filter i metodi filter, spesso chiamati anche metodi ranking, sono algoritmi che ordinano le feature secondo un preciso indice di rilevanza legato a delle proprietà intrinseche delle feature; scelta una soglia, è possibile selezionare tutte quelle feature che soddisfino la condizione posta dalla soglia ed eliminare le rimanenti; esistono diversi indici in base ai quali ordinare le feature, tra cui indici basati su coefficienti di correlazione tra input e output, indici basati su test statistici oppure indici basati sulla teoria dell'informazione [31].
I metodi filter tendono ad avere una buona performance in presenza di un alto numero di feature e di un numero di esempi relativamente contenuto; essi sono in grado di selezionare in maniera rapida un sottoinsieme di feature che, indipendentemente dal successivo classificatore, offrono dei buoni risultati. D'altra parte, i metodi filter tendono tuttavia a generare sottoinsiemi di feature contenenti un numero di feature molto alto [30].

Metodi Wrapper i metodi wrapper sono algoritmi che selezionano le feature basandosi sui risultati di un classificatore prescelto; il classificatore è utilizzato dal wrapper come una black-box: il wrapper seleziona un sottoinsieme di feature e lo passa al classificatore; quest'ultimo, dopo l'addestramento basato sul sottoinsieme di feature ricevute, restituisce un valore indicativo dell'accuratezza raggiunta nella classificazione. Il wrapper seleziona dunque iterativamente diversi sottoinsiemi di feature finchè identifica il sottoinsieme di feature che garantisce le prestazioni migliori [31].
A differenza dei metodi filter, il risultato ottenuto da un metodo wrapper ha quindi meno generalità (in quanto ottimizzato per un particolare classificatore), ma è in grado di ottenere performance migliori con il classificatore scelto; lo svantaggio principale nell'uso di un metodo wrapper è il tempo necessario per valutare tutti i possibili sottoinsiemi di feature che possono essere selezionati [30].

Metodi Embedded I metodi embedded sono algoritmi integrati all'interno di una specifica learning machine; durante il processo di training della learning machine, i metodi embedded selezionano automaticamente il sottoinsieme ottimo di feature per la successiva classificazione; a differenza dei metodi wrapper il classificatore non è utilizzato come una black-box, ma vi è una interazione tra la learning machine e il metodo embedded; esempi di metodi embedded sono *ID3* o *CART* [69].

A seconda del metodo utilizzato per esplorare lo spazio delle feature e selezionarne un sottoinsieme, le tecniche di feature selection si dividono in [30]:

Metodi best first: i metodi best first ordinano le feature secondo un preciso criterio e selezionano quindi le feature dalla migliore alla peggiore. I metodi best first possono essere ulteriormente classificati in [31]:

- *Metodi univariati*: i metodi univariati assumono che le feature siano tra loro indipendenti e determinano l'indice di rilevanza delle singole feature indipendentemente dal valore delle altre feature; a causa delle loro assunzioni i metodi univariati corrono due gravi rischi: il primo è quello di non valutare l'informazione complementare contenuta nelle feature, ovvero eliminare alcune feature che, prese indipendentemente, non hanno alcuna rilevanza, ma considerate nel contesto di altre feature acquisiscono potere discriminante; il secondo è quello di conservare troppe feature ridondanti; esempi di metodi univariati sono il *coefficiente di correlazione di Pearson* o il *Fisher rank*;

- *Metodi multivariati*: i metodi multivariati determinano l'indice di rilevanza delle feature tenendo conto di possibili dipendenze esistenti tra le feature; i metodi multivariati ottengono spesso prestazioni migliori dei metodi univariati dal momento che non fanno assunzioni sull'indipendenza delle feature; un esempio di metodo multivariato è il *metodo Relief* [31].

Metodi esponenziali: i metodi esponenziali ricercano ordinatamente, secondo una precisa euristica, tra tutti i possibili sottoinsiemi di feature. Questi metodi garantiscono che al termine dell'esecuzione la soluzione ottimale sia identificata; tuttavia la loro complessità cresce esponenzialmente con la dimensionalità dei dati, e, così, in presenza di un elevato numero di feature questi metodi risultano impraticabili a causa del loro costo computazionale. Esempi di metodi esponenziali sono la *ricerca esaustiva* e il *branch and bound (B&B)*;

Metodi sequenziali: i metodi sequenziali ricercano un sottoinsieme di feature utili aggiungendo o rimuovendo sequenzialmente feature a partire da un insieme di feature dato. Questi metodi sono concettualmente semplici e computazionalmente rapidi da eseguire; per velocizzare la loro esecuzione i metodi sequenziali tradizionali non sono programmati per riconsiderare a posteriori le scelte fatte: se nel corso dell'esecuzione una feature è stata eliminata, essa non potrà più essere selezionata anche se in un secondo momento potrebbe risultare utile; a causa della loro implementazione, questi metodi potrebbero quindi rimanere bloccati in un minimo locale e selezionare così un sottoinsieme di feature subottimali. Esempi di metodi sequenziali sono il *sequential forward selection (SFS)*, il *sequential backward selection (SBS)* e il *plus-l minus-r selection (l-r)*;

Metodi randomizzati: i metodi randomizzati ricercano un sottoinsieme di feature inserendo dei fattori aleatori all'interno della ricerca che permettano di evitare di bloccarsi in un minimo locale. Questi metodi hanno una complessità contenuta e, se impostati correttamente, sono in grado

di evitare i minimi locali e giungere a una soluzione ottimale; la difficoltà maggiore nell'uso di questi metodi risiede, invece, nella corretta determinazione di quei parametri che regolano l'impatto dei fattori aleatori nell'algoritmo. Esempi di metodi randomizzati sono il *simulated annealing* e gli *algoritmi genetici (GA)*.

Di seguito si analizzano più in dettaglio alcuni dei metodi e degli algoritmi utilizzati per la selezione delle feature durante i nostri esperimenti in laboratorio.

Fisher Rank Il Fisher rank (o *Fisher score* o *Fisher ratio* o *Fisher criterion*) è un metodo di selezione delle feature di tipo *filter*, basato su una politica di esplorazione dello spazio delle feature di tipo *best first univariata*.

Il Fisher rank è per definizione il rapporto tra la varianza interclasse, ovvero la varianza tra classi differenti, e la varianza intraclasse, ovvero la varianza all'interno di una stessa classe.

Più rigorosamente [83], si consideri un problema di classificazione con $C = \{1, 2...c\}$ classi e sia X la matrice delle osservazioni di dimensioni $m \times n$. Per ogni classe $i \in C$ esiste allora un insieme M^i di m_i vettori di osservazioni di dimensione n:

$$M^i = \left\{ \mathbf{x}_1^i, \mathbf{x}_2^i ... \mathbf{x}_{m_i}^i \right\}.$$

Si definisce la probabilità a priori per la classe i nel seguente modo:

$$p_i = \frac{m_i}{\sum_{i=1}^{c} m_i},$$

ovvero come rapporto tra il numero di osservazioni della classe i e il numero di osservazioni totali. Si definisce inoltre la media per la classe i come:

$$\hat{\mu}_i = \frac{1}{m_i} \sum_{j=1}^{m_i} \mathbf{x}_j^i,$$

e la media tra le classi come:

$$\hat{\mu} = \sum_{i=1}^{c} p_i \hat{\mu}_i.$$

Infine, si definisce la matrice di covarianza per la classe i nel seguente modo:

$$\hat{S}_i = \frac{1}{m_i} \sum_{j=1}^{m_i} (\mathbf{x}_j^i - \hat{\mu}_i)(\mathbf{x}_j^i - \hat{\mu}_i)^T.$$

Da queste definizioni è ora possibile calcolare la matrice della varianza interclasse (S_b) e la matrice della varianza intraclasse (S_w):

$$
\begin{aligned}
S_b &= \sum_{i=1}^{c} p_i(\hat{\mu}_i - \hat{\mu})(\hat{\mu}_i - \hat{\mu})^T, \\
S_w &= \sum_{i=1}^{c} p_i \hat{S}_i.
\end{aligned}
$$

Il Fisher rank della feature $k \in \{1, 2...n\}$ sarà quindi dato da:

$$
FR(k) = \frac{S_b(k)}{S_w(k)},
$$

ovvero dal rapporto tra il k-esimo elemento diagonale della matrice S_b e il k-esimo elemento diagonale della matrice S_w.

Dopo aver calcolato il Fisher rank di tutte le feature, è possibile allora ordinare le feature in maniera decrescente e rimuovere quelle con valore di Fisher rank più basso, ovvero quelle feature irrilevanti o rumorose che, prese singolarmente, non darebbero alcun contributo utile alla successiva classificazione [83].

Come evidente dalla spiegazione, il Fisher rank può essere utilizzato sia nel caso di un problema con due classi sia nel caso più generale di un problema multiclasse; inoltre, è possibile dimostrare una stretta relazione tra il Fisher rank e altri metodi *filter best first univariati*, come il coefficiente di correlazione di Pearson $R(i)^2$ o i criteri basati su T-test [31].

Wilcoxon test Il test di Wilcoxon (*Wilcoxon Rank-Sum Test*) o test di omogeneità di Wilcoxon-Mann-Whitney è un test statistico non parametrico utilizzato per stimare se due campioni casuali indipendenti sono omogenei, ovvero se sono regolati dallo stesso modello [24].

Dati due campioni casuali indipendenti

$$
\begin{aligned}
X_1, X_2 ... X_n &\sim A; \\
Y_1, Y_2 ... Y_m &\sim B;
\end{aligned}
$$

si vogliono testare le due ipotesi alternative seguenti:

$$
\begin{aligned}
H0 : A &= B; \\
H1 : A &\neq B.
\end{aligned}
$$

ovvero l'ipotesi secondo cui la distribuzione A sia la stessa della distribuzione B. A tale scopo è necessario innanzitutto costruire la statistica test di Wilcoxon:

$$
W_A = \sum_{i=1}^{n} rank(X_i),
$$

dove la funzione $rank()$ restituisce la posizione ordinale del campione X_j nell'ordinamento crescente di tutti i campioni X_i e Y_i. A questo punto è possibile ricavare la statistica di Mann-Whitney, definita come:

$$U = w_A - \frac{n(n+1)}{2} = \sum_{i=1}^{n}\sum_{j=1}^{m} 1_{(Y_j,\infty)}X_i,$$

che fornisce il numero di X_i maggiori di Y_j per ogni X_i e per ogni Y_j.

Assumendo che l'ipotesi nulla sia vera, allora per valori contenuti di n ed m i quantili della distribuzione W_A sono noti, mentre per alti valori di n ed m è possibile approssimare W_A a una gaussiana, la cui media e varianza possono essere ricavate dalla media e varianza di U. Scelto un livello di significatività α, è possibile formulare la seguente regola per decidere se rifiutare l'ipotesi nulla:

$$Rifiuto\ H0\ se\ W_A \notin [w_{\frac{\alpha}{2}}, w_{\frac{1-\alpha}{2}}],$$

ovvero, si rifiuta l'ipotesi nulla e si accetta l'ipotesi alternativa secondo cui le distribuzioni sono non omogenee, se la statistica test non appartiene all'intervallo definito dai quantili $w_{\frac{\alpha}{2}}, w_{\frac{1-\alpha}{2}}$ [24, 86].

Nel contesto della feature selection, si può classificare il test di Wilcoxon come un metodo *filter* basato su una tecnica di esplorazione dello spazio delle feature di tipo *best first univariata*. Il test di Wilcoxon può infatti essere utilizzato per selezionare un sottoinsieme di feature prima di eseguire una classificazione binaria. Per ogni possibile feature si valuta se la distribuzione di tale feature nella prima classe sia omogenea con la distribuzione della stessa feature nella seconda classe. Nel caso la distribuzione della feature nelle due classi risulti omogenea, la feature può essere rimossa dal sottoinsieme finale di feature, in quanto priva di potere discriminante tra le due classi. Al contrario, se il test di Wilcoxon fallisse e la distribuzione nelle due classi non risultasse omogenea, allora la feature può essere conservata nel sottoinsieme finale di feature.

Sequential Selection Gli algoritmi di sequential selection sono algoritmi greedy per la ricerca di un sottoinsieme di feature. Gli algoritmi di sequential selection possono essere utilizzati sia come metodi *wrapper* sia come metodi *embedded*; in entrambi i casi utilizzano un metodo *sequenziale* per l'esplorazione dello spazio delle feature. Tra i possibili algoritmi di sequential selection i due più noti e intuitivi sono:

Sequential Forward Selection (SFS) L'algoritmo di sequential forward selection parte da un sottoinsieme di feature vuoto e a ogni iterazione aggiunge la feature migliore al sottoinsieme finale di feature.
Più rigorosamente l'algoritmo può essere riassunto come segue:

1. $F = \{f_1, f_2, \ldots f_n\}$; $G = \{\emptyset\}$;
2. select $f_i \in F \mid \forall j, j \neq i, f_j \in F \quad P_{class}(G \cup f_i) > P_{class}(G) \wedge P_{class}(G \cup f_i) > P_{class}(G \cup f_j)$;

3. $F = F \setminus f_i$; $G = G \cup f_i$;
4. while (exists f_i) goto 2;
5. end.

dove F è l'insieme di tutte le possibili feature, G il sottoinsieme finale di feature e P_{class} la funzione che ritorna il valore di performance del classificatore dato un insieme di feature.

Sequential Backward Selection (SBS) All'opposto, l'algoritmo di sequential backward selection parte dall'intero insieme di possibili feature e a ogni iterazione rimuove la feature meno utile dal sottoinsieme finale di feature.
Più rigorosamente l'algoritmo può essere riassunto come segue:

1. $G = \{f_1, f_2, \dots f_n\}$;
2. select $f_i \in G \mid \forall j, j \neq i, f_j \in G \quad P_{class}(G \setminus f_i) > P_{class}(G) \wedge P_{class}(G \setminus f_i) > P_{class}(G \setminus f_j)$;
3. $G = G \setminus f_i$;
4. while (exists f_i) goto 2;
5. end.

dove G è il sottoinsieme finale di feature e P_{class} la funzione che ritorna il valore di performance del classificatore dato un insieme di feature.

Entrambe le procedure di sequential selection sono relativamente rapide e robuste all'overfitting [31]; generalmente, pur partendo da uno stesso insieme iniziale di feature è possibile che i due algoritmi giungano a soluzioni differenti; dopotutto, trattandosi di algoritmi greedy, il sequential forward selection e il sequential backward selection offrono un buon trade-off tra il tempo e la qualità dei risultati, ma non garantiscono il raggiungimento della soluzione ottima [41].

Algoritmi genetici Gli algoritmi genetici sono un metodo stocastico di ricerca basato sul paradigma dell'evoluzione biologica. Gli algoritmi genetici possono essere utilizzati sia come metodi *wrapper* sia come metodi *embedded* per la selezione di un sottoinsieme di feature; l'esplorazione dello spazio delle feature è basata su un metodo *randomizzato*.

Un algoritmo genetico è un metodo che imita il processo dell'evoluzione biologica, basandosi sull'assunzione secondo cui l'evoluzione, attraverso la selezione naturale, sia un processo di ottimizzazione efficiente [21]. Per fare questo un algoritmo genetico opera su una popolazione di potenziali soluzioni e a ogni iterazione genera nuove soluzioni applicando il principio della sopravvivenza del più adatto (*survival of the fittest*) [17].

Le componenti chiavi che definiscono un algoritmo genetico sono [41]:

Popolazione la popolazione di un algoritmo genetico in un dato istante è costituita da un insieme di individui, ciascuno dei quali rappresenta una possibile soluzione del problema;

Codifica la codifica specifica in quale modo le possibili soluzioni sono codificate nella popolazione. Ogni individuo della popolazione è rappresentato per mezzo di una stringa, detta *cromosoma*, definita su un dato alfabeto; ogni possibile configurazione del cromosoma, detta *genotipo*, è in relazione univoca con una possibile soluzione del problema, detta *fenotipo* [17];

Operatori genetici gli operatori genetici specificano in quale modo la ricerca procede nello spazio delle soluzioni nel corso dell'evoluzione; per simulare l'evoluzione biologica naturale, a ogni iterazione alcune funzioni per la manipolazione di stringhe, dette *operatori genetici*, possono essere applicate ai genotipi della popolazione per generare individui con un nuovo cromosoma. I due operatori genetici più comuni sono:

1. Operatore di *cross-over:* date due stringhe di lunghezza l, l'operatore di cross-over seleziona in maniera casuale uno o più valori k^i compresi tra 1 e l e ritorna una stringa di lunghezza l in cui gli elementi da 1 a k^1 sono tratti dalla prima stringa, gli elementi da k^1 a k^2 sono tratti dalla seconda stringa, gli elementi da k^2 a k^3 sono tratti dalla prima stringa e così via; in base alla *building blocks hypothesis*, per mezzo dell'operatore di cross-over le singole sottoparti della soluzione di un problema, dette *schemata*, possono essere ricombinate per giungere infine alla soluzione finale;
2. Operatore di *mutazione*: data una stringa di lunghezza l, l'operatore di mutazione seleziona in maniera casuale uno o più valori k^i compresi tra 1 e l e ritorna una stringa di lunghezza l equivalente alla stringa originaria a eccezione degli elementi in posizione k^i, il cui valore è variato in maniera casuale; per mezzo dell'operatore di mutazione è possibile evitare che il processo di ricerca della soluzione ottima si arresti in un minimo locale.

Funzione di fitness definito K l'insieme di tutti i possibili cromosomi, la funzione di fitness è una funzione $f(x) : K \to \Re$ che valuta la bontà delle soluzioni trovate; a ogni iterazione, la funzione di fitness restituisce per ogni individuo della popolazione un valore numerico proporzionale alla performance della soluzione codificata nel genotipo di ogni individuo.

L'obiettivo di un algoritmo genetico è dunque quello di massimizzare la funzione di fitness, ovvero, generare nel tempo individui il cui cromosoma codifichi soluzioni sempre migliori nella speranza di raggiungere la soluzione ottima.

L'algoritmo base che definisce il funzionamento di un algoritmo genetico può essere sinteticamente riassunto come segue [44]:

1. *Inizializzazione*: generazione casuale di una popolazione di partenza costituita da n individui;

2. *Valutazione della fitness*: per ogni individuo della popolazione viene valutata la sua fitness;

3. *Evoluzione*: a partire dalla popolazione attuale viene prodotta una nuova generazione di n individui ripetendo m volte le seguenti operazioni:

   (a) *Selezione degli individui genitori*: dalla popolazione attuale sono selezionati due individui; per garantire il principio della sopravvivenza dei più adatti la probabilità che un individuo sia selezionato è dipendente dalla fitness dell'individuo stesso;

   (b) *Applicazione degli operatori genetici*: gli operatori genetici sono applicati in maniera probabilistica agli individui genitori per generare un individuo figlio;

   (c) *Inserimento nella nuova generazione*: l'individuo figlio viene inserito nell'insieme degli individui della nuova generazione che alla fine dello step 3 sostituiranno la popolazione attuale;

4. *Valutazione delle condizioni di terminazione*: se si riscontrano le condizioni per terminare l'algoritmo, si restituisce la migliore soluzione trovata; altrimenti si torna allo step 2.

Un algoritmo genetico è dunque un metodo di ricerca e ottimizzazione general-purpose che può essere applicato alla ricerca di un sottoinsieme ottimo di feature. Può essere opportuno notare che un algoritmo genetico si differenzia dai tradizionali metodi di ricerca e ottimizzazione per le seguenti ragioni [17]:

- L'algoritmo genetico conduce la ricerca valutando una popolazione di possibili soluzioni in parallelo, e non una singola soluzione alla volta;
- L'algoritmo genetico non richiede o necessita di informazioni ausiliarie; la direzione della ricerca è determinata esclusivamente dalla funzione di fitness;
- L'algoritmo genetico lavora sulla codifica delle soluzioni e non direttamente sulle soluzioni;
- L'algoritmo genetico usa regole di transizioni probabilistiche e non deterministiche.

Una delle conseguenze di queste proprietà è che, sebbene un algoritmo genetico possa raggiungere in un tempo infinito la soluzione ottima, in un tempo finito un algoritmo genetico fornisce all'utente una soluzione subottimale; tale soluzione dipende da vari fattori (e.g., codifica del problema, parametrizzazione dell'algoritmo, condizioni iniziali, evoluzione probabilistica) e, anche se generalmente è una buona soluzione, non necessariamente costituisce la soluzione ottima.

Feature projection

Una soluzione alternativa o complementare alla feature selection per affrontare il problema della dimensionalità dei dati è quella di ricorrere a metodi di *feature projection* (o *feature transformation*). Mentre nel caso della feature selection si procedeva a selezionare un sottoinsieme di tutte le possibili feature, conservando le feature informative ed eliminando quelle inutili, nel caso della feature projection si ricerca una trasformazione lineare o non-lineare che permetta di proiettare le feature da uno spazio N-dimensionale a uno spazio M-dimensionale, con $M \leq N$, conservando quanta più informazione possibile.

Anche in questo caso, a seconda della disponibilità dei target, ovvero dei valori attesi, possiamo suddividere le tecniche di feature projection in [30]:

Tecniche supervisionate le tecniche supervisionate sfruttano l'informazione offerta dai target per ridurre la dimensionalità dei dati con il vincolo di conservare quanto più possibile l'informazione discriminatoria delle feature in relazione ai target. Esempi di tecniche supervisionate sono la *linear discriminant analysis (LDA)* e la *non-parametric LDA (NPLDA)*;

Tecniche non supervisionate le tecniche non supervisionate sono prive dell'informazione offerta dal target e tentano quindi di ridurre la dimensionalità dei dati con il vincolo di conservare quanto più possibile la struttura dei dati. Esempi di tecniche non supervisionate sono la *principal component analysis (PCA)*, la *exploratory projection pursuit (EPP)* e *la indipendent component analysis (ICA)*.

Le tecniche di feature projection riescono a ridurre la dimensionalità dei dati sfruttando due proprietà dei dati reali: la prima consiste nel fatto che i dati reali in uno spazio a molte dimensioni sono sparsi, non occupano l'intero spazio e possono quindi essere proiettati in uno spazio a dimensionalità ridotta; la seconda riguarda il fatto che il risultato desiderato può comunque essere predetto recuperando, attraverso un processo simile all'interpolazione, l'informazione contenuta nei dati scartati [7].

Una buona tecnica di feature projection è quindi in grado di determinare un sottoinsieme di feature che garantisca un alto *information packaging*, ovvero la capacità di comprimere l'informazione utile in un numero ridotto di feature [77].

Si noti che, in alcuni particolari contesti, è anche possibile proiettare i dati da uno spazio N-dimensionale a uno spazio M-dimensionale, con $M > N$ [31]; nel caso di dati con dimensionalità ridotta è cioè possibile incrementare il numero di feature generando nuove feature a partire dalle feature originali; questa operazione è eseguita solo nel caso di problemi particolarmente complessi in cui le interazioni di primo ordine tra le feature non siano sufficienti per ottenere buoni risultati.

Di seguito si analizzano più in dettaglio alcuni dei metodi e degli algoritmi utilizzati per la proiezione di feature durante i nostri esperimenti in laboratorio.

Linear Discriminant Analysis La *linear discriminant analysis* (o *Fisher discriminant analysis* o *canonical discriminant analysis*) è una tecnica di feature projection supervisionata progettata per proiettare le feature in uno spazio dimensionale ridotto che massimizzi la discriminabilità tra le diverse classi di feature [41]; questo risultato è ottenuto proiettando le feature dallo spazio originale N-dimensionale a uno spazio M-dimensionale con $M \leq N$ che massimizzi la separabilità delle classi [22].

Più rigorosamente [22], si consideri un problema di classificazione con $C = \{1, 2...c\}$ classi e sia X la matrice delle osservazioni di dimensioni $m \times n$. Si supponga inoltre che il numero di feature sia maggiore del numero di classi, ovvero $n > c$. Per ogni classe $i \in C$ esiste allora un insieme M^i di m_i vettori di osservazioni di dimensione n:

$$M^i = \left\{ \mathbf{x}_1^i, \mathbf{x}_2^i ... \mathbf{x}_{m_i}^i \right\}.$$

Si definisce quindi la media per la classe i come:

$$\hat{\mu}_i = \frac{1}{m_i} \sum_{j=1}^{m_i} \mathbf{x}_j^i,$$

e la media tra le classi come:

$$\hat{\mu} = \frac{1}{m} \sum_{i=1}^{c} m_i \hat{\mu}_i.$$

Infine, si definisce la matrice di covarianza per la classe i nel seguente modo:

$$\hat{S}_i = \sum_{j=1}^{m_i} (\mathbf{x}_j^i - \hat{\mu}_i)(\mathbf{x}_j^i - \hat{\mu}_i)^T.$$

Da queste definizioni è ora possibile calcolare anzitutto la matrice della varianza intraclasse (S_w):

$$S_w \quad = \quad \sum_{i=1}^{c} \hat{S}_i.$$

In secondo luogo, è possibile calcolare anche la matrice della varianza totale (S_T) come:

$$S_T \quad = \quad \sum_{i=1}^{c} \sum_{j=1}^{m_i} (\mathbf{x}_j^i - \hat{\mu})(\mathbf{x}_j^i - \hat{\mu})^T. \tag{5.24}$$

Ora, la formula 5.24 può essere riscritta come:

$$
\begin{aligned}
S_T &= \sum_{i=1}^{c}\sum_{j=1}^{m_i}(\mathbf{x}_j^i + \hat{\mu}_i - \hat{\mu}_i - \hat{\mu})(\mathbf{x}_j^i + \hat{\mu}_i - \hat{\mu}_i - \hat{\mu})^T = \\
&= \sum_{i=1}^{c}\sum_{j=1}^{m_i}(\mathbf{x}_j^i - \hat{\mu}_i)(\mathbf{x}_j^i - \hat{\mu}_i)^T + \sum_{i=1}^{c}\sum_{j=1}^{m_i}(\hat{\mu}_i - \hat{\mu})(\hat{\mu}_i - \hat{\mu})^T = \\
&= S_w + \sum_{i=1}^{c}\sum_{j=1}^{m_i}(\hat{\mu}_i - \hat{\mu})(\hat{\mu}_i - \hat{\mu})^T.
\end{aligned}
$$

Si imponga ora che la matrice della varianza totale (S_T) sia uguale alla somma della matrice della varianza intraclasse (S_w) e della matrice della varianza interclasse (S_b):

$$S_T = S_W + S_b.$$

Da ciò segue che la matrice della varianza interclasse (S_b) è definita come:

$$S_b = \sum_{i=1}^{c}\sum_{j=1}^{m_i}(\hat{\mu}_i - \hat{\mu})(\hat{\mu}_i - \hat{\mu})^T.$$

Per proiettare i dati dallo spazio N-dimensionale a uno spazio $(c-1)$-dimensionale è necessario valutare $(c-1)$ funzioni discriminanti:

$$y_i = \mathbf{w}_i^t\mathbf{x} \quad i = 1...c,$$

ovvero risolvere l'equazione matriciale:

$$y = \mathbf{W}^t\mathbf{x}.$$

Ogni vettore di osservazioni N-dimensionale $\mathbf{x}$ è quindi proiettato in un vettore $(c-1)$-dimensionale $\mathbf{y}$. Calcolando le medie $(\tilde{\mu}_i, \tilde{\mu})$ e le matrici di varianza $(\tilde{S}_w, \tilde{S}_b)$ basate sul nuovo insieme di vettori $(c-1)$-dimensionale $\mathbf{y}$, si ottiene che:

$$
\begin{aligned}
\tilde{S}_w &= \mathbf{W^t}S_w\mathbf{W}; \\
\tilde{S}_b &= \mathbf{W^t}S_b\mathbf{W};
\end{aligned}
$$

ovvero si dimostra che anche la matrice di varianza intraclasse ($\tilde{S}_w$) e la matrice di varianza interclasse ($\tilde{S}_b$) sono ottenute come proiezione da uno spazio N-dimensionale a uno spazio $(c-1)$-dimensionale.

Per ottenere una proiezione utile è opportuno scegliere una matrice di trasformazione $\mathbf{W}$ che massimizzi la varianza interclasse e minimizzi la varianza intraclasse. Si può dunque scegliere come funzione da massimizzare il rapporto tra la matrice di varianza interclasse ($\tilde{S}_b$) e la matrice di varianza intraclasse ($\tilde{S}_w$):

$$J(\mathbf{W}) = \frac{\left|\tilde{S}_b\right|}{\left|\tilde{S}_w\right|}.$$

Si può infine dimostrare [22] che la matrice di trasformazione $\mathbf{W}$ ottima è una matrice rettangolare le cui colonne sono gli autovettori che corrispondono ai massimi autovettori ottenuti risolvendo:

$$S_b\mathbf{w}_i = \lambda_i S_W \mathbf{w}_i.$$

La linear discriminant analysis proietta quindi i dati da uno spazio N-dimensionale a uno spazio $(c-1)$-dimensionale o inferiore. Questo tipo di analisi presenta tuttavia alcuni limiti: anzitutto, se la distribuzione delle classi non è gaussiana, l'informazione derivante dalla struttura complessa dei dati potrebbe essere persa durante la proiezione; in secondo luogo, se l'informazione discriminatoria è contenuta nella varianza piuttosto che nella media dei dati, la linear discriminant analysis non è in grado di fornire una proiezione che evidenzi questa informazione [30].

Principal Component Analysis La *principal component analysis* (o *trasformata di Karhunen-Loeve*) è una tecnica di feature projection non supervisionata progettata per proiettare le feature in uno spazio dimensionale ridotto attraverso una trasformazione ortonormale che preservi la varianza dei dati originali [41].

Più rigorosamente [22], data la matrice delle osservazioni X di dimensioni $m \times n$, sia μ il vettore delle medie di dimensione $n \times 1$ e sia Σ la matrice di covarianza di dimensione $n \times n$. Si calcolino ora gli autovettori e_i e i relativi autovalori λ_i della matrice di covarianza Σ. Si ordinino infine gli autovettori e_i in ordine decrescente secondo l'autovalore λ_i associato. Dal momento che gli autovettori e_i rappresentano le direzioni principali della distribuzione e gli autovalori λ_i rappresentano la varianza della corrispondente direzione, selezionando i primi k autovettori si ottiene una proiezione dei dati in uno spazio dimensionale ridotto che preserva quanto più possibile la varianza dei dati originali. La principal component analysis proietta quindi i dati lungo gli assi che massimizzano la varianza dei dati o, equivalentemente, lungo quegli assi che minimizzano la sommatoria dei quadrati delle distanze tra i punti originali e le relative proiezioni [78].

La principal component analysis è un metodo di semplice applicazione, molto usato nel campo della compressione dati e della rappresentazione dei segnali [41]; tuttavia, nell'ambito della classificazione dei dati, in cui il criterio per valutare la bontà delle feature dovrebbe essere la discriminabilità tra le classi, la principal component analysis presenta un serio limite: infatti, trattandosi di una tecnica non supervisionata che non considera l'informazione offerta dalle classi, la principal component analysis non può garantire che la proiezione lungo l'asse di massima varianza sia la proiezione che massimizzi la discriminabilità tra le classi [30].

5.2.8 Classificazione

Dopo aver generato un insieme di feature e averne opportunamente ridotto la dimensionalità, è possibile utilizzare queste informazioni per dedurre le funzioni sottostanti ai dati.

Si ricordi che i dati prodotti possono essere rappresentati per mezzo della consueta matrice delle feature D, di dimensione $n \times m$, dove le righe denotano i singoli campioni acquisiti e le colonne le feature; e dal vettore delle classi C, di dimensione $n \times 1$, dove le righe denotano sempre i singoli campioni acquisiti e la colonna contiene il valore atteso (o classe):

$$\begin{array}{c} \\ t_1 \\ t_2 \\ \dots \\ t_N \end{array} \begin{array}{c} \overset{Feature}{\overbrace{\phantom{x_{11}\ x_{12}\ \dots\ x_{1M}}}} \\ \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1M} \\ x_{21} & x_{22} & \dots & x_{2M} \\ \dots & \dots & \dots & \dots \\ x_{N1} & x_{N2} & \dots & x_{NM} \end{bmatrix} \end{array} \quad \begin{array}{c} Valore\,atteso \\ \begin{bmatrix} trg_1 \\ trg_2 \\ \dots \\ trg_N \end{bmatrix} \end{array}.$$

Ora, il processo di *classificazione* consiste nell'uso di questi dati per il calcolo di una funzione $f(x)$ in grado di spiegare i dati disponibili; la funzione $f(x)$ ha come dominio lo spazio m-dimensionale delle feature e come codominio lo spazio unidimensionale dei valori attesi. La classificazione è quindi un particolare tipo di *apprendimento supervisionato* il cui l'obiettivo ultimo è quello di trovare una funzione $f(x)$ in grado di predire quanto meglio possibile le classi; ovvero, dato un vettore $1 \times m$ che costituisca l'$(n+1)$-esimo campione osservato e di cui è ignoto il valore atteso, la funzione $f(x)$ è in grado di restituire la classe dell'$(n+1)$-esimo campione sulla base dei valori attesi osservati negli n campioni precedenti.

Il processo di classificazione è costituito da tre distinti passi consecutivi [30]:

Scelta del classificatore la scelta del classificatore consiste nella selezione di un preciso classificatore e nell'eventuale definizione dei parametri del classificatore;

Addestramento del classificatore l'addestramento del classificatore consiste nel calcolo della funzione di classificazione $f(x)$; a partire dai dati contenuti nella matrice D e nel vettore C, un algortimo dipendente dal tipo di classificatore scelto computa una funzione di classificazione $f(x)$;

Valutazione del classificatore la valutazione del classificatore consiste nel calcolo della bontà della funzione di classificazione $f(x)$; utilizzando solo i dati contenuti nella matrice D è possibile confrontare le classi predette dalla funzione di classificazione $f(x)$ e le classi corrette contenute nel vettore C e avere in questo modo una stima della bontà del classificatore.

Più correttamente, per valutare un classificatore è necessario innanzitutto definire un'opportuna *loss function* $L(f(D), C)$, ovvero una funzione che, ricevendo in ingresso la funzione di classificazione $f(D)$, la matrice delle feature D e il vettore delle classi C, valuti quantitativamente quanto

bene la funzione di classificazione classifichi i dati disponibili; la più semplice *loss function* $L(f(D), C)$ è probabilmente la *misclassification function* che calcola, in valore assoluto o percentuale, il numero di campioni erroneamente classificati dalla funzione di classificazione $f(D)$; altri esempi di *loss function* $L(f(D), C)$ sono lo *squared error* o la *binomial deviance* [31].

Inoltre, se il processo di addestramento del classificatore e il processo di valutazione del classificatore sono condotti sugli stessi dati si corre il rischio di commettere un errore di *overfitting*; infatti, se si addestra il classificatore con i dati contenuti nella matrice D e lo si ottimizza valutandone la *loss function* con lo stesso set di dati, si rischia di ottenere un classificatore in grado di discriminare perfettamente i dati noti della matrice D, ma le cui reali performance sono inferiori a quelle calcolate. Un classificatore che commetta errori di *overfitting* ha, in altre parole, un limitato potere di generalizzazione, ovvero è incapace di classificare in maniera soddisfacente i dati che si acquisiranno in futuro; ciò è dovuto al fatto che essendo stato ottimizzato per ottenere i risultati migliori possibili sul set di dati della matrice D, il classificatore ha appreso anche informazioni non significative contenute nella matrice D (e.g., rumore) che influenzano negativamente i risultati ottenuti nel momento in cui si classificano nuovi dati. Per risolvere questo problema sono state proposte diverse soluzioni [30]:

- *Holdout:* il metodo dell'holdout consiste nel dividere il set di dati D in due sottoinsiemi $D^{(1)}$ e $D^{(2)}$, il primo dei quali è utilizzato per il processo di addestramento e il secondo per il processo di valutazione; il limite principale di questo metodo è che in presenza di un numero limitato di dati, sia il processo di addestramento che il processo di valutazione possono essere influenzati dallo specifico sottoinsieme di dati considerato;
- *Random sub-sampling:* il metodo del random sub-sampling consiste nell'estrarre dal set di dati D un sottoinsime H di h dati selezionati casualmente; il processo di addestramento è quindi eseguito utilizzando i dati in $D \setminus H$, mentre il processo di valutazione è eseguito utilizzando i dati in H; questo procedimento è ripetuto k volte, durante ciascuna delle quali h nuovi dati sono scelti casualmente e il classificatore è addestrato da zero; la performance finale è data dalla media delle performance calcolate a ogni iterazione;
- *k-fold cross-validation:* il metodo di k-fold cross-validation consiste nel dividere il set di dati D in k sottoinsiemi, $D^{(1)}$, $D^{(2)} \dots D^{(k)}$; quindi, per k volte, a ogni iterazione i, il processo di addestramento è eseguito utilizzando i dati in $D^{(1)}$, $D^{(2)} \dots D^{(i-1)}$, $D^{(i+1)} \dots D^{(k)}$, mentre il processo di valutazione è eseguito utilizzando i dati in $D^{(i)}$. La performance finale è data dalla media delle performance calcolate a ogni iterazione; il metodo di k-fold cross-validation è molto simile

al metodo di random sub-sampling, ma ha il vantaggio che tutti i dati vengono sicuramente utilizzati sia per l'addestramento sia per la valutazione. Il valore numerico di k è un parametro variabile, comunemente compreso tra 3 e 10; è noto che alti valori di k garantiscono una stima della performance del classificatore più accurata, ma comportano anche una varianza della performance maggiore e tempi di computazione più lunghi; viceversa con bassi valori di k si ottengono risultati in un tempo più breve, con una varianza della performance minore, ma con un'accuratezza inferiore nella stima delle performance.

- *Leave-one-out cross-validation:* il metodo di leave-one-out cross-validation è il caso estremo di k-fold cross-validation quando $k = |D|$;
- *Bootstrap:* il metodo di bootstrap consiste nell'estrazione con ripetizione dal set di dati D di un sottoinsieme H di h campioni; quindi, per k volte, a ogni iterazione i, il processo di addestramento è eseguito utilizzando i dati in H, mentre il processo di valutazione è eseguito utilizzando i dati in $D \setminus H$; la performance finale è data dalla media delle performance calcolate a ogni iterazione.

Inoltre, in base al tipo di funzione $f(x)$ scelta per classificare i dati, si possono distinguere i classificatori in due grandi famiglie:

Classificatori lineari i classificatori lineari sono classificatori che, dato un vettore $1 \times m$ che costituisca l'$(i+1)$-esimo campione osservato, determinano la classe di tale campione in base a una combinazione lineare del valore delle feature; un classificatore lineare separa dunque le classi per mezzo di piani.

I classificatori lineari sono molto usati nel campo delle interfacce cervello-computer per via delle loro proprietà [56]: tali classificatori sono infatti concettualmente semplici, facili da implementare anche per utenti con poca esperienza e, a causa del numero limitato di parametri configurabili, hanno una flessibilità limitata e, conseguentemente, risultano più robusti e meno soggetti all'overfitting. Tuttavia in alcuni casi, soprattutto in presenza di outlier o rumore eccessivo, i risultati della classificazione lineare possono essere insoddisfacenti; in quest'ultimo caso, l'applicazione di tecniche di regolarizzazione e l'utilizzo di conoscenza a priori permettono di migliorare sensibilmente le performance. Non da ultimo, l'applicazione delle tecniche di *kernel trick,* che consentono di mappare uno spazio di input che richiederebbe una classificazione non-lineare in uno spazio delle feature in cui è possibile utilizzare una classificazione lineare, hanno permesso di ricorrere a classificatori lineari per affrontare problemi originariamente non lineari. In definitiva, i classificatori lineari sono generalmente preferiti per via della loro semplicità e risultano la scelta ideale qualora sia disponibile solo una quantità limitata di dati e di informazioni sui dati [56].

Classificatori non lineari i classificatori non lineari sono classificatori che, dato un vettore $1 \times m$ che costituisca l'$(i+1)$-esimo campione osservato, determinano la classe di tale campione in base a una combinazione del valore delle feature; un classificatore non lineare separa dunque due o più classi per mezzo di generiche superfici.
L'utilizzo dei classificatori non lineari necessita di una solida conoscenza del problema [56]: i classificatori non lineari richiedono infatti la scelta e la configurazione di diversi parametri che dovrebbero essere selezionati sulla base delle informazioni disponibili; la determinazione di tali parametri risulta essere un lavoro delicato, in grado di influenzare sia le prestazioni sia la robustezza del classificatore. I classificatori non lineari sono utilizzati principalmente quando si dispone di grandi moli di dati e, partendo da conoscenze a priori su questi dati, si vogliono ricercare strutture complesse nei dati [56].

Classificatore lineare standard

Si consideri innanzitutto il problema della classificazione tra due classi, $C1$ e $C2$. Dato un vettore di feature $\mathbf{x} = [x_1\, x_2\, \ldots\, x_n]$, il classificatore lineare equivale a una semplice funzione discriminante del tipo:

$$y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0,$$

dove y è l'indice della classe, $\mathbf{w} = [w_1\, w_2\, \ldots\, w_n]^T$ è il vettore dei pesi e w_0 è il bias o threshold [77]; in altre parole $\mathbf{x}$ appartiene a $C1$ se $y > 0$, mentre appartiene a $C2$ se $y < 0$; il caso $y = 0$ può essere arbitrariamente assegnato a una delle due classi. L'equazione $y = 0$ corrisponde all'iperpiano H che separa le due classi; è possibile perciò distinguere due semispazi corrispondenti alle due regioni di decisione relative a $C1$ e a $C2$; l'orientamento dell'iperpiano è dato dal vettore normale $\mathbf{w}$, mentre la locazione dell'iperpiano è determinata dal bias. Alternativamente, è possibile interpretare il modulo di $y(\mathbf{x})$ come la distanza tra $\mathbf{x}$ e l'iperpiano H, e il segno di $y(\mathbf{x})$ come la regione di decisione nella quale si trova $\mathbf{x}$ [22].

Generalizzando al caso di K classi, si potrebbe pensare di risolvere il problema della classificazione lineare multiclasse combinando tra loro piu classificatori lineari binari. Una prima soluzione potrebbe essere quella di ricorrere a $K-1$ classificatori binari che risolvano il problema *one-versus-the-rest*, ovvero separino una singola classe da tutti i punti che non appartengono a quella classe. Una seconda soluzione potrebbe essere quella di costruire $\frac{K(K-1)}{2}$ classificatori binari che risolvano il problema *one-versus-one*, ovvero separino tra loro ogni possibile coppia di classi. Entrambe queste soluzioni, tuttavia, presentano il problema che combinando più classificatori binari differenti possono generarsi regioni di spazio ambigue che non appartengono a nessuna classe. Per risolvere questo problema, è conveniente definire una singola funzione discriminante multiclasse data dall'unione di K funzioni lineari:

$$y_k(\mathbf{x}) = \mathbf{w_k}^T\mathbf{x} + w_{k0}.$$

In questo modo, $\mathbf{x}$ appartiene alla classe C_k se:

$$y_k(\mathbf{x}) > y_j(\mathbf{x}) \quad \forall\, j \neq k,$$

conseguentemente, l'iperpiano che separa C_k e C_j sarà dato dall'equazione $y_k(\mathbf{x}) = y_j(\mathbf{x})$ [8]. Tale classificatore lineare definisce dunque un insieme di piani che partizionano lo spazio in K regioni di decisione, ciascuna delle quali risulta essere, per costruzione, convessa [22].

Per il calcolo dei parametri di un classificatore lineare è possibile ricorrere a diverse tecniche, tra cui metodi basati sulla Fisher discriminant analysis, metodi basati sui minimi quadrati o metodi basati sull'uso di semplici reti neurali (percettroni) [8].

Classificatore quadratico

Un classificatore quadratico è un classificatore non lineare che separa i dati in due o più classi per mezzo di superfici quadratiche.

Un classificatore quadratico può essere costruito come una generalizzazione di un classificatore lineare. Si consideri innanzitutto la formula di un classificatore lineare:

$$y_k(\mathbf{x}) = \mathbf{w_k}^T\mathbf{x} + w_{k0},$$

in cui $y_k(\mathbf{x})$ è il risultato della classificazione, $\mathbf{w}_k$ è il vettore dei pesi e w_{k0} è il bias o threshold.

Partendo da questa equazione è ora possibile aggiungere un ulteriore termine dato dal prodotto di coppie di componenti del vettore da classificare $\mathbf{x}$ e ottenere in questo modo una funzione discriminante quadratica [22]:

$$y_k(\mathbf{x}) = \mathbf{w_{kj}x_kx_j} + \mathbf{w_k}^T\mathbf{x} + w_{k0}.$$

I termini aggiuntivi $\mathbf{w}_{kj}$ permettono quindi di descrivere una superficie iperquadratica, ovvero una superficie discriminante di secondo grado nello spazio delle feature.

Si noti che è inoltre possibile ripetere questo procedimento di generalizzazione e, partendo da un classificatore quadratico, aggiungere nuovi termini; in questo modo è possibile costruire classificatori polinomiali che definiscono funzioni discriminanti lineari generalizzate del tipo [22]:

$$y_k(\mathbf{x}) = \mathbf{a}^T\phi(\mathbf{x}),$$

dove $\mathbf{a}$ è un vettore di pesi d-dimensionale e $\phi(\mathbf{x})$ è una funzione del vettore da classificare $\mathbf{x}$.

Il vantaggio principale di un classificatore quadratico e, più in generale, di un classificatore polinomiale è che essi permettono di risolvere problemi di classificazione che un semplice classificatore lineare non potrebbe risolvere. Il più emblematico tra questi è il cosiddetto problema dello *XOR* [77]; tale problema richiede di trovare un classificatore in grado di separare quattro punti (le cui

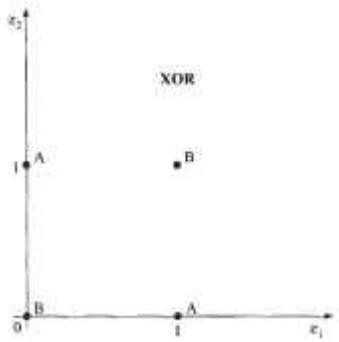

Figura 5.5: Problema dello *XOR* [77]

coordinate sul piano cartesiano sono $(0;0)$, $(0;1)$, $(1;0)$, $(1;1)$) divisi in due classi ($(0;1)$ e $(1;0)$ appartenenti alla prima classe e $(0;0)$ e $(1;1)$ appartenenti alla seconda classe); come evidente dalla Figura 5.5, questo problema non può essere risolto per mezzo di un classificatore lineare.

Tuttavia, l'uso di un classificatore quadratico richiede di determinare un numero di parametri maggiore di un normale classificatore lineare; ciò implica la necessità di un maggiore numero di dati di training e un maggiore sforzo computazionale che, soprattutto nel caso di classificatori polinomiali, può risultare eccessivamente elevato [22].

Classificatore KNN

Il classificatore KNN (*k-nearest neighbours*) classifica i dati in base alla regola dei k campioni piu vicini (*k-nearest neighbours rule*).

Sia dato un insieme di N vettori $[z_1\, z_2\, \ldots\, z_n]$ di training di cui sia nota la classe e sia $\mathbf{x} = [x_1\, x_2\, \ldots\, x_n]$ un vettore di feature di cui non è nota la classe. La regola dei k campioni piu vicini impone dunque di determinare K, ovvero l'insieme dei k vettori piu vicini a $\mathbf{x}$; la scelta è condizionata da due parametri: il numero di vicini k e la metrica per valutare la vicinanza. Per il numero di vicini generalmente si sceglie un valore di k dispari per i problemi di classificazione binaria o un valore che non sia un multiplo delle classi nel caso di problemi di classificazione multiclasse. Per la metrica è possibile ricorrere a diverse metriche, ad esempio, la distanza Euclidea o la distanza di Mahalanobis [77].

Una volta selezionato il sottoinsieme di k vicini si calcola, per ogni classe C_i, il numero di vicini E_i appartenenti alla classe i:

$$E_i = \sum_{j=1}^{k} 1_{k_j \in C_i}.$$

Infine si assegna $\mathbf{x}$ alla classe C_i a cui appartengono il maggior numero di vicini.

$$y(x) = C_i \mid E_i > E_j \ \ \forall j \neq i.$$

Il classificatore KNN è un classificatore concettualmente semplice e di facile implementazione [22]. Tuttavia esso presenta anche alcuni seri limiti. Anzitutto per poter operare, il classificatore KNN necessita che l'intero set di dati di training sia sempre disponibile; nel caso questo set di dati sia particolarmente grande, ciò comporta un elevato consumo di memoria e lunghi tempi di calcolo; questo problema potrebbe in parte essere risolto costruendo offline delle apposite strutture di ricerca che minimizzino lo spazio occupato in memoria e il tempo necessario per la classificazione [8]. Un secondo problema è connesso alla scelta dei parametri dell'algoritmo KNN, tra cui la metrica per il calcolo della distanza e il numero di vicini k da considerare; in particolare è evidente che modificando k le performance della classificazione e i tempi di calcolo possono variare consistentemente. Da ultimo, è opportuno evidenziare che le performance sono influenzate dal numero di dati di training N disponibili; in particolare, aumentando il numero dei dati di training e ottenendo così per il vettore $\mathbf{x}$ un insieme di vicini quanto più prossimo al vettore ignoto $\mathbf{x}$ è possibile incrementare le prestazioni; idealmente, le performance ottime si ottengono per $N \to \infty$ [22].

Classificatore ad albero

Un classificatore ad albero (o classificatore gerarchico) è un classificatore non lineare in grado di eseguire una classificazione attraverso un procedimento di decisione a più stadi. Idealmente il classificatore ad albero opera escludendo sequenzialmente le possibili classi fino a raggiungere l'ultima classe accettabile [77]; a ogni stadio successivo il classificatore ad albero risolve un nuovo problema impostato su un numero minore di feature [43]. Lo spazio delle feature viene cosi sequenzialmente partizionato in una serie di regioni di decisione, ciascuna corrispondente a una classe.

Il processo di decisione che a partire dal vettore da classificare $\mathbf{x}$ porta alla determinazione della classe $y(\mathbf{x})$ è convenientemente rappresentato per mezzo di un albero di decisione, in cui a partire dal nodo radice iniziale derivano una serie di rami (o connessioni), corrispondenti agli esiti delle singole decisioni, fino a raggiungere i nodi foglia che identificano il risultato della classificazione [22]. Per la generazione dell'albero di decisione esistono differenti tecniche, tra cui CART, ID3 e C4.5 [22].

Un classificatore ad albero presenta molti vantaggi. Di certo, il maggiore beneficio di un classificatore ad albero è la sua interpretabilità; dal momento che le decisioni prese ai vari livelli dell'albero possono essere espresse con formule logiche interpretabili da un'operatore umano, risulta semplice comprendere le ragioni che determinano il risultato della classificazione; a differenza di altri metodi di classificazione, il classificatore ad albero permette dunque di capire le dinamiche che determinano la classificazione. Dalla facile interpretabilità segue anche un'altra utile proprietà: la semplicità con cui è possibile integrare

la competenza di dominio o la conoscenza a priori all'interno del classificatore; dal momento che tale sapere può essere espresso in forma logica, risulta naturale inserire tale conoscenza all'interno della struttura ad albero del classificatore. Infine, il classificatore ad albero risulta computazionalmente efficiente sia dal punto di vista dello spazio che del tempo; esso, infatti, occupa un limitato spazio in memoria, dal momento che non richiede che l'intero set di dati di training sia conservato; inoltre, il processo di decisione può essere ridotto a una serie di interrogazioni logiche di rapida esecuzione [8].

Il principale problema di un classificatore ad albero riguarda invece la sua costruzione. Partendo dai vettori di dati di training e privi di conoscenza a priori, si presentano alcuni problemi, tra cui quali feature utilizzare per il processo di decisione o quanto a lungo sviluppare l'albero di decisione. In generale per costruire un albero di decisione è necessario affrontare i seguenti problemi [77]:

1. A ogni livello dell'albero è necessario identificare un insieme di espressioni logiche che permettano la ramificazione dell'albero;

2. Per ogni insieme di espressioni logiche è necessario scegliere l'espressione logica che garantisca le migliori performance dell'albero di decisione;

3. E' necessario determinare un criterio di arresto, in base al quale arrestare la crescita dell'albero;

4. Per ogni nodo foglia è necessario assegnare una classe.

Da ciò è evidente che la progettazione di un classificatore ad albero ottimale è un processo molto lungo che richiederebbe una ricerca esaustiva dello spazio di tutte le possibili strutture ad albero e delle combinazioni di feature; dal punto di vista pratico, la costruzione dell'albero è dunque generalmente eseguita per mezzo di metodi di ricerca subottimali [43]. A rendere il problema ancora più complesso, vi è il fatto che può essere dimostrato che l'ottimizzazione locale dei processi di decisione ai singoli livelli dell'albero non implica necessariamente un miglioramento globale delle performance generali del classificatore ad albero [43].

Classificatore SVM

Un classificatore SVM (*support vector machine*) è un classificatore lineare che permette di classificare i dati di due o più classi che siano linearmente separabili o non-linearmente separabili [77].

Nel caso di dati linearmente separabili, il classificatore SVM ricerca tra i possibili iperpiani che separano i dati, quell'iperpiano che massimizza la proprietà di margine, ovvero che massimizza la distanza minima tra l'iperpiano stesso e punti appartenenti a classi diverse più vicini all'iperpiano [22, 41]; più precisamente, il classificatore SVM cerca di massimizzare la distanza tra l'iperpiano di decisione e i vettori, paralleli al piano di decisione, passanti per i punti più vicini all'iperpiano e appartenenti a classi differenti; tali vettori sono chiamati

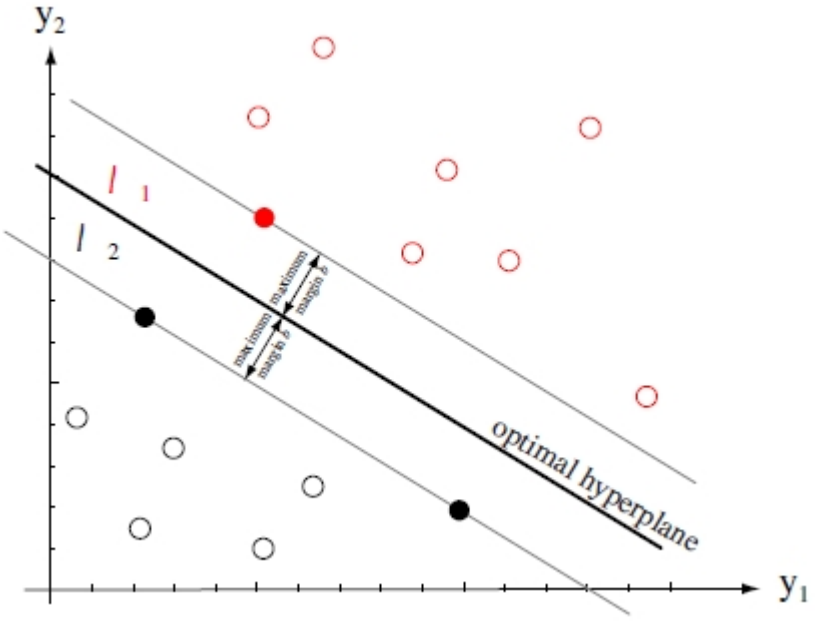

Figura 5.6: Classificatore SVM. [77]

vettori di supporto, da cui il nome di questo algoritmo di classificazione (vedi Figura 5.6).

Nel caso di dati non linearmente separabili, il classificatore SVM ricerca e calcola l'iperpiano che massimizza il margine e minimizza il limite superiore dell'errore di classificazione; il trade-off tra la massimizzazione del margine e la minimizzazione dell'errore di classificazione è determinato da una costante scelta a priori.

Dato un vettore di feature $\mathbf{x} = [x_1\, x_2\, \ldots\, x_n]$, l'obiettivo è sempre quello di trovare una semplice funzione discriminante del tipo:

$$y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0,$$

dove y è l'indice della classe, $\mathbf{w} = [w_1\, w_2\, \ldots\, w_n]^T$ è il vettore dei pesi e w_0 è il bias o threshold. Si noti che ogni possibile iperpiano è caratterizzato dalla sua direzione, data da $\mathbf{w}$, e dalla sua esatta posizione nello spazio, determinata da w_0; una volta trovati i piani che massimizzano il margine per ogni possibile direzione, è necessario scegliere la direzione che garantisca il massimo margine possibile. Per trovare una soluzione è opportuno impostare un problema di ottimizzazione [77]:

$$min\;\; J(\mathbf{w}) = \frac{1}{2}\left\|\mathbf{w}\right\|^2,$$

$$vincolo \;\; y_i(\mathbf{w}^T\mathbf{x}_i + w_0) \geq 1 \quad i = 1, 2...n,$$

in cui si cerca di massimizzare il margine tra il piano e i vettori di supporto rispettando il vincolo che nessun punto sia contenuto nello spazio tra i vettori di supporto e l'iperpiano di decisione. Questo è un problema di ottimizzazione appartenente alla classe dei problemi di programmazione quadratica, il cui metodo di risoluzione standard è il metodo dei *moltiplicatori di Lagrange*. E' provato che tale problema ammette un'unica soluzione e che, conseguentemente, l'iperpiano di decisione selezionato dal classificatore SVM è unico [43].

Nel caso di dati non linearmente separabilili la condizione di vincolo specificata precedentemente non può essere più rispettata; comunque si scelga l'iperpiano di decisione sarà possibile avere punti contenuti nello spazio delimitato dai vettori di supporto oppure punti contenuti nello spazio di un'altra classe. Per risolvere questo problema è possibile riscrivere il problema di ottimizzazione precedente nel seguente modo:

$$min \;\; J(\mathbf{w},w_0,\xi_i) = \frac{1}{2}\left\|\mathbf{w}\right\|^2 + C\sum_{i=1}^{n}\xi_i,$$

$$vincolo \;\; y_i(\mathbf{w}^T\mathbf{x}_i + w_0) \geq 1 - \xi_i \quad i = 1, 2...n,$$

dove è stata introdotta una nuova costante C e delle nuove variabili, ξ_i, dette *slack variables*; il nuovo obiettivo è dunque quello di massimizzare il margine del classificatore SVM e, nel contempo, mantenere il numero di punti con $\xi_i > 0$ il più basso possibile; il trade-off tra ampiezza del margine e numero di punti che non risultano linearmente separabili è determinato dal valore della costante C. Ancora una volta questo problema di ottimizzazione è un problema di programmazione quadratica risolvibile calcolando i moltiplicatori di Lagrange [77, 43].

Si noti, infine, che un classificatore SVM può essere opportunamente esteso per calcolare superfici di decisione non lineari invece di semplici iperpiani [41]; il metodo più utilizzato per determinare funzioni di decisione non-lineari è il metodo del kernel; tale metodo permette di proiettare le feature in uno spazio con più dimensioni e utilizzare quindi, in questo nuovo spazio, un classificatore SVM lineare [43].

5.2.9 Feedback

L'ultima operazione eseguita da un'interfaccia cervello-computer consiste nell'utilizzo del segnale per generare un output e offrire un feedback all'utente. È possibile usare il segnale classificato per determinare un output oppure utilizzare il segnale non classificato come input di un'apposita funzione creata per calcolare la risposta dell'interfaccia cervello-computer. Utilizzare il segnale classificato per determinare l'output significa far corrispondere un determinato feedback a ciascuna possibile classe; il valore del feedback per ogni classe dovrebbe essere scelto in maniera da facilitare il compito dell'utente. Utilizzare il segnale in una

funzione per calcolare l'output è molto usato soprattutto in quelle applicazioni per interfacce cervello-computer che presentano un output continuo e facilmente computabile, come le interfacce cervello-computer sensomotorie per controllare il movimento di un cursore in una o due dimensioni su uno schermo.

Il feedback da parte di un'interfaccia cervello-computer è una parte fondamentale del corretto funzionamento dell'interfaccia; esso non è da considerarsi come una funzionalità aggiuntiva dell'applicazione dell'interfaccia cervello-computer, ma come una parte integrante del sistema. Il ruolo del feedback è essenziale soprattutto durante la fase di addestramento: è infatti attraverso la risposta del sistema che l'utente può comprendere se ha un controllo sull'interfaccia cervello-computer e in che modo migliorare questo controllo; gli effetti del feedback durante le prime fasi permettono dunque a un soggetto di sviluppare un'abilità duratura di controllo dell'interfaccia cervello-computer [48]. Ma l'importanza del feedback non si limita alla fase di apprendimento: è infatti dimostrato [48] che un'applicazione cervello-computer che offra un feedback continuo e costante permette a un utente di ottenere delle performance migliori rispetto alla stessa applicazione priva di feedback. Un opportuno feedback può inoltre funzionare come stimolo e mantenere l'attenzione del soggetto alta per tutta la durata dell'utilizzo dell'interfaccia cervello-computer; si noti, tuttavia, che gli effetti del feedback possono variare da soggetto a soggetto: per alcuni possono essere uno stimolo e una guida, per altri possono essere fonte di distrazione e di frustrazione; sarebbe quindi una buona pratica verificare l'opportunità di un dato feedback con ogni utente. In generale, comunque, sebbene sia possibile sospendere il feedback per brevi periodi, il suo contributo all'apprendimento del controllo di un'interfaccia cervello-computer sembra insostituibile.

Capitolo 6

Sistema sviluppato e sua validazione

Nel presente capitolo si espongono innanzitutto le configurazioni sperimentali e i protocolli di laboratorio utilizzati per l'acquisizione dei dati; in seguito, l'attenzione è volta alla descrizione e alla valutazione del nostro sistema di analisi e classificazione del segnale che è stato implementato presso il Laboratorio di Intelligenza Artificiale e Robotica; da ultimo sono presentati i risultati ottenuti durante la validazione del nostro sistema.

Si ricorda che l'obiettivo ultimo del presente lavoro di tesi è stato lo studio, la realizzazione e la valutazione dei processi e degli algoritmi di calcolo delle feature, di feature projection, di feature selection e di classificazione in un'interfaccia cervello-computer basata su motor imagery; nel perseguire questo obiettivo abbiamo inoltre deciso di concentrare la nostra attenzione sui seguenti aspetti:

- *Analisi delle performance delle sessioni di motor imagery senza feedback:* abbiamo analizzato se un'interfaccia cervello-computer che utilizzi algoritmi elementari di analisi e classificazione del segnale sia in grado di garantire un controllo accettabile e quale sia quantitativamente il livello di controllo raggiunto; i dati ottenuti nella realizzazione di questo esperimento sono stati in seguito utilizzati come riferimento nelle successive analisi;

- *Analisi delle performance delle sessioni di motor imagery con feedback:* abbiamo analizzato se l'introduzione di un feedback nell'interfaccia cervello-computer influisse sul controllo raggiunto dall'utente; abbiamo utilizzato gli stessi algoritmi elementari adottati nell'esperimento precedente e abbiamo aggiunto un feedback all'applicazione; i dati ottenuti nella realizzazione di questo esperimento ci hanno permesso di valutare il contributo del feedback sulle prestazioni dell'utente;

- *Analisi delle performance di diversi classificatori:* abbiamo sviluppato ulteriori e più complessi algoritmi per la feature projection, feature selection

e la classificazione; abbiamo quindi utilizzato gli stessi dati degli esperimenti precedenti per valutare le performance raggiunte dalle nuove combinazioni di algoritmi di analisi e classificazione del segnale; i dati ottenuti ci hanno permesso di dare una valutazione sul contributo dei singoli algoritmi e sulle differenze tra gli stessi;

- *Validazione statistica delle performance dei diversi classificatori:* per realizzare quanto detto precedentemente, abbiamo eseguito una serie di test statistici per la validazione delle differenze tra le performance dei diversi classificatori sviluppati; abbiamo utilizzato dei test multiclassificatore multidataset per garantire la validità statistica dei risultati calcolati; i dati ottenuti ci hanno permesso di concludere se le differenze registrate fossero realmente significative o dovute al caso;

- *Impatto della generazione e selezione automatica delle feature:* abbiamo sviluppato e realizzato un algoritmo genetico originale per la generazione e la selezione automatica delle feature, con l'intento di automatizzare l'intero processo di generazione e selezione delle feature; l'algoritmo è stato implementato in Matlab e testato sullo stesso insieme di dati su cui tutti i precedenti algoritmi sono stati valutati; i dati ottenuti ci hanno permesso di confrontare questo algoritmo con le altre soluzioni pre-esistenti e valutare la bontà dello stesso.

6.1 Configurazione sperimentale

Tutti gli esperimenti per lo sviluppo dell'interfaccia cervello-computer sono stati da noi condotti all'interno del Laboratorio di Intelligenza Artificiale e Robotica del Politecnico di Milano. Gli esperimenti sono consistiti nell'utilizzo da parte di volontari sani dell'interfaccia cervello-computer e nell'acquisizione dei relativi dati encefalografici.

Il sistema utilizzato per eseguire le acquisizioni può essere rappresentato come un sistema costituito da due blocchi principali, un blocco di acquisizione del segnale, composto dall'elettroencefalografo, e un blocco di elaborazione del segnale, costituito da un computer. Di seguito si analizza in dettaglio la configurazione di tali blocchi.

6.1.1 Acquisizione del segnale

Modulo di acquisizione amplificatore EEG EBNeuro BE Light; si tratta di un dispositivo per l'acquisizione e l'amplificazione di segnali elettroencefalografici, dotato di 21 canali di ingresso monopolari, 4 canali bipolari e 3 canali con riferimento separato (poligrafici);

Frequenza di acquisizione 4096 sample al secondo; si sono acquisiti 256 blocchi da 16 sample al secondo; la massima frequenza di acquisizione consentita dal modulo utilizzato è 8 KHz per canale;

Filtraggio filtro passa-alto a 0.1 Hz e filtro passa-basso a 1 KHz; il modulo di acquisizione integra un filtro passa-alto per la rimozione delle componenti continue del segnale e un filtro passa-basso per anti-aliasing; è presente poi anche un secondo filtro passa-basso digitale in grado di filtrare tutte le frequenze superiori a 0.45 volte la frequenza di campionamento scelta;

Montaggio degli elettrodi 21 elettrodi montati secondo il Sistema 10-20 per il Posizionamento degli Elettrodi; tutti gli elettrodi sono integrati in una cuffia, che garantisce un processo di preparazione rapido e semplice;

Segnali acquisiti Fp1, Fp2, F7, F3, Fz, F4, F8, T3, C3, Cz, C4, T4, T5, P3, Pz, P4, T6, O1, O2, EOG[1], dummy[2].

6.1.2 Elaborazione del segnale

Periferica di elaborazione online computer laptop; l'elaborazione dati online è stata condotta per mezzo di un comune computer laptop;

Collegamento alla periferica di acquisizione cavo in fibra ottica; il collegamento tra il modulo di acquisizione e la periferica di elaborazione è realizzato per mezzo di un cavo in fibra ottica che garantisce sia alte velocità di trasferimento dati che l'isolamento elettrico tra i moduli;

Software per l'acquisizione del segnale l'acquisizione del segnale è gestita per mezzo di Galileo, un software sviluppato da EBNeuro per i propri moduli di acquisizione elettroencefalografica; tale software è in grado di acquisire il segnale, mostrare a video i dati registrati ed eseguire delle semplici funzioni di analisi del segnale;

Software per l'elaborazione del segnale online l'elaborazione online del segnale è eseguita per mezzo di BCI2000, un software open source per lo sviluppo di intefacce cervello-computer realizzato presso il Wadsworth Center di Albany, New York; BCI2000 è un sistema general-purpose, che permette di integrare moduli personalizzati per l'acquisizione dei segnali, per l'elaborazione dei segnali e per l'applicazione e l'interfaccia utente;

Software per l'elaborazione del segnale offline Matlab; l'elaborazione offline del segnale è stata eseguita per mezzo di Matlab, un software di analisi numerica sviluppato da Mathworks.

6.2 Protocollo di laboratorio

Nel corso degli esperimenti in laboratorio, abbiamo condotto acquisizioni ripetute su cinque soggetti sani adulti maschi: FA, PC, TD, SM, FZ.

[1] Il segnale EOG è stato acquisito in maniera differenziale, per mezzo di due elettrodi posizionati al di sopra e al di sotto dell'occhio sinistro in configurazione vEOG.

[2] Il segnale dummy è un segnale di controllo costante generato dal modulo di acquisizione.

Durante ogni *giornata di acquisizioni* sono state realizzate acquisizioni su uno o più soggetti. Per ogni soggetto sono state eseguite una o più *sessioni di acquisizioni*. Ogni sessione di acquisizione era a sua volta costituita da una o più *run*. Ciascuna run è infine composta da uno o più *trial*. Un trial consiste in un compito elementare, durante il quale viene presentata su schermo una freccia la cui direzione è selezionata in maniera casuale e viene richiesto al soggetto di immaginare il movimento corrispondente alla direzione mostrata; le possibili immagini e i movimenti immaginari associati sono:

1. *Freccia verso sinistra,* corrispondente alla richiesta di immaginare il movimento della mano sinistra;
2. *Freccia verso destra,* corrispondente alla richiesta di immaginare il movimento della mano destra;
3. *Freccia verso l'alto,* corrispondente alla richiesta di immaginare il movimento di entrambe le mani;
4. *Freccia verso il basso,* corrispondente alla richiesta di immaginare il movimento di entrambi i piedi;
5. *Lettera "R"*, corrispondente alla richiesta di non immaginare alcun movimento.

Ogni trial è separato dal successivo da un intervallo di tempo variabile, durante il quale l'utente può rilassarsi, compiendo movimenti che durante la fase di immaginazione motoria è preferebile evitare (e.g., battere le palpebre).

Nel corso delle acquisizioni sono state condotti due differenti tipologie di acquisizioni:

Acquisizioni di motor imagery senza feedback durante queste prime acquisizioni è stato richiesto ai soggetti di immaginare il movimento presentato a schermo senza che fosse loro restituito alcun feedback.
Il protocollo per le acquisizioni di motor imagery senza feedback prevede la realizzazione di 2 sessioni di acquisizione per ogni giornata, separate da una pausa di tempo variabile durante la quale il soggetto ha modo di riposarsi. Ogni sessione comprende 7 run, ciascuna delle quali composta da 40 trial. La durata di ogni trial è di 6 secondi, seguita da un un intervallo di 1.5 secondi (a eccezione del ventesimo trial, dopo il quale vi è un intervallo più lungo, pari a 4 secondi). In ogni sessione sono stati quindi eseguiti 280 trial (vedi Figura 6.1).
I dati raccolti con acquisizioni di motor imagery senza feedback sono stati utilizzati per l'analisi delle prestazioni offline del classificatore e per lo studio delle performance di diversi algoritmi di calcolo delle feature, feature extraction e classificazione;

Acquisizioni di motor imagery con feedback durante queste acquisizioni è stato richiesto ai soggetti di immaginare il movimento presentato a schermo e, dopo ciascuna richiesta, era restituito all'utente il risultato della

Soggetto	Numero di giornate di motor imagery senza feedback	Numero di giornate di motor imagery con feedback
FA	4	-
PC	5	6
TD	3	3
SM	4	-
FZ	5	5

Tabella 6.1: Dati acquisiti durante gli esperimenti di motor imagery

classificazione del segnale elettroencefalografico acquisito.
Il protocollo per le acquisizioni di motor imagery con feedback prevede la realizzazione di 2 sessioni di acquisizione per ogni giornata, separate da una pausa di tempo variabile durante la quale il soggetto ha modo di riposarsi. Ogni sessione comprende 7 run, ciascuna delle quali composta da 40 trial. La durata di ogni trial è di 6 secondi, seguita da un un intervallo di 1.5 secondi (a eccezione del ventesimo trial, dopo il quale vi è un intervallo più lungo, pari a 4 secondi); durante ciascun intervallo è inoltre presentato all'utente come feedback il risultato della classificazione. In ogni sessione sono stati quindi eseguiti 280 trial (vedi Figura 6.1).
I dati raccolti con acquisizioni di motor imagery con feedback sono stati utilizzati per l'analisi delle prestazioni online del classificatore.

Per ogni soggetto sono stati acquisiti un ammontare di dati variabili, come riportato in Tabella 6.1.

6.3 Sistema di analisi e classificazione dei dati

I dati raccolti durante le sessioni di acquisizione sono stati importati in Matlab per poter eseguire analisi offline. Il sistema di analisi dei dati sviluppato in laboratorio è stato progettato e implementato in Matlab come un sistema componibile che, partendo dai dati dei diversi soggetti, permette di elaborare tali dati combinando tra loro diversi script Matlab; gli algoritmi implementati sono i seguenti:

Algoritmi per il calcolo dello spettro MEM, Marple

Algoritmi per il filtraggio in frequenza FIR, IIR

Algoritmi per il filtraggio spaziale Large Laplacian, CAR, ICA Infomax, Fast ICA, CSP, CSP Bin

Algoritmi per il calcolo delle feature Feature, Evolution, Sync Rate

Algoritmi di feature selection Fisher Rank, Sequential Forward Selection, Sequential Backward Selection, Wilcoxon test, Genetic Algorithm

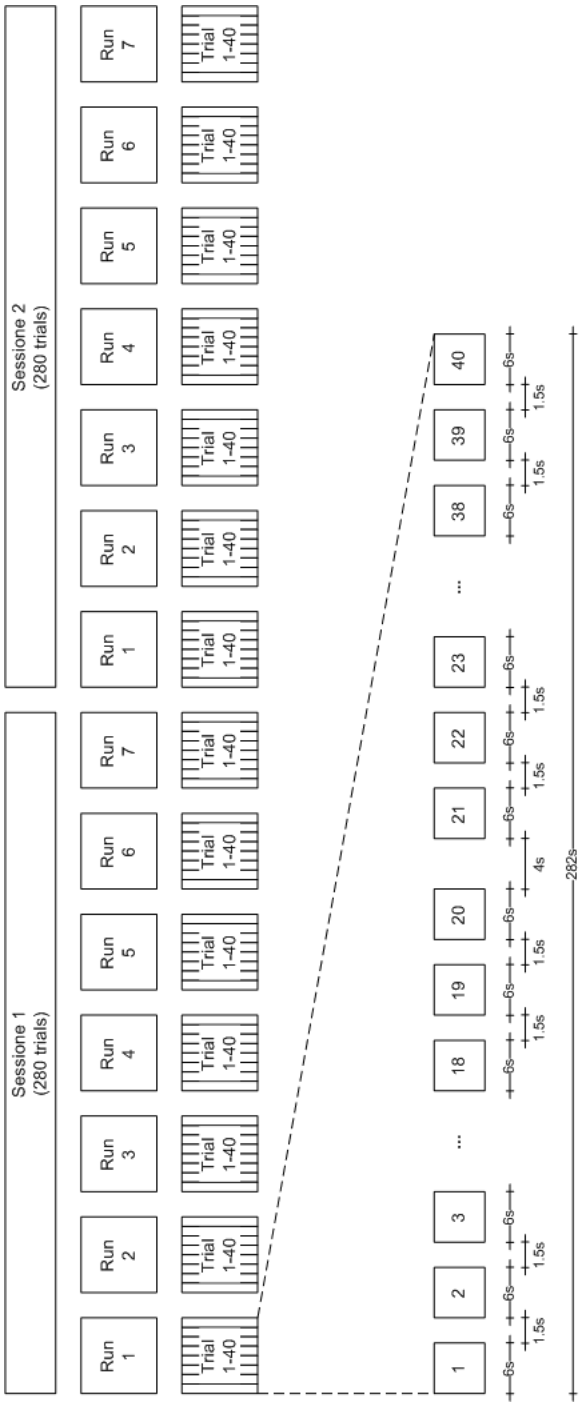

Figura 6.1: Protocollo di laboratorio

Algoritmi di feature projection LDA, PCA

Algoritmi di classificazione Lineare Multiclasse, Lineare Binaria, Ad Albero, Quadratica, KNN, SVM

La Figura 6.2 illustra l'architettura generale del sistema di analisi dei dati sviluppato in Matlab a partire dai dati che abbiamo raccolto in laboratorio. Il sistema è costituito da una serie ordinata di moduli connessi a formare una pipeline in analogia con un classico sistema di analisi e classificazione dei segnali (vedi Figura 5.1). I moduli sviluppati sono i seguenti:

- Modulo *batch_compute_spectra_for_feature()*: questo modulo riceve in ingresso i dati nel dominio del tempo e, a seconda dei dati forniti, dell'algoritmo per il calcolo dello spettro, dell'algoritmo per il filtraggio in frequenza e dell'algoritmo per il filtraggio spaziale scelti, trasforma i dati nel dominio delle frequenze; i risultati sono salvati su disco nella cartella */data/* e inoltrati al modulo successivo.

- Modulo *extract_alldate_all()*: questo modulo riceve i dati nel dominio delle frequenze e in base all'algoritmo per il calcolo delle feature selezionato, calcola un insieme di feature; i risultati sono scritti su disco nella cartella */feat_data/[feat_type]/* e inotrati al modulo successivo.

- Moduli ausiliari *compute_rank()*, *compute_ga()*, *compute_svm()*, *compute_knn()*: l'insieme di feature calcolate dal modulo precedente può essere opzionalmente inviato a questi moduli ausiliari che utilizzano le feature per stimare il valore di parametri che saranno in seguito usati durante la fase di feature selection o di classificazione; ciascuno di questi moduli riceve i propri input e salva i risultati nelle relative posizioni su disco.

- Modulo *save_data_for_stat_analysis()*: questo modulo riceve in ingresso un insieme di feature e, a seconda degli algoritmi di feature selection, feature projection e classificazione impostati, valuta le performance finali del sistema di classificazione; qualora necessario, può accedere ai parametri stimati dai moduli ausiliari; i risultati della classificazione sono salvati su disco nella cartella */statistics/statistics_data/*.

- Modulo *perform_stat_test()*: questo modulo riceve in ingresso i risultati della classificazione e, in base ai test statistici selezionati, restituisce i risultati delle validazioni statistiche.

Come detto, nel contesto di questa tesi, l'interesse è stato rivolto principalmente agli algoritmi per il calcolo delle feature, agli algoritmi di feature selection, agli algoritmi di feature projection e agli algoritmi di classificazione; in tutte le successive analisi, se non specificato diversamente, si assume che nel modulo *batch_compute_spectra_for_feature()* sia stato usato l'algoritmo MEM per il calcolo dello spettro, l'algoritmo FIR per il filtraggio in frequenza e l'algoritmo Large Laplacian per il filtraggio spaziale.

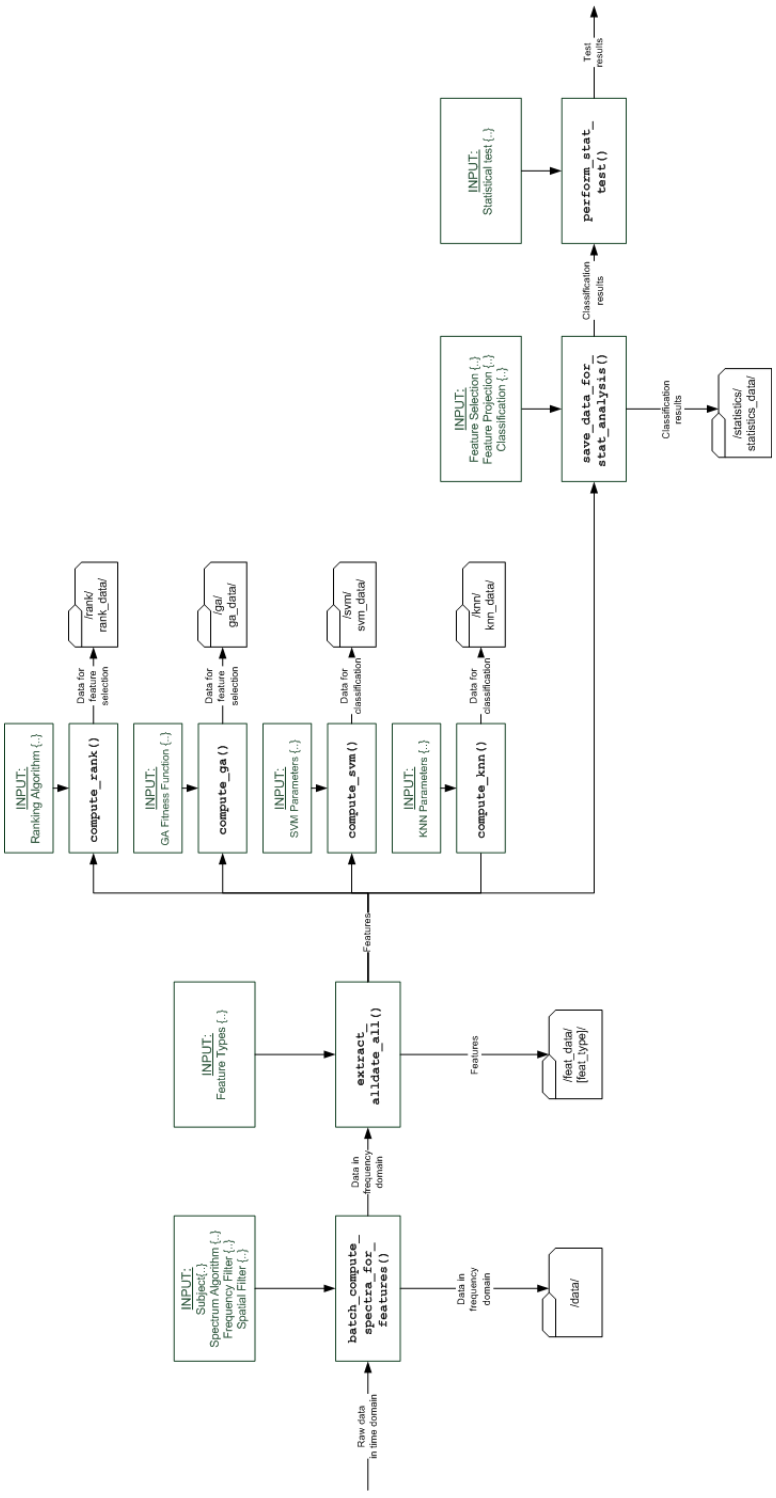

Figura 6.2: Schema a blocchi dei moduli e dei file Matlab costituenti il sistema di analisi offline.

	FA	PC	TD	SM	FZ	Media (*std dev*)
Accuratezza media classe 1 (mano sinistra)	0.3839	0.8000	0.4673	0.5915	0.4339	0.5353 (±0.1666)
Accuratezza media classe 2 (mano destra)	0.3505	0.8429	0.5655	0.7321	0.4607	0.5903 (±0.1993)
Accuratezza media classe 3 (entrambe le mani)	0.3036	0.7554	0.4345	0.5469	0.3054	0.4691 (±0.1893)
Accuratezza media classe 4 (entrambi i piedi)	0.5603	0.9125	0.7887	0.7143	0.7107	0.7373 (±0.1284)
Accuratezza media globale	*0.3996*	*0.8277*	*0.5640*	*0.6462*	*0.4777*	*0.5830* (±0.1651)
Deviazione standard globale	0.0644	0.0435	0.0220	0.0484	0.0680	0.0493 (±0.0184)

Tabella 6.2: Performance nelle sessioni di motor imagery senza feedback

6.4 Risultati

6.4.1 Risultati delle performance delle sessioni di motor imagery senza feedback

Le performance base delle sessioni di motor imagery senza feedback sono state valutate calcolando l'accuratezza di un semplice classificatore lineare. Sono state definite quattro classi corrispondenti all'immaginazione motoria della mano sinistra, all'immaginazione motoria della mano destra, all'immaginazione motoria di entrambe le mani e all'immaginazione motoria di entrambi i piedi (vedi Paragrafo 6.2); i dati raccolti sono stati processati utilizzando gli algoritmi Feature ed Evolution per il calcolo delle feature (vedi Paragrafo 6.3), nessun algoritmo di feature selection, l'algoritmo LDA per la feature projection e un algoritmo di classificazione lineare multiclasse per la classificazione (vedi Paragrafo 6.3); l'accuratezza è stata calcolata usando come *loss function* la *misclassification function* (vedi Paragrafo 5.2.8);

La Tabella 6.2 riporta l'accuratezza media nell'esecuzione di ogni singolo task, la media e la deviazione standard nell'esecuzione di ogni singolo task, l'accuratezza media globale e la deviazione standard dell'accuratezza globale per ognuno dei cinque soggetti (FA, PC, TD, SM, FZ).

6.4.2 Risultati delle performance delle sessioni di motor imagery con feedback

Lo stesso metodo esposto nel paragrafo precedente è stato utilizzato per valutare le performance base delle sessioni di motor imagery con feedback.

Come esposto nel Paragrafo 5.2.9 esistono due metodi principali per convertire il segnale di un'interfaccia cervello-computer basata su motor imagery in un feedback per l'utente: feedback basato sull'utilizzo di un segnale classificato o feedback basato l'utilizzo di un segnale continuo. Entrambe queste soluzioni sono state implementate e testate.

In primo luogo è stato sviluppato un feedback basato sull'utilizzo di un segnale classificato, simile al feedback realizzato durante i primi esperimenti condotti presso il Wadsworth Center; J.R. Wolpaw e i suoi colleghi hanno creato un'applicazione di un'interfaccia cervello-computer che prevedeva il movimento monodimensionale di un cursore [91]; dopo aver scelto come frequenza di interesse la frequenza del ritmo mu, il segnale è stato classificato in cinque classi in base all'ampiezza: nella prima classe rientravano i segnali con ampiezza 0-1μv, nella seconda segnali con ampiezza 1-2 μv, nella terza segnali con ampiezza 2-3 μv, nella quarta segnali con ampiezza 3-4 μv e nella quinta e ultima classe segnali con ampiezza maggiore di 4 μv; per ogni utente dell'interfaccia cervello-computer, un operatore esperto determinava poi lo spostamento associato a ogni classe, espresso in passi (steps), tali che un numero negativo di passi significasse uno spostamento del cursore verso il basso e un numero positivo uno spostamento verso l'alto; ad esempio, per uno dei soggetti, il feedback corrispondente a ciascuna delle cinque classi sopraelencate era rispettivamente -12 passi, 0 passi, +5 passi, +18 passi, +36 passi. Analogamente, il nostro sistema utilizza le feature generate dal segnale elettroencefalografico per discriminare la volontà dell'utente; quest'ultima viene classificata in quattro classi corrispondenti alle direzioni secondo cui è possibile muovere un cursore e per ognuna è offerto un feedback differente (e.g., una freccia che indica la direzione selezionata dall'utente o una parte dello schermo corrispondente alla direzione scelta che si illumina).

È stato inoltre sviluppato anche un feedback basato sull'utilizzo di un segnale continuo. Anche in questo caso si è adottata una soluzione studiata e realizzata presso il Wadsworth Center [89, 94, 73]; il segnale ricevuto dall'utente è elaborato dal sistema come variabile indipendente in una equazione lineare che restituisce l'ampiezza e la direzione del movimento del cursore sullo schermo. La formulazione di questa equazione lineare è la seguente [51]:

$$\Delta V = b(x - a),$$

dove:

$$\begin{aligned} \Delta V &= \textit{movimento del cursore}, \\ x &= \textit{segnale registrato ed elaborato}, \\ a &= \textit{intercetta dell'equazione lineare}, \\ b &= \textit{pendenza dell'equazione lineare}. \end{aligned}$$

Nell'applicazione di un interfaccia cervello-computer per muovere un cursore che utilizza questa formula, la pendenza b determina l'ampiezza del movimento del cursore ogni volta che la sua posizione viene aggiornata; il valore numerico della pendenza può essere calcolato manualmente o automaticamente in maniera da limitare la durata media di ogni esecuzione; infatti a un valore maggiore dell'intercetta corrisponde una velocità maggiore del cursore; una scelta comune è quella di utilizzare una pendenza contenuta durante le prime sessioni così da rendere i trial più lunghi (nell'ordine di 5 secondi) per poi aumentare il valore della pendenza e diminuire la durata dei trial (nell'ordine di 2 secondi) quando l'utente acquista confidenza con l'interfaccia cervello-computer. L'intercetta a è responsabile della direzione del movimento del cursore; è scelta in modo che se l'attuale segnale non differisce dai segnali registrati precedentemente, allora il movimento totale del cursore sia nullo; lo scopo è quindi quello di ridurre la tendenza del cursore a muoversi in una data direzione e massimizzare così il controllo dell'utente; il valore numerico dell'intercetta può essere determinato automaticamente online calcolando il valore medio dei segnali registrati nel corso dei precedenti trial; per ottenere le performance migliori l'intercetta dovrebbe essere scelta in maniera che tutti i target siano equamente accessibili [51]. L'intercetta permette dunque una sorta di normalizzazione del segnale. In generale, i parametri a e b dovrebbero essere scelti in maniera da permettere all'utente di colpire qualsiasi target con la stessa facilità, in quanto si è dimostrato che con questa configurazione si ottengono le performance migliori [51].

Una formulazione simile, ma tale che il movimento finale del cursore sia determinato dal segnale proveniente da più canali o da diverse frequenze è [90]:

$$\Delta V = b \left(\left(\sum_{i=1}^{n} w_i x_i \right) - a \right),$$

dove:

$$w_i = \textit{peso del segnale dal canale o della frequenza } i.$$

Si noti che questa formulazione altro non è che l'unione dell'equazione lineare per calcolare il feedback con un'equazione per la combinazione lineare del segnale.

Sebbene sia il feedback basato su segnale classificato sia il feedback basato su segnale continuo siano stati implementati con successo, la nostra attenzione si è concentrata principalmente sul primo metodo per la generazione del feedback. La Tabella 6.3 riporta quindi l'accuratezza media nell'esecuzione di ogni singolo

	PC	TD	FZ	Media (*std dev*)
Accuratezza media classe 1 (mano sinistra)	0.6771	0.4196	0.6750	0.5906 (±0.1481)
Accuratezza media classe 2 (mano destra)	0.7946	0.4881	0.6696	0.6508 (±0.1541)
Accuratezza media classe 3 (entrambe le mani)	0.6548	0.3512	0.5500	0.5187 (±0.1542)
Accuratezza media classe 4 (entrambi i piedi)	0.8438	0.6697	0.8125	0.7753 (±0.0928)
Accuratezza media globale	*0.7426*	*0.4821*	*0.6768*	*0.6338* (±0.1355)
Deviazione standard globale	0.0664	0.0548	0.1245	0.0819 (±0.0373)

Tabella 6.3: Performance nelle sessioni di motor imagery con feedback

task, la media e la deviazione standard nell'esecuzione di ogni singolo task, l'accuratezza media globale e la deviazione standard dell'accuratezza globale per ogni singolo soggetto (PC, TD, FZ) in presenza di un feedback determinato da segnale classificato.

6.4.3 Risultati delle performance dei diversi classificatori

Nel tentativo di migliorare i risultati della classificazione, sono stati implementati e utilizzati diversi algoritmi di feature selection, di feature projection e di classificazione; varie combinazioni di tali algoritmi sono state testate al fine di analizzare l'impatto di ciascuno di questi metodi e incrementare le prestazioni dell'interfaccia cervello-computer. Si riportano di seguito le configurazioni utilizzate per testare le performance dei diversi classificatori:

1. Classificazione lineare multiclasse (*script_test_linear.m*):

   Feature selection -

   Feature projection LDA (*LDAFisherProjection.m*)

   Classificazione Classificatore lineare multiclasse (*test_feature.m*)

   Questa configurazione, priva di algoritmi di feature extraction, corrisponde alla classificazione elementare utilizzata per valutare le performance base del sistema, come riportato nel Paragrafo 6.4.1.

2. Classificazione lineare binaria con Fisher Rank
(*script_test_fisher.m*):

   Feature selection Fisher Rank (*fisher_rank.m*)

   Feature projection LDA (*LDAFisherProjection.m*)

   Classificazione Classificatore lineare binario (*test_feature_binary.m*)

   In questa configurazione le feature vengono prima ordinate e selezionate dall'algoritmo di Fisher Rank, proiettate dall'algoritmo LDA e infine classificate in quattro classi per mezzo di partizioni binarie dello spazio delle feature.

3. Classificazione lineare binaria con Wilcoxon test
(*script_test_wilcoxon.m*):

   Feature selection Wilcoxon test (*wilcoxon.m*)

   Feature projection LDA (*LDAFisherProjection.m*)

   Classificazione Classificatore lineare binario (*test_feature_binary.m*)

   In questa configurazione le feature vengono prima ordinate e selezionate dall'algoritmo per il test di Wilcoxon, proiettate dall'algoritmo LDA e infine classificate in quattro classi per mezzo di partizioni binarie dello spazio delle feature.

4. Classificazione ad albero
(*script_test_tree.m*):

   Feature selection -

   Feature projection -

   Classificazione Classificatore ad albero (*test_tree.m*)

   In questa configurazione le feature vengono immediatamente classificate da parte di un classificatore ad albero.

5. Classificazione lineare multiclasse con FFS
(*script_test_FFS.m*):

   Feature selection FFS (*FFS.m*)

   Feature projection LDA (*LDAFisherProjection.m*)

   Classificazione Classificatore lineare multiclasse (*test_feature.m*)

   In questa configurazione le feature vengono prima selezionate dall'algoritmo di Forward Feature Selection, proiettate dall'algoritmo LDA e infine classificate in quattro classi.

6. Classificazione lineare multiclasse con BFS
(*script_test_BFS.m*):

Feature selection BFS (*BFS.m*)

Feature projection LDA (*LDAFisherProjection.m*)

Classificazione Classificatore lineare multiclasse (*test_feature.m*)

In questa configurazione le feature vengono prima selezionate dall'algoritmo di Backward Feature Selection, proiettate dall'algoritmo LDA e infine classificate in quattro classi.

7. Classificazione lineare multiclasse con algoritmo genetico (*script_test_ga_255.m*):

   Feature selection GA (*genetic_algorithm_255.m*)

   Feature projection LDA (*LDAFisherProjection.m*)

   Classificazione Classificatore lineare multiclasse (*test_feature.m*)

   In questa configurazione un algoritmo genetico seleziona un sottoinsieme delle 255 feature originali (255 geni); le feature scelte sono poi proiettate dall'algoritmo LDA e infine classificate in quattro classi.

8. Classificazione lineare binaria con algoritmo genetico (*script_test_ga_257.m*):

   Feature selection GA (*genetic_algorithm_257.m*)

   Feature projection LDA (*LDAFisherProjection.m*)

   Classificazione Classificatore lineare binario (*test_feature_binary.m*)

   In questa configurazione un algoritmo genetico seleziona un sottoinsieme delle 255 feature originali (255 geni) e l'ordine con cui effettuare le partizioni binarie dello spazio delle feature (2 geni); le feature scelte sono poi proiettate dall'algoritmo LDA e infine classificate in quattro classi per mezzo di partizioni binarie dello spazio delle feature.

9. Classificazione lineare binaria con algoritmo genetico (*script_test_ga_767.m*):

   Feature selection GA (*genetic_algorithm_767.m*)

   Feature projection LDA (*LDAFisherProjection.m*)

   Classificazione Classificatore lineare binario (*test_feature_binary.m*)

   In questa configurazione un algoritmo genetico seleziona tre sottoinsiemi delle 255 feature originali (765 geni) da usare nel corso dei diversi stadi della classificazione binaria e l'ordine con cui effettuare le partizioni binarie dello spazio delle feature (2 geni); le feature scelte sono poi proiettate dall'algoritmo LDA e infine classificate in quattro classi per mezzo di partizioni binarie dello spazio delle feature.

10. Classificazione lineare multiclasse con algoritmo genetico (*script_test_ga_var_interclass.m*):

    Feature selection GA (*genetic_algorithm_var_interclass.m*)

    Feature projection LDA (*LDAFisherProjection.m*)

    Classificazione Classificatore lineare multiclasse (*test_feature.m*)

    In questa configurazione un algoritmo genetico seleziona un sottoinsieme delle 255 feature originali (255 geni) tale da massimizzare la varianza interclasse in seguito alla classificazione; le feature scelte sono poi proiettate dall'algoritmo LDA e infine classificate.

11. Classificazione lineare multiclasse con algoritmo genetico (*script_test_ga_var_intertrial.m*):

    Feature selection GA (*genetic_algorithm_var_intertrial.m*)

    Feature projection LDA (*LDAFisherProjection.m*)

    Classificazione Classificatore lineare multiclasse (*test_feature.m*)

    In questa configurazione un algoritmo genetico seleziona un sottoinsieme delle 255 feature originali (255 geni) tale da minimizzare la varianza intraclasse in seguito alla classificazione; le feature scelte sono poi proiettate dall'algoritmo LDA e infine classificate.

12. Classificazione lineare multiclasse con algoritmo genetico (*script_test_ga_C3.m*):

    Feature selection GA (*genetic_algorithm_C3.m*)

    Feature projection LDA (*LDAFisherProjection.m*)

    Classificazione Classificatore lineare multiclasse (*test_feature.m*)

    In questa configurazione un algoritmo genetico seleziona un sottoinsieme delle 255 feature originali (255 geni) tale da massimizzare l'accuratezza media della classe 3, ovvero quella classe che nel corso degli esperimenti precedenti è sempre risultata la classe meno discriminabile; le feature scelte sono poi proiettate dall'algoritmo LDA ed infine classificate.

13. Classificazione lineare binario con algoritmo genetico (*script_test_PCA.m*):

    Feature selection -

    Feature projection PCA (*PCA.m*)

    Classificazione Classificatore lineare multiclasse (*test_feature_PCA.m*)

    In questa configurazione priva di feature selection, le feature vengono proiettate dall'algoritmo PCA e quindi classificate in quattro classi.

14. Classificazione lineare multiclasse con classificatore quadratico (*script_test_quadratic.m*):

    Feature selection -

    Feature projection LDA (*LDAFisherProjection.m*)

    Classificazione Classificatore quadratico (*test_feature_quadratic.m*)

    In questa configurazione priva di feature selection, le feature sono proiettate dall'algoritmo LDA e quindi classificate da un classificatore quadratico.

15. Classificazione lineare multiclasse con algoritmo genetico e classificatore quadratico
    (*script_test_ga_quadratic.m*):

    Feature selection GA (*genetic_algorithm_quadratic.m*)

    Feature projection LDA (*LDAFisherProjection.m*)

    Classificazione Classificatore quadratico (*test_feature_quadratic.m*)

    In questa configurazione un algoritmo genetico seleziona un sottoinsieme delle 255 feature originali (255 geni); le feature scelte sono poi proiettate dall'algoritmo LDA e infine classificate da un classificatore quadratico.

16. Classificazione lineare multiclasse con classificatore KNN
    (*script_test_KNN.m*):

    Feature selection -

    Feature projection LDA (*LDAFisherProjection.m*)

    Classificazione Classificatore KNN (*test_feature_KNN.m*)

    In questa configurazione priva di feature selection, le feature sono proiettate dall'algoritmo LDA e quindi classificate da un classificatore k-nearest neighbours; un semplice algoritmo iterativo è utilizzato per determinare i parametri del classifcatore KNN.

17. Classificazione lineare multiclasse con algoritmo genetico e classificatore KNN
    (*script_test_ga_KNN.m*):

    Feature selection GA (*genetic_algorithm_KNN.m*)

    Feature projection LDA (*LDAFisherProjection.m*)

    Classificazione Classificatore KNN (*test_feature_KNN.m*)

    In questa configurazione un algoritmo genetico seleziona un sottoinsieme delle 255 feature originali (255 geni); le feature scelte sono poi proiettate dall'algoritmo LDA e infine classificate da un classificatore k-nearest neighbours; un semplice algoritmo iterativo è utilizzato per determinare i parametri del classifcatore KNN.

18. Classificazione lineare multiclasse con classificatore SVM (*script_test_SVM_RBF.m*):

    Feature selection -

    Feature projection LDA (*LDAFisherProjection.m*)

    Classificazione Classificatore SVM_RBF (*test_feature_SVM_RBF.m*)

    In questa configurazione priva di feature selection, le feature sono proiettate dall'algoritmo LDA e quindi classificate da un classificatore a support vector machine con radial basis functions; un semplice algoritmo iterativo è utilizzato per determinare i parametri del classifcatore SVM_RBF.

19. Classificazione lineare multiclasse con classificatore SVM (*script_test_ga_SVM_RBF.m*):

    Feature selection GA (*genetic_algorithm_SVM_RBF.m*)

    Feature projection LDA (*LDAFisherProjection.m*)

    Classificazione Classificatore SVM_RBF (*test_feature_SVM_RBF.m*)

    In questa configurazione un algoritmo genetico seleziona un sottoinsieme delle 255 feature originali (255 geni); le feature scelte sono poi proiettate dall'algoritmo LDA e infine classificate da un classificatore a support vector machine con radial basis functions; un semplice algoritmo iterativo è utilizzato per determinare i parametri del classifcatore SVM_RBF.

Le Figure 6.3 e 6.4 offrono un'immagine sinottica dei singoli algoritmi utilizzati in ciascun test.

Nella Tabella 6.4 sono riportati i risultati delle performance calcolate sui set di dati ottenuti durante le sessioni di motor imagery senza feedback; per ogni soggetto e per ogni classificatore è riportata sia l'accuratezza media globale sia, tra parentesi, la deviazione standard dell'accuratezza media globale.

6.4.4 Validazione statistica delle performance dei diversi classificatori

Per la validazione statistica dei risultati è stato necessario eseguire un set di test statistici per il confronto delle performance di un insieme di classificatori su un insieme di set di dati; a questo scopo è stata implementata la serie di test statistici suggeriti da Demsar in [20].

Sono state prese in considerazione le performance di 17 classificatori[3] su 5 set di dati, utilizzando come misura della performance di un classificatore l'accuratezza dello stesso (vedi Tabella 6.4 per i dati grezzi).

[3]Sono stati esclusi a priori dai test il classificatore ad albero (*script_test_tree.m*) e il classificatore lineare con proiezione PCA (*script_test_PCA.m*). Dal momento che si desiderava valutare la presenza di differenze significative tra classificatori le cui performance potevano inizialmente apparire simili, si è deciso di non considerare nei test i due classificatori suddetti a causa delle loro performance notevolmente inferiori a qualsiasi altro classificatore.

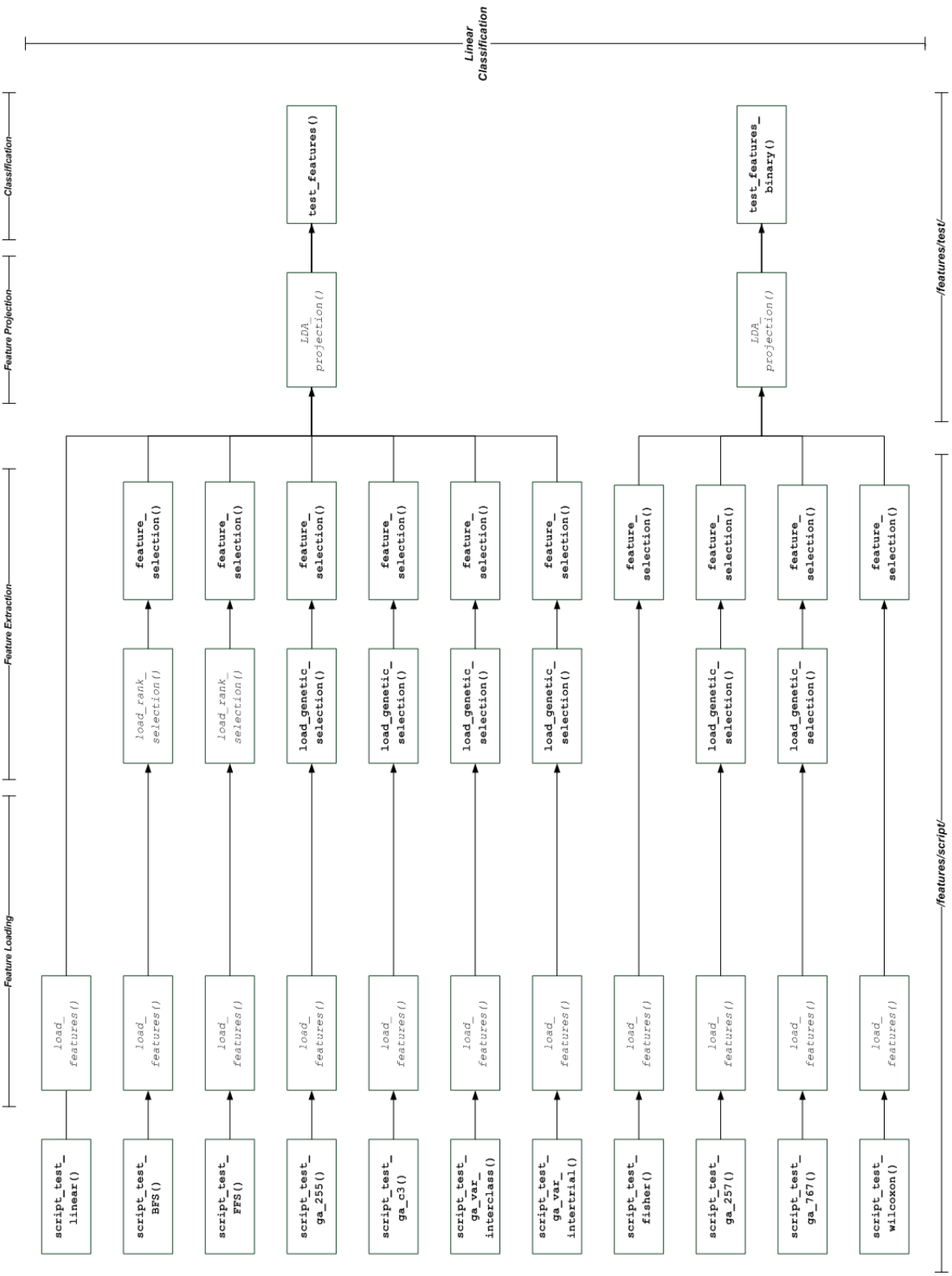

Figura 6.3: Schema a blocchi degli step e degli script Matlab per testare le performance di differenti classificatori.

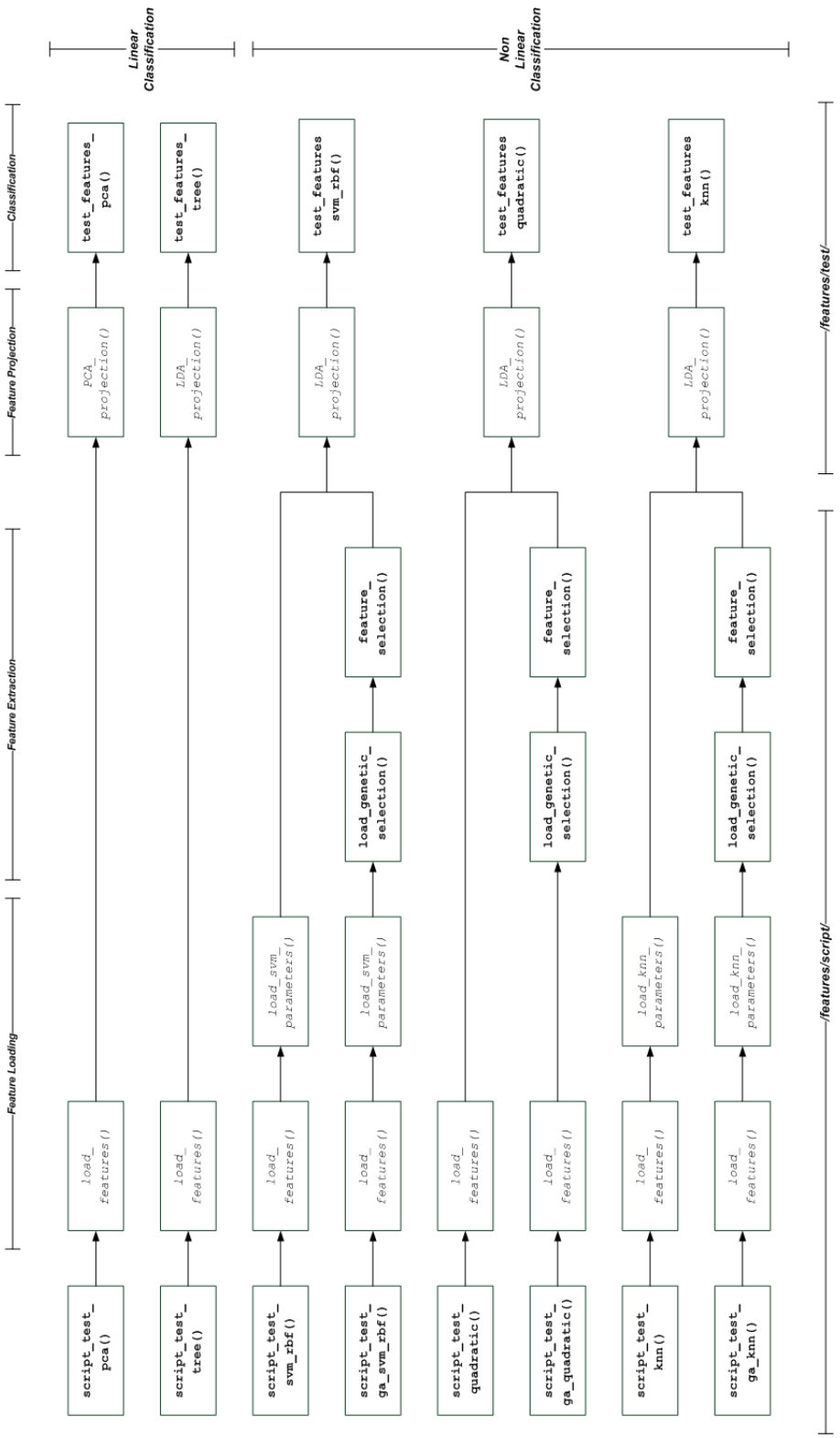

Figura 6.4: Schema a blocchi degli step e degli script Matlab per testare le performance di differenti classificatori.

	FA	PC	TD	SM	FZ
1. ***script_test_linear.m***	**0.3996** (±0.0644)	**0.8277** (±0.0435)	**0.5640** (±0.0220)	**0.6462** (±0.0484)	**0.4777** (±0.0680)
2. ***script_test_fisher.m***	**0.3717** (±0.0458)	**0.8156** (±0.0615)	**0.5535** (±0.0191)	**0.6256** (±0.0658)	**0.4710** (±0.0504)
3. ***script_test_wilcoxon.m***	**0.3817** (±0.0571)	**0.8187** (±0.0595)	**0.5982** (±0.0161)	**0.6222** (±0.0617)	**0.4795** (±0.0436)
4. ***script_test_tree.m***	**0.3354** (±0.0165)	**0.6978** (±0.0566)	**0.4970** (±0.0158)	**0.5413** (±0.0468)	**0.3701** (±0.0443)
5. ***script_test_FFS.m***	**0.3996** (±0.0527)	**0.8170** (±0.0410)	**0.6131** (±0.0441)	**0.6663** (±0.0413)	**0.4558** (±0.0426)
6. ***script_test_BFS.m***	**0.4035** (±0.0600)	**0.8241** (±0.0423)	**0.5685** (±0.0186)	**0.6479** (±0.0547)	**0.4830** (±0.0678)
7. ***script_test_ga_255.m***	**0.4604** (±0.0527)	**0.8500** (±0.0426)	**0.6414** (±0.0152)	**0.7015** (±0.0529)	**0.5201** (±0.0567)
8. ***script_test_ga_257.m***	**0.4425** (±0.0580)	**0.8491** (±0.0399)	**0.6510** (±0.0078)	**0.7042** (±0.0470)	**0.5192** (±0.0589)
9. ***script_test_ga_767.m***	**0.4470** (±0.0512)	**0.8554** (±0.0370)	**0.6458** (±0.0329)	**0.7048** (±0.0416)	**0.5188** (±0.0333)
10. ***script_test_ga_var_interclass.m***	**0.4040** (±0.0156)	**0.8170** (±0.0210)	**0.6027** (±0.0009)	**0.6646** (±0.0110)	**0.4781** (±0.0010)
11. ***script_test_ga_var_intertrial.m***	**0.3823** (±0.0011)	**0.8214** (±0.0121)	**0.6116** (±0.0009)	**0.6557** (±0.0124)	**0.4768** (±0.0108)
12. ***script_test_ga_C3.m***	**0.3945** (±0.0482)	**0.8344** (±0.0427)	**0.5841** (±0.0078)	**0.6657** (±0.0421)	**0.4670** (±0.0665)
13. ***script_test_PCA.m***	**0.2573** (±0.0580)	**0.4478** (±0.0577)	**0.3214** (±0.0703)	**0.3772** (±0.1589)	**0.2652** (±0.0646)
14. ***script_test_quadratic.m***	**0.4029** (±0.0618)	**0.8183** (±0.0519)	**0.5685** (±0.0303)	**0.6473** (±0.0469)	**0.4826** (±0.0683)
15. ***script_test_ga_quadratic.m***	**0.4581** (±0.0564)	**0.8509** (±0.0420)	**0.6615** (±0.0034)	**0.6959** (±0.0503)	**0.5250** (±0.0529)
16. ***script_test_KNN.m***	**0.4118** (±0.0550)	**0.8304** (±0.0455)	**0.5885** (±0.0078)	**0.6540** (±0.0480)	**0.4888** (±0.0479)
17. ***script_test_ga_KNN.m***	**0.4509** (±0.0427)	**0.8451** (±0.0450)	**0.6399** (±0.0064)	**0.7009** (±0.0365)	**0.5107** (±0.0550)
18. ***script_test_SVM_RBF.m***	**0.4152** (±0.0604)	**0.8299** (±0.0449)	**0.5893** (±0.0059)	**0.6540** (±0.0517)	**0.4906** (±0.0534)
19. ***script_test_ga_SVM_RBF.m***	**0.4570** (±0.0548)	**0.8473** (±0.0343)	**0.6287** (±0.0229)	**0.6819** (±0.0525)	**0.5143** (±0.0479)

Tabella 6.4: Performance dei diversi classificatori

Il confronto delle performance di due o più classificatori basato sull'uso di differenti set di dati (confronto multi-classificatore multi-dataset) differisce dal tradizionale e ben più noto confronto delle performance di due classificatori su un singolo dataset [20]. Innanzitutto, le differenze nella varianza della performance tra i differenti set di dati sono indipendenti e, possibilmente, generate da cause differenti; in secondo luogo, quando si confrontano tra loro diversi classificatori e per ogni confronto si esegue un test statistico, è importante tenere conto che il rifiuto di un certo numero di ipotesi nulle sarà dovuto al caso. In base a queste considerazioni, non è dunque corretto utilizzare i classici test statistici per il confronto tra due soli classificatori, quali il *paired T-test*, il *test dei segni* o il *test dei segni di Wilcoxon*. In letteratura, uno dei test più comunemente usato per il confronto multi-classificatore multi-dataset è il metodo *ANOVA* e i relativi test post-hoc (in particolare, *test di Tukey* e *test di Dunnett*); tuttavia, l'utilizzo del metodo ANOVA richiede che alcune condizioni particolarmente stringenti siano soddisfatte: la prima è la proprietà di Gaussianità (ovvero, che i set di dati siano estratti da una distribuzione normale); la seconda è la proprietà di sfericità (che presuppone la proprietà di omoschedasticità, ovvero che i set di dati abbiano la stessa varianza). Sfortunatamente, nel caso presente, queste due proprietà non possono essere garantite: la Gaussianità dei set di dati non può essere asserita a priori e il numero di set di dati disponibili risulta troppo piccolo affinchè un test di Gaussianità (e.g., *test di Kolmogorov-Smirnov*) restituisca un risultato significativo e affidabile; la sfericità dei set di dati è stata negata nel momento in cui si è sottolineato che la varianza dei differenti set di dati è indipendente. Scartato il metodo ANOVA, si è deciso quindi di ricorrere all'utilizzo di una serie di test non-parametrici, tra cui il *test di Friedman*, il *test post-hoc di Nemenyi* e i test post-hoc con correzione dell'errore familywise (e.g., *test di Bonferroni-Dunn*, *test di Hommel*, *test di Hochberg* o *test di Holm*).

Nella realizzazione dei seguenti test statistici sono state rispettate le seguenti assunzioni:

Gaussianità la distribuzione dei set di dati NON è stata considerata gaussiana;

Sfericità la varianza dei set di dati NON è stata considerata sferica;

Indipendenza i set di dati sono stati considerati indipendenti;

Commensurabilità i set di dati sono stati considerati commensurabili.

Avendo già spiegato per quale ragione i set di dati non possono essere assunti gaussiani e sferici, restano ora da giustificare le assunzioni di indipendenza e commensurabilità. L'indipendenza dei set di dati è dovuta alla natura stessa degli esperimenti condotti; ciascun set di dati contiene infatti le registrazioni di un singolo soggetto e le sessioni di raccolta dati sono state condotte in maniera separata per ogni utente. La commensurabilità dei set di dati è dovuta al fatto che tutti i soggetti hanno eseguito gli stessi compiti e i risultati sono stati calcolati in maniera analoga.

Test di Friedman

Il primo test eseguito sui dati è il test di Friedman; il test di Friedman è un test statistico eseguito nel confronto multi-classificatore multi-dataset per determinare se tra le performance di tutti i classificatori esistano delle differenze significative; tale test costituisce dunque una verifica preliminare delle performance dei classificatori prima di eseguire ulteriori confronti tra i singoli classificatori.

Il test di Friedman può essere considerato come una versione non-parametrica del test ANOVA, in grado di restituire risultati più accurati della sua controparte parametrica quando le assunzioni del test ANOVA non sono rispettate. La statistica test di Friedman, utilizzando la correzione di Iman e Davenport [20], è:

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2},$$

dove χ_F^2 è la consueta statistica di Friedman:

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4}\right],$$

dove:

$$\begin{aligned} N &= numero\,di\,set\,di\,dati, \\ k &= numero\,di\,classificatori, \\ R_j &= rango\,medio\,del\,j-esimo\,classificatore. \end{aligned}$$

La statistica test F_F si distribuisce come una F-distribuzione con $(k-1)$ e $(k-1)(N-1)$ gradi di libertà.

Le ipotesi, i parametri e il risultato del test sono riassunti di seguito:

H0 NON ci sono differenze significative nelle performance dei classificatori

H1 Ci sono differenze significative nelle performance dei classificatori

Alfa 0.05

Valore critico del test di Friedman 5.844117

Risultato L'F-value del test è 16.178042. Di conseguenza, si rifiuta l'ipotesi nulla. I dati suggeriscono che vi sono delle differenze significative nelle performance dei classificatori.

Test di Nemenyi

Il secondo test eseguito sui dati è il test di Nemenyi; una volta verificato che vi sono differenze significative nelle performance dei differenti classificatori, il test di Nemenyi permette di comparare tra loro i diversi classificatori ed evidenziare tra quali coppie di classificatori esistono differenze significative.

Il test di Nemenyi è un test post-hoc eseguito quando l'ipotesi nulla del test di Friedman è rifiutata; è equivalente al test post-hoc di Tukey nel caso del test parametrico ANOVA.

Nel confronto tra due classificatori A e B, l'ipotesi nulla del test di Nemenyi è rifiutata se:

$$\mid R_A - R_B \mid > q_\alpha \sqrt{\frac{k(k+1)}{6N}},$$

dove:

$$\begin{aligned} N &= numero\,di\,set\,di\,dati, \\ k &= numero\,di\,classificatori, \\ R_j &= rango\,medio\,del\,j-esimo\,classificatore, \\ q_\alpha &= differenza\,critica\,del\,test\,di\,Nemenyi. \end{aligned}$$

La differenza critica del test di Nemenyi q_α è basata su una statistica *Studentized range* e può essere ricavata dalle relative tavole [20]. Le ipotesi, i parametri e il risultato del test sono riassunti di seguito:

H0 NON ci sono differenze significative tra le performance dell'i-esimo e del j-esimo classificatore

H1 Ci sono differenze significative tra le performance dell'i-esimo e del j-esimo classificatore

Alfa 0.05

Valore critico del test di Nemenyi 11.043966

Risultato I risultati dei confronti tra i vari classificatori sono riportati nella Tabella 6.5 (si noti che in Tabella 6.5, a causa di problemi di spazio, sono stati usati dei numeri al posto dei nomi estesi degli algoritmi di feature selection, di feature projection e di classificazione; per trovare la corrispondenza tra questi numeri e i relativi algoritmi si faccia riferimento alla lista numerata nel Paragrafo 6.4.3). Il test di Nemenyi suggerisce che un insieme di classificatori (principalmente i classificatori basati su algoritmi genetici) ha prestazioni significativamente migliori di un classificatore lineare binario che utilizza il test di Fisher per la proiezione delle feature; uno dei precedenti classificatori, il classificatore quadratico che utilizza un algoritmo genetico per la selezione delle feature, offre anche prestazioni significativamente migliori del classificatore che utilizza il test di Wilcoxon

per la selezione delle feature. Tali risultati sembrerebbero indicare che tra i restanti algoritmi non vi siano differenze significative.

Test di Holm

Il terzo test eseguito sui dati è il test di Holm; in questo caso, il test di Holm non è altro che un test di Nemenyi con correzione di Holm.

La correzione di Holm può essere eseguita quando tra i classificatori da confrontare è possibile eleggere uno di questi a classificatore di controllo, rispetto al quale paragonare le performance di tutti gli altri classificatori. Quando questo requisito è soddisfatto la correzione di Holm permette di incrementare la potenza del test statistico.

Di conseguenza il test di Holm è un test che permette di confrontare le prestazioni di un insieme di classificatori in rapporto alle performance di un classificatore di controllo per evidenziare se vi siano classificatori con risultati significativamente migliori del controllo.

Nel presente caso è stato facile scegliere un classificatore di controllo per eseguire il test di Nemenyi con correzione di Holm; la scelta è ricaduta sul primo e più semplice classificatore, il classificatore lineare multiclasse, che non a caso è stato utilizzato come primo elementare classificatore come spiegato nei Paragrafi 6.4.1 e 6.4.2.

Le ipotesi, i parametri e il risultato del test sono riassunti di seguito:

H0 NON ci sono differenze significative tra le performance dell'i-esimo classificatore e il controllo

H1 Ci sono differenze significative tra le performance dell'i-esimo classificatore e il controllo

Alfa 0.05

Risulato I risultati del test di Holm sono riportati in Tabella 6.6. Il nuovo test di Nemenyi migliorato con la correzione di Holm dimostra che se consideriamo come classificatore di riferimento il classificatore lineare multiclasse, allora alcuni dei classificatori studiati che utilizzano algoritmi genetici per la selezione delle feature possono garantire delle performance significativamente migliori del controllo.

6.4.5 Impatto della generazione e della selezione automatica delle feature

Come alternativa alla generazione manuale delle feature basata sullo studio della letteratura, sulla conoscenza a priori del problema e sull'analisi manuale dei dati, si è studiato e implementato un algoritmo genetico originale per la generazione e la selezione automatica delle feature. Tale algoritmo è basato su un algoritmo

	1.	2.	3.	5.	6.	7.	8.	9.	10.	11.	12.	14.	15.	16.	17.	18.	19.
1.	-																
2.	3.2	-															
3.	0.2	3.0	-														
5.	1.4	4.6	1.6	-													
6.	1.6	4.8	1.8	0.2	-												
7.	10.6	***13.8***	10.8	9.2	9.0	-											
8.	9.8	***13.0***	10.0	8.4	8.2	0.8	-										
9.	10.4	***13.6***	10.6	9.0	8.8	0.2	0.6	-									
10.	2.2	5.4	2.4	0.8	0.6	8.4	7.6	8.2	-								
11.	1.4	4.6	1.6	0.0	0.2	9.2	8.4	9.0	0.8	-							
12.	1.6	4.8	1.8	0.2	0.0	9.0	8.2	8.8	0.6	0.2	-						
14.	0.6	3.8	0.8	0.8	1.0	10.0	9.2	9.8	1.6	0.8	1.0	-					
15.	11.0	***14.2***	***11.2***	9.6	9.4	0.4	1.2	0.6	8.8	9.6	9.4	10.4	-				
16.	3.8	7.0	4.0	2.4	2.2	6.8	6.0	6.6	1.6	2.4	2.2	3.2	7.2	-			
17.	8.2	***11.4***	8.4	6.8	6.6	2.4	1.6	2.2	6.0	6.8	6.6	7.6	2.8	4.4	-		
18.	4.0	7.2	4.2	2.6	2.4	6.6	5.8	6.4	1.8	2.6	2.4	3.4	7.0	0.2	4.2	-	
19.	8.2	***11.4***	8.4	6.8	6.6	2.4	1.6	2.2	6.0	6.8	6.6	7.6	2.8	4.4	0.0	4.2	-

Tabella 6.5: Risultati del test di Nemenyi

	p-value	adjusted-alpha
15. *script_test_ga_quadratic.m*	0.000573	0.004167
7. *script_test_ga_255.m*	0.000903	0.010000
9. *script_test_ga_767.m*	0.001128	0.007143
8. *script_test_ga_257.m*	0.002151	0.008333
17. *script_test_ga_KNN.m*	0.010243	0.003571
19. *script_test_ga_SVM_RBF.m*	-	-
18. *script_test_SVM_RBF.m*	-	-
16. *script_test_KNN.m*	-	-
10. *script_test_ga_var_interclass.m*	-	-
6. *script_test_BFS.m*	-	-
12. *script_test_ga_C3.m*	-	-
5. *script_test_FFS.m*	-	-
11. *script_test_ga_var_intertrial.m*	-	-
14. *script_test_quadratic.m*	-	-
3. *script_test_wilcoxon.m*	-	-
2. *script_test_fisher.m*	-	-

Tabella 6.6: Risultati del test di Holm usando *script_test_linear* come controllo

genetico che genera delle feature e ricerca nello spazio di queste possibili feature l'insieme ottimale per la futura classificazione.

Lo spazio delle feature è costituito da un ampio insieme di feature, estratte e generate a partire dai dati grezzi, e suddivise in tre grandi categorie:

Feature base si tratta di feature elementari, molte delle quali rientrano tra le feature generate manualmente; tali feature corrispondono ai valori dello spettro registrati in uno dei 6 possibili intervalli di tempo (0:1 secondi, 1:2 secondi, 2:3 secondi, 3:4 secondi, 4:5 secondi, 5:6 secondi), in una delle 17 possibili frequenze (le frequenze unitarie da 8 Hz a 24 Hz), da uno dei 10 possibili canali dell'elettroencefalografo (i canali che circondano C3 e C4); il totale delle feature base è 1020.

Feature finali si tratta di feature calcolate esclusivamente alla fine di un singolo task; tali feature corrispondono al valore dello spettro calcolato utilizzando tutti i dati raccolti in un intervallo di 6 secondi, in una delle 17 possibili frequenze (le frequenze unitarie da 8 Hz a 24 Hz), da uno dei 10 possibili canali dell'elettroencefalografo (i canali che circondano C3 e C4); il totale delle feature finali è 170.

Feature avanzate si tratta di feature artificiali ottenute dalle valutazioni e dalle combinazioni di altre feature; tali feature corrispondono al valore calcolato per mezzo di un preciso operatore (differenza, rapporto, inverso, derivata), in uno dei 6 possibili intervalli di tempo (0:1 secondi, 1:2 secondi, 2:3 secondi, 3:4 secondi, 4:5 secondi, 5:6 secondi), in una delle 17 possibili frequenze (le frequenze unitarie da 8 Hz a 24 Hz), da uno dei 10 possibili

canali dell'elettroencefalografo (i canali che circondano C3 e C4); il totale delle feature avanzate è 680.

Il totale di tutte le feature considerate per la generazione automatica delle feature è quindi 1870.

L'insieme di tali feature è analizzato da un'algoritmo creato specificatamente per la selezione di feature. Un algoritmo genetico che si muove in uno spazio a 1870 dimensioni pone ovvi problemi di complessità di calcolo; se a questo si aggiunge il tempo richiesto dalla proiezione LDA e dalla classificazione per poter valutare la funzione di fitness, il problema rischia di diventare intrattabile in termini di tempo. Considerati questi problemi è stato progettato un algoritmo genetico vincolato che permetta di selezionare un sottoinsieme ottimo di feature nel rispetto di alcune condizioni sulla cardinalità di tali insiemi. I vincoli sul numero di feature selezionabili limitano il numero di combinazioni di feature selezionabili (e conseguentemente riducono lo spazio di ricerca), diminuiscono il tempo di calcolo di proiezione LDA e classificazione, oltre a garantire che l'insieme finale di feature abbia una dimensione contenuta. Quest'ultima proprietà è particolarmente utile per evitare di generare insiemi di feature troppo numerosi, che assicurano performance solo leggermente superiori rispetto a un sottoinsieme proprio e che risulterebbero inutilizzabili online. Lo svantaggio principale di questo algoritmo genetico vincolato è che se l'insieme ottimo è dato da un numero di feature maggiori di quelle consentite dal vincolo, tale ottimo risulta irragiungibile; tuttavia, considerando che comunque non è mai garantito che un algoritmo genetico riesca a raggiungere l'ottimo, le limitazioni imposte risultano assolutamente accettabili dal momento che permettono l'esecuzione di un algoritmo che altrimenti richiederebbe tempi di esecuzione troppo lunghi.

L'algoritmo genetico vincolato implementato è sinteticamente delineato di seguito:

Codifica stringa binaria di *1870* elementi.

Dimensione della popolazione p individui.

Condizione di terminazione t iterazioni senza miglioramenti.

Vincolo selezione di un massimo di f feature. Il vincolo dell'algoritmo genetico è, per la verità, un vincolo elastico; f è, idealmente, il numero massimo di feature desiderate; tuttavia è possibile per l'algoritmo genetico selezionare un numero di feature leggermente superiore a f se questo implica delle performance sensibilmente migliori; questo tradeoff è ottenuto penalizzando la fitness di un individuo in maniera proporzionale al numero di feature selezionate oltre f.

Algoritmo per la creazione della popolazione la popolazione iniziale è costituita da una matrice binaria $p \times 1870$; ogni riga rappresenta un individuo e ogni colonna rappresenta una feature. Un valore uguale a uno nella cella *(i, j)* significa che l'individuo i ha selezionato la feature j come parte della soluzione; un valore pari a 0 significa che la relativa feature non è

stata selezionata. In base al vincolo imposto, ogni riga può contenere al più f uno. Per garantire questo vincolo l'algoritmo per la creazione della popolazione è stato implementato secondo il seguente pseudocodice:

```
genera una matrice di zeri px1870
foreach(individuo):
    genera f interi casuali tra 1 e 1870;
    individuo(r) = 1;
end
```

Questo algoritmo garantisce che il numero di uno sia al più f. La probabilità $P(f)$ di avere esattamente f uno può essere calcolata come la probabilità di selezionare a ogni iterazione una feature che non sia già stata selezionata.

$$P(f) = \frac{1870!}{1870^f (1870 - f)!}$$

Di conseguenza, per ciascun individuo della popolazione, c'è una probabilità pari a $1 - P(f)$ che questo contenga un numero minore di p feature.

Algoritmo dell'operatore di mutazione l'operatore di mutazione è stato implementato in maniera che, nel corso dell'esecuzione, il numero di feature rimanga costante. Sia m la probabilità di avere una mutazione, allora lo pseudocodice dell'operatore di mutazione è il seguente:

```
foreach(individuo)
    genera un reale casuale x tra 0 e 1;
    if(x < m)
        genera un intero casuale y tra 1 e 1870;
        genera un reale casuale z tra 0 e 1;
        if(z < (f/1870))
            individuo(y) = 0;
        else
            individuo(y) = 1;
        end
    end
end
```

In altre parole, una volta che la decisione di mutare un gene di un individuo è stata presa, l'algoritmo garantisce che nel corso dell'esecuzione il numero di elementi convertiti a uno o a zero sia bilanciato. Di conseguenza l'effettiva probabilità di mutazione è leggermente inferiore a m. Diversamente da un tradizionale operatore di mutazione, il presente operatore evita che il numero dei geni posti a uno aumenti col tempo violando il vincolo imposto da f.

Algoritmo dell'operatore di crossover l'operatore di crossover è stato implementato in maniera che, nel corso dell'esecuzione, il numero di feature

rimanga costante. Per garantire questo vincolo, l'algoritmo esegue un crossover a più punti tra due individui; la distanza tra i punti di crossover è scelta in modo che, in media, ciascun segmento di un individuo generato dal crossover contenga un singolo uno. Lo pseudocodice dell'algoritmo è il seguente:

```
foreach(coppia di individui)
    calcola la lunghezza del segmento s=floor(1870/f);
    genera un intero casuale o tra 0 ed s;
    genera un array x di zeri 1x1870;
    copia individuo1(1:o) in x(1:o);
    copia individuo2(o:o+s) in x(o:o+s);
    copia individuo1(o+s:o+2s) in x(o+s:o+2s);
    ...
end
```

In altre parole, i due individui coinvolti nel crossover sono divisi in f segmenti, ciascuno dei quali contenente, in media, un singolo uno; questi segmenti sono quindi riassemblati per generare un nuovo individuo. Attraverso questo operatore di crossover, l'algoritmo genetico è quindi in grado di esplorare l'intero spazio delle soluzioni e, nel contempo, evita di generare soluzioni inaccettabili che contengano troppi uno.

Funzione di fitness La funzione utilizzata per calcolare la fitness di un individuo è basata sul calcolo della performance della classificazione lineare eseguita usando le feature selezionate dall'individuo; il valore della performance è dato dall'accuratezza della classificazione (un valore negativo compreso tra *0* e *-1*, dove *-1* rappresenta una perfetta classificazione e *0* una classificazione assolutamente casuale[4]) più un fattore di penalizzazione (un valore positivo compreso tra *0* e *1*). Assunta x come la differenza tra il numero massimo di feature accettabile ed *f*, lo pseudocodice per il calcolo della funzione di fitness è il seguente:

```
foreach(individuo)
    calcola l'accuratezza della classificazione c;
    if(|individuo| > f)
        penalità p = ((|individuo|-f)/x)^2;
    else
        penalità p = 0;
    end
    fitness f = c+p;
end
```

In altre parole, la penalità è *0* se il numero delle feature selezionate è inferiore a f; altrimenti il valore della penalità cresce quadraticamente tra

[4]Si ricordi che l'algoritmo genetico ha come obiettivo la minimizzazione della fitness della popolazione.

popolazione	iterazioni	feature	accuratezza media
100	300	300	0.6839
100	500	250	0.6850
100	500	300	0.7029
200	500	250	0.7103
200	500	300	0.7189
300	500	250	0.7275
300	500	300	0.7171
500	500	250	0.7321

Tabella 6.7: Performance dell'algoritmo di selezione automatica delle feature.

> f e $f+x$; quando $f+x$ feature sono selezionate, il valore della penalità è 1; di conseguenza, la classificazione ottenuta con un insieme di feature che viola il limite massimo di feature selezionabili ha un valore di fitness sufficientemente elevato da non essere considerata una possibile soluzione candidata.

Utilizzando questo algoritmo, si è prima creato in maniera automatica, senza nessun intervento da parte di utenti esperti, un ampio insieme di feature; quindi si è proceduto all'esplorazione di questo spazio e alla riduzione dell'insieme di feature.

La Tabella 6.7 riporta l'accuratezza media ottenuta utilizzando l'algoritmo di generazione e selezione automatica delle feature al variare dei parametri di popolazione p, iterazioni t e feature f dell'algoritmo genetico; l'accuratezza è stata calcolata usando l'insieme di feature prodotto dall'algoritmo di generazione e selezione automatica delle feature e classificandolo per mezzo di un classificatore lineare; il set originale di dati su cui è stato eseguito questo test sono i dati ottenuti durante le sessioni di motor imagery senza feedback per il soggetto PC.

Capitolo 7

Discussione

In questo capitolo si discutono i risultati degli esperimenti riportati nel capitolo precedente e se ne propone un'interpretazione coerentemente con gli obiettivi che ci eravamo prefissati. Si propongono infine alcuni spunti per quelli che potrebbero essere gli sviluppi futuri dello stesso progetto.

7.1 Discussione

I risultati ottenuti durante lo sviluppo di questa tesi hanno confermato l'importanza degli algoritmi di generazione delle feature, di feature selection, di feature projection e di classificazione rispetto alle prestazioni di un'interfaccia cervello-computer.

I risultati conseguiti nell'analisi delle performance delle sessioni di motor imagery senza feedback (vedi Paragrafo 6.4.1) hanno innanzitutto dimostrato la possibilità di realizzare un'interfaccia cervello-computer funzionante utilizzando anche semplici algoritmi di analisi del segnale, come la proiezione LDA e la classificazione lineare; se si escludono i risultati del soggetto FA (le cui basse prestazioni potrebbero, secondo la letteratura, essere dovute al fatto di essere mancino e richiedere dunque uno studio particolare), tutti i soggetti hanno raggiunto un controllo prossimo o superiore allo 0.5, con un massimo a 0.8, in un sistema che discriminava tra 4 classi; ciò sembrerebbe confermare che una corretta combinazione di training e di algoritmi di analisi del segnale possono garantire una buona accuratezza nell'uso dell'interfaccia cervello-computer.

I risultati raggiunti nell'analisi delle performance delle sessioni di motor imagery con feedback (vedi Paragrafo 6.4.2) sembrerebbero confermare la possibilità di controllo dell'interfaccia cervello-computer, con valori di accuratezza dell'interfaccia cervello-computer che oscillano ancora una volta tra 0.5 e 0.7, anche con l'aggiunta di un feedback. E' interessante notare come l'introduzione del feedback non abbia influito univocamente sul controllo degli utenti: un utente su tre ha migliorato le sue prestazioni, mentre i restanti due hanno visto le loro performance rimanere costanti o diminuire. Questo risultato è dopotutto in li-

nea con la letteratura, secondo cui il contributo del feedback varia da soggetto a soggetto, in alcuni casi in maniera positiva, in altri in maniera negativa.

I risultati ottenuti nell'analisi delle prestazioni dei diversi classificatori (vedi Paragrafo 6.4.3) mostrano in maniera chiara quanto importante sia il contributo degli algoritmi di analisi del segnale rispetto alle performance finali; considerando il solo soggetto PC (ovvero il soggetto che ha ottenuto in generale le prestazioni migliori) è possibile constatare come al variare degli algoritmi di analisi del segnale la sua accuratezza vari da un minimo di 0.45 a un massimo di 0.85. Anche gli altri soggetti, pur non raggiungendo gli stessi risultati di PC, mostrano uno spettro di variabilità analoga. In generale, coerentemente con quanto espresso in letteratura [56], tecniche di feature projection elementari (come la LDA) e metodi di classificazione semplici (come la classificazione lineare) sono in grado di garantire risultati accettabili; tuttavia, data la complessità del problema affrontato, è stato dimostrato che tecniche di feature selection e classificazione più elaborate e raffinate possono apportare un contributo significativo al risultato finale.

Per quanto riguarda gli algoritmi di feature selection, gli algoritmi genetici hanno garantito gli incrementi delle performance migliori; a differenza di altri metodi di feature selection basati su metriche sintetiche di selezione differenti, gli algoritmi genetici hanno potuto esplorare l'intero spazio delle feature e selezionare le feature che sono risultate migliori relativamente all'obiettivo di classificazione scelto; l'unico lato negativo di tali algoritmi genetici è che insieme a un aumento delle performance si accompagna anche un incremento dei tempi di computazione; quest'ultimo è tuttavia un problema di secondaria importanza dal momento che la selezione delle feature per mezzo di un algoritmo genetico è comunque un processo da eseguire una volta soltanto e offline.

Per quanto riguarda gli algoritmi di classificazione, sebbene buoni risultati possano essere ottenuti per mezzo di un classificatore lineare, l'utilizzo di classificatori non lineari come i classificatori quadratici hanno permesso un lieve aumento delle prestazioni; questo suggerisce che il problema della separabilità delle classi non è un problema di semplice risoluzione e che spesso le classi identificate tendono a sovrapporsi (questo è evidente analizzando la Tabella 6.2: la bassa accuratezza della terza classe, corrispondente all'immaginazione del movimento di entrambe le mani, è dovuta alla sua naturale sovrapposizione con le prime due classi, corrispondenti rispettivamente all'immaginazione del movimento della mano sinistra e della mano destra; viceversa le alte prestazioni della quarta classe, corrispondente al movimento dei piedi, sono dovute all'indipendenza tra l'immaginazione del movimento dei piedi e l'immaginazione del movimento delle mani). E' quindi sensato aspettarsi che un algoritmo di classificazione non lineare, in grado di tracciare superfici di separazione più complesse possa contribuire a una migliore classificazione. In definitiva, la scelta e lo sviluppo di determinati algoritmi di generazione delle feature, di feature selection, di feature projection e di classificazione è una decisione critica nello sviluppo di un'interfaccia cervello-computer, in grado di determinare sensibilmente le prestazioni finali dell'interfaccia cervello-computer.

I risultati conseguiti nella validazione statistica delle performance dei clas-

sificatori (vedi Paragrafo 6.4.4) hanno poi confermato l'esistenza di differenze significative tra i vari algoritmi di analisi del segnale implementati; in altre parole, è stato possibile confermare che le differenze registrate tra gli algoritmi implementati sono dovute a reali differenze nelle prestazioni degli algoritmi e non al semplice caso. Il test di Nemenyi e il test di Holm suggeriscono che le performance ottenute utilizzando metodi di analisi del segnale elementare come quelli descritti nel Paragrafo 6.4.1, possono essere significativamente migliorati adottando nuovi algoritmi; in particolare, gli algoritmi genetici utilizzati per la feature selection, combinati con classificatori quadratici, lineari o KNN permettono di incrementare significativamente le prestazioni.

I risultati ottenuti utilizzando l'algoritmo di selezione automatica delle feature (vedi Paragrafo 6.4.5) sono purtroppo risultati inferiori alle prestazioni ottenute per mezzo della selezione manuale delle feature. Da una parte questo sottolinea l'impatto positivo che una scelta accurata e consapevole delle feature può avere nel determinare l'accuratezza finale; dall'altra, suggeriscono comunque la possibilità di sviluppare e affinare algoritmi di selezione automatica delle feature in grado di generare insiemi di feature tali da garantire buone performance. Si noti inoltre il progressivo miglioramento dei risultati dell'algoritmo di selezione automatica delle feature all'aumentare dei valori dei parametri dell'algoritmo genetico; si tratta di un fenomeno tipico degli algoritmi genetici: consentendo all'algoritmo genetico di esplorare lo spazio delle feature più estesamente o per più tempo, è possibile che l'algoritmo genetico si muova, abbandonando la soluzione trovata in un massimo locale per una nuova, migliore soluzione in un nuovo massimo; lo svantaggio nell'aumento dei valori dei parametri dell'algoritmo di selezione automatica delle feature è il conseguente aumento dei tempi di computazione che possono protrarsi molto a lungo.

7.2 Sviluppi futuri

I risultati di questa tesi confermano l'importanza e la rilevanza degli algoritmi di generazione delle feature, di feature projection, di feature selection e di classificazione per lo sviluppo delle interfacce cervello-computer. D'altra parte il campo degli algoritmi di analisi dei segnali è talmente vasto che gli algoritmi considerati in questa tesi sono solo un piccolo sottoinsieme di tutte le possibili tecniche sviluppate per estrarre informazione da un segnale; un'interessante futura direzione di ricerca è dunque costituita dallo sviluppo e dalla sperimentazione di nuovi e diversi algoritmi in grado di garantire performance migliori. Questa via è stata poi effettivamente perseguita come spiegato sommariamente nel Paragrafo 7.2.1.

Di grande interesse pratico è inoltre lo studio di algoritmi di generazione e selezione automatica delle feature. L'algoritmo che abbiamo proposto in questa tesi raggiunge risultati accettabili, ma è indubbio che sia possibile migliorarlo. Tali algoritmi risulterebbero particolarmente utili in quanto consentirebbero di automatizzare e velocizzare il processo di selezione delle feature, rendendolo indipendente da operatori umani esperti.

7.2.1 Ulteriori sviluppi durante lo stage presso la NTT

Grazie allo sviluppo di questa tesi e alle conoscenze acquisite durante la ricerca condotta presso il Politecnico di Milano, mi è stato possibile prendere parte a uno stage di otto mesi presso i laboratori *Basic Research Laboratory* della *NTT* a Morinosato, in Giappone. Sono così entrato a far parte del gruppo di ricerca di *Material Science* e ho avuto l'opportunità di lavorare fianco a fianco con ricercatori provenienti da diversi campi di specializzazione. Sotto la supervisione del Dr. Akiyoshi Shimada, ho lavorato a un progetto per lo sviluppo di un modello originale di interfaccia cervello-computer. Uno dei miei compiti principali è stato quello di studiare e analizzare i segnali acquisiti durante gli esperimenti in laboratorio; ho avuto così modo di sfruttare l'esperienza accumulata durante lo sviluppo di questa tesi e di approfondire la mia ricerca riguardo agli algoritmi di analisi dei segnali. In particolare ho potuto studiare e sviluppare diversi tipi di reti neurali per il clustering e la classificazione dei dati raccolti; la nostra attenzione si è concentrata soprattutto su una specifica famiglia di reti neurali chiamate ART (Adaptive Resonance Network); questo tipo di reti neurali, proposte per la prima volta nel 1987 da Stephen Grossberg e Gail Carpenter, hanno attratto il nostro interesse non solo per le loro intrinsiche proprietà di accuratezza e robustezza, ma anche per le affascinanti analogie esistenti tra l'architettura di queste reti neurali artificiali e il funzionamento stesso del cervello umano; diverse tipologie di reti ART sono state sviluppate e testate nel corso di questa ricerca, tra cui *ART1* [11], *ARTMAP* [13], *FuzzyART* [14], *FuzzyARTMAP* [12], *DistributedART[10]* ed *EnsembleART*.

Appendice A

Dimostrazioni

A.1 Determination Coefficient (r^2)

Si riporta in appendice la dimostrazione del'uguaglianza tra la definizione standard di determination coefficient data nel Paragrafo 4.8.2:

$$r^2 = \frac{SS_{exp}}{SS_{tot}} = \frac{\sum_i (f_i - \bar{y})^2}{\sum_i (y_i - \bar{y})^2},$$

e l'implementazione della stessa in BCI2000:

$$r^2 = \frac{\frac{\left(\sum_{i=1}^{N_q} q_i\right)^2}{N_q} + \frac{\left(\sum_{j=1}^{N_r} r_j\right)^2}{N_r} - G}{\sum_{i=1}^{N_q}(q_i)^2 + \sum_{j=1}^{N_r}(r_j)^2 - G}.$$

Siano A e B due vettori di dati del tipo:

$$A = \left[\begin{array}{cccc} y_{11} & y_{12} & \cdots & y_{1n_a} \end{array} \right], \tag{A.1}$$

$$B = \left[\begin{array}{cccc} y_{21} & y_{22} & \cdots & y_{2n_b} \end{array} \right]. \tag{A.2}$$

Si definisca inoltre Y come giustapposizione dei due vettori:

$$Y = \left[\begin{array}{c} A \\ B \end{array} \right] = \left[\begin{array}{cccc} y_{11} & y_{12} & \cdots & y_{1n_a} \\ y_{21} & y_{22} & \cdots & y_{2n_b} \end{array} \right]. \tag{A.3}$$

Si hanno conseguentemente i seguenti valori per le medie:

$$\bar{A} = \frac{\sum_{i=1}^{n_a} A_{1i}}{n_a}, \tag{A.4}$$

$$\bar{B} = \frac{\sum_{i=1}^{n_b} B_{2i}}{n_b}, \tag{A.5}$$

$$\bar{Y} = \frac{\sum_{i=1}^{n_a} A_{1i} + \sum_{i=1}^{n_b} B_{2i}}{n_a + n_b}. \tag{A.6}$$

Infine, si ponga che il valore predetto per ogni classe sia uguale alla media dei valori osservati, ovvero:

$$\hat{y}_{ij} = E\left[y_{ij}\right] = \bar{y}_i, \tag{A.7}$$

quindi:

$$\hat{y}_{1j} = E\left[y_{1j}\right] = \bar{y}_1 = \bar{A}, \tag{A.8}$$
$$\hat{y}_{2j} = E\left[y_{2j}\right] = \bar{y}_2 = \bar{B}. \tag{A.9}$$

Si era inizialmente definito r^2 in (4.1) nel seguente modo:

$$r^2 = \frac{SS_{exp}}{SS_{tot}}.$$

La varianza spiegata (SS_{exp}) era poi stata definita in (4.2) come:

$$SS_{exp} = \sum_i (f_i - \bar{y})^2.$$

Nel nostro caso, questo significa:

$$SS_{exp} = \sum_i (\hat{y}_i - \bar{Y})^2, \tag{A.10}$$

in quanto, con la simbologia appena introdotta, il valore modellato f_i è equivalente al nostro $\hat{y}_i$ definito in (A.7) e, ovviamente, la media dei dati osservati $\bar{y}$ corrisponde alla media $\bar{Y}$ presentata in (A.6).

Ricordando che Y è data dall'unione di due vettori come dalla formula (A.3), per la linearità della sommatoria è possibile riscrivere (A.10) separando i valori appartenenti alla prima classe dai valori appartenenti alla seconda classe:

$$SS_{exp} = \sum_i (\hat{y}_i - \bar{Y})^2 = \sum_{i=1}^{n_a} (\hat{y}_{1i} - \bar{Y})^2 + \sum_{i=1}^{n_b} (\hat{y}_{2i} - \bar{Y})^2, \tag{A.11}$$

ma dalla (A.8) e dalla (A.9) sappiamo che i valori predetti per le due classi non sono altro che $\bar{A}$ e $\bar{B}$:

$$\sum_{i=1}^{n_a} (\hat{y}_{1i} - \bar{Y})^2 + \sum_{i=1}^{n_b} (\hat{y}_{2i} - \bar{Y})^2 = \sum_{i=1}^{n_a} (\bar{A} - \bar{Y})^2 + \sum_{i=1}^{n_b} (\bar{B} - \bar{Y})^2; \tag{A.12}$$

tuttavia $\bar{A}$, $\bar{B}$ e $\bar{Y}$ si mantengono costanti indipendentemente dai campioni, quindi:

$$\sum_{i=1}^{n_a} (\bar{A} - \bar{Y})^2 + \sum_{i=1}^{n_b} (\bar{B} - \bar{Y})^2 = n_a(\bar{A} - \bar{Y})^2 + n_b(\bar{B} - \bar{Y})^2. \tag{A.13}$$

Svolgendo i quadrati:

$$n_a(\bar{A}-\bar{Y})^2+n_b(\bar{B}-\bar{Y})^2 = n_a\bar{A}^2+n_b\bar{B}^2+(n_a+n_b)\bar{Y}^2+2\bar{Y}(n_a\bar{A}+n_b\bar{B}). \quad (A.14)$$

Prima di proseguire, si dimostra sommariamente usando le formule (A.4), (A.5) e (A.6) la seguente uguaglianza:

$$\bar{Y}(n_a\bar{A}+n_b\bar{B}) = \left(\frac{\sum_{i=1}^{n_a} A_{1i}+\sum_{i=1}^{n_b} B_{2i}}{n_a+n_b}\right)\left(\sum_{i=1}^{n_a} A_{1i}+\sum_{i=1}^{n_b} B_{2i}\right)\left(\frac{n_a+n_b}{n_a+n_b}\right) =$$

$$= \frac{\left(\sum_{i=1}^{n_a} A_{1i}+\sum_{i=1}^{n_b} B_{2i}\right)^2}{(n_a+n_b)^2}(n_a+n_b) = \bar{Y}^2(n_a+n_b); \quad (A.15)$$

sostituendo la formula (A.15) in (A.14) si ottiene:

$$n_a\bar{A}^2+n_b\bar{B}^2+(n_a+n_b)\bar{Y}^2+2\bar{Y}(n_a\bar{A}+n_b\bar{B}) = n_a\bar{A}^2+n_b\bar{B}^2-(n_a+n_b)\bar{Y}^2. \quad (A.16)$$

Riscrivendo il secondo membro in accordo con le formule (A.4), (A.5) e (A.6), si ottiene infine:

$$SS_{exp} = n_a\bar{A}^2+n_b\bar{B}^2-(n_a+n_b)\bar{Y}^2 =$$

$$= \frac{\left(\sum_{i=1}^{n_a} A_{1i}\right)^2}{n_a}+\frac{\left(\sum_{i=1}^{n_b} B_{2i}\right)^2}{n_b}+\frac{\left(\sum_{i=1}^{n_a} A_{1i}+\sum_{i=1}^{n_b} B_{2i}\right)^2}{n_a+n_b}.$$

La varianza totale era invece stata definita in (4.3) come:

$$SS_{tot} = \sum_i (y_i-\bar{y})^2;$$

nel nostro caso, questo significa:

$$SS_{tot} = \sum_{ij} (y_{ij}-\bar{Y})^2, \quad (A.17)$$

in quanto i valori sperimentali del vettore y_i corrispondono ai valori y_{ij} registrati, come definito in (A.3), e, ovviamente, la media dei dati osservati $\bar{y}$ corrisponde alla media $\bar{Y}$ presentata in (A.6).

Ancora una volta, ricordando che Y è data dall'unione di due vettori come dalla formula (A.3), per la linearità della sommatoria è possibile riscrivere la formula (A.17) separando i valori appartenenti alla prima classe dai valori appartenenti alla seconda classe:

$$SS_{tot} = \sum_{ij} (y_{ij}-\bar{Y})^2 = \sum_{i=1}^{n_a} (A_{1i}-\bar{Y})^2+\sum_{i=1}^{n_b} (B_{2i}-\bar{Y})^2; \quad (A.18)$$

svolgendo i quadrati:

$$\sum_{i=1}^{n_a}(A_{1i}-\bar{Y})^2+\sum_{i=1}^{n_b}(B_{2i}-\bar{Y})^2=\sum_{i=1}^{n_a}(A_{1i}^2-2A_{1i}\bar{Y}+\bar{Y}^2)+\sum_{i=1}^{n_b}(B_{2i}^2-2B_{2i}\bar{Y}+\bar{Y}^2); \tag{A.19}$$

sempre per la linearità della sommatoria e ricordando che $\bar{Y}$ si mantiene costante indipendentemente dai campioni, è possibile riscrivere la (A.19) come:

$$\sum_{i=1}^{n_a}(A_{1i}^2-2A_{1i}\bar{Y}+\bar{Y}^2)+\sum_{i=1}^{n_b}(B_{2i}^2-2B_{2i}\bar{Y}+\bar{Y}^2)=$$

$$=\sum_{i=1}^{n_a}A_{1i}^2+\sum_{i=1}^{n_b}B_{2i}^2+n_a\bar{Y}^2+n_b\bar{Y^2}-2\bar{Y}\sum_{i=1}^{n_a}A_{1i}-2\bar{Y}\sum_{i=1}^{n_b}B_{1i}. \tag{A.20}$$

Raccogliendo a fattore comune:

$$\sum_{i=1}^{n_a}A_{1i}^2+\sum_{i=1}^{n_b}B_{2i}^2+n_a\bar{Y}^2+n_b\bar{Y}^2-2\bar{Y}\sum_{i=1}^{n_a}A_{1i}-2\bar{Y}\sum_{i=1}^{n_b}B_{1i}=$$

$$=\sum_{i=1}^{n_a}A_{1i}^2+\sum_{i=1}^{n_b}B_{2i}^2+\bar{Y}^2(n_a+n_b)-2\bar{Y}\left(\sum_{i=1}^{n_a}A_{1i}+\sum_{i=1}^{n_b}B_{1i}\right); \tag{A.21}$$

applicando l'uguaglianza (A.15) si ottiene:

$$\sum_{i=1}^{n_a}A_{1i}^2+\sum_{i=1}^{n_b}B_{2i}^2+\bar{Y}^2(n_a+n_b)-2\bar{Y}\left(\sum_{i=1}^{n_a}A_{1i}+\sum_{i=1}^{n_b}B_{1i}\right)=$$

$$=\sum_{i=1}^{n_a}A_{1i}^2+\sum_{i=1}^{n_b}B_{2i}^2+\bar{Y}(n_a\bar{A}+n_b\bar{B})-2\bar{Y}\left(\sum_{i=1}^{n_a}A_{1i}+\sum_{i=1}^{n_b}B_{1i}\right). \tag{A.22}$$

Tuttavia dalla definizione (A.4) e (A.5) di $\bar{A}$ e $\bar{B}$ segue che:

$$\sum_{i=1}^{n_a}A_{1i}^2+\sum_{i=1}^{n_b}B_{2i}^2+\bar{Y}(n_a\bar{A}+n_b\bar{B})-2\bar{Y}\left(\sum_{i=1}^{n_a}A_{1i}+\sum_{i=1}^{n_b}B_{1i}\right)=$$

$$=\sum_{i=1}^{n_a}A_{1i}^2+\sum_{i=1}^{n_b}B_{2i}^2+\bar{Y}\left(\sum_{i=1}^{n_a}A_{1i}+\sum_{i=1}^{n_b}B_{1i}\right)-2\bar{Y}\left(\sum_{i=1}^{n_a}A_{1i}+\sum_{i=1}^{n_b}B_{1i}\right)=$$

$$=\sum_{i=1}^{n_a}A_{1i}^2+\sum_{i=1}^{n_b}B_{2i}^2-\bar{Y}\left(\sum_{i=1}^{n_a}A_{1i}+\sum_{i=1}^{n_b}B_{1i}\right). \tag{A.23}$$

Infine, sostituendo anche la definizione di $\bar{Y}$ in (A.6) si ottiene:

$$SS_{tot} = \sum_{i=1}^{n_a} A_{1i}^2 + \sum_{i=1}^{n_b} B_{2i}^2 - \bar{Y}\left(\sum_{i=1}^{n_a} A_{1i} + \sum_{i=1}^{n_b} B_{1i}\right) =$$

$$= \sum_{i=1}^{n_a} A_{1i}^2 + \sum_{i=1}^{n_b} B_{2i}^2 - \frac{\left(\sum_{i=1}^{n_a} A_{1i} + \sum_{i=1}^{n_b} B_{1i}\right)^2}{(n_a + n_b)}. \qquad \text{(A.24)}$$

In definitiva si è ottenuto che r^2 può essere scritto come:

$$r^2 = \frac{SS_{exp}}{SS_{tot}} = \frac{\frac{\left(\sum_{i=1}^{n_a} A_{1i}\right)^2}{n_a} + \frac{\left(\sum_{i=1}^{n_b} B_{2i}\right)^2}{n_b} + \frac{\left(\sum_{i=1}^{n_a} A_{1i} + \sum_{i=1}^{n_b} B_{2i}\right)^2}{n_a + n_b}}{\sum_{i=1}^{n_a} A_{1i}^2 + \sum_{i=1}^{n_b} B_{2i}^2 - \frac{\left(\sum_{i=1}^{n_a} A_{1i} + \sum_{i=1}^{n_b} B_{1i}\right)^2}{(n_a + n_b)}}.$$

A questo punto, abbandonando la scrittura utilizzata per comodità in questa dimostrazione e adottando quella usata nel codice *rsqu.m*, risulta evidente che:

$$\begin{aligned} A_{1i} &= q_i, \\ n_a &= N_q, \\ B_{2i} &= r_i, \\ n_b &= N_r. \end{aligned}$$

Imponendo poi:

$$\frac{\left(\sum_{i=1}^{n_a} A_{1i} + \sum_{i=1}^{n_b} B_{1i}\right)^2}{(n_a + n_b)} = \frac{\left(\sum_{i=1}^{N_q} q_i + \sum_{j=1}^{N_r} r_j\right)^2}{N_q + N_r} = G,$$

si ottiene:

$$r^2 = \frac{SS_{exp}}{SS_{tot}} = \frac{\frac{\left(\sum_{i=1}^{n_a} A_{1i}\right)^2}{n_a} + \frac{\left(\sum_{i=1}^{n_b} B_{2i}\right)^2}{n_b} + \frac{\left(\sum_{i=1}^{n_a} A_{1i} + \sum_{i=1}^{n_b} B_{2i}\right)^2}{n_a + n_b}}{\sum_{i=1}^{n_a} A_{1i}^2 + \sum_{i=1}^{n_b} B_{2i}^2 - \frac{\left(\sum_{i=1}^{n_a} A_{1i} + \sum_{i=1}^{n_b} B_{1i}\right)^2}{(n_a + n_b)}} =$$

$$= \frac{\frac{\left(\sum_{i=1}^{N_q} q_i\right)^2}{N_q} + \frac{\left(\sum_{j=1}^{N_r} r_j\right)^2}{N_r} - G}{\sum_{i=1}^{N_q}(q_i)^2 + \sum_{j=1}^{N_r}(r_j)^2 - G}.$$

Si è dimostrata così l'uguaglianza tra la definizione standard di r^2 e la formula adottata in BCI2000 per il calcolo di r^2 dati due vettori monodimensionali di dati. Con lo stesso procedimento è poi possibile generalizzare a casi multidimensionali.

A.2 Analisi in frequenza

Si riportano in appendice alcuni teoremi accompagnati da dimostrazioni della teoria dell'analisi in frequenza; essi sono necessari per comprendere come sia definita la stima spettrale autoregressiva.

A.2.1 Teorema della diseguaglianza di Jensen

Per dimostrare il teorema della diseguaglianza di Jensen, è necessario anzitutto introdurre alcune definizioni riguardo le funzioni concave e convesse [18].

Primo, data una funzione $f(x)$, la funzione $f(x)$ si definisce convessa sull'intervallo (a, b) se:

$$\begin{aligned} \forall x_1\, x_2 &\in (a,b), \\ \forall \lambda \in \Re &, \quad 0 \leq \lambda \leq 1, \\ f(\lambda x_1 + (1-\lambda)x_2 &\leq \lambda f(x_1) + (1-\lambda) f(x_2). \end{aligned} \tag{A.25}$$

Inoltre, se l'uguaglianza è vera solo per $\lambda = 1$ o $\lambda = 0$ $f(x)$, è strettamente convessa.

Secondo, data una funzione $f(x)$ convessa, allora $-f(x)$ è concava.

Terzo, data una una funzione $f(x)$, se $f(x)$ ha una derivata seconda sempre non negativa, allora la funzione è convessa; se $f(x)$ ha una derivata seconda sempre positiva, allora la funzione è strettamente convessa.

Si assuma inoltre che $E[\cdot]$ denoti il valore atteso, che, come noto, è rispettivamente nel caso discreto e nel caso continuo:

$$E[x] = \sum_{x \in S} x p_X(x), \tag{A.26}$$

$$E[x] = \int_{x \in \Re} x f_X(x) dx. \tag{A.27}$$

Poste queste premesse, il teorema della diseguaglianza di Jensen [18] afferma che, data una funzione convessa $f(x)$ e una variabile aleatoria X allora:

$$E[f(X)] \geq f(E[X]). \tag{A.28}$$

Inoltre, se$f(x)$ è strettamente convessa, allora dall'uguaglianza appena enunciata segue che $X = E[x]$ con probabilità 1, ovvero, X è costante.

La validità del teorema della diseguaglianza di Jensen può essere provata per induzione sul numero dei punti di massa. Si consideri una funzione di distribuzione di massa a due valori $p1$ e $p2$; allora la formula (A.28) può essere riscritta come:

$$p_1 f(x_1) + p_2 f(x_2) \geq f(p_1 x_1 + p_2 x_2). \tag{A.29}$$

Quest'ultima formula risulta valida per la definizione stessa di funzione convessa data nella formula (A.25).

Ora, si supponga che il teorema sia valido per funzioni di distribuzione di massa con $k-1$ punti. Si ponga:

$$p'_1 = \frac{p_i}{1-p_k} \;\; \forall 1 \leq i \leq k-1. \tag{A.30}$$

È possibile allora scrivere la seguente uguaglianza:

$$\sum_{i=1}^{k} p_i f(x_i) = p_k f(x_k) + (1-k) \sum_{i=1}^{k-1} p'_i f(x_i). \tag{A.31}$$

Ora, per l'ipotesi di induzione sui $k-1$ punti, si può riscrivere:

$$p_k f(x_k) + (1-p_k) \sum_{i=1}^{k-1} p'_i f(x_i) \geq p_k f(x_k) + (1-p_k) f\left(\sum_{i=1}^{k-1} p'_i x_i \right). \tag{A.32}$$

Applicando la definizione di convessità data nella formula (A.25), segue che:

$$p_k f(x_k) + (1-p_k) f\left(\sum_{i=1}^{k-1} p'_i x_i \right) \geq f\left(p_k x_k + (1-p_k) \sum_{i=1}^{k-1} p'_i x_i \right). \tag{A.33}$$

Infine, sostituendo p' con la definizione data nella formula (A.30), si ha:

$$f\left(p_k x_k + (1-p_k) \sum_{i=1}^{k-1} p'_i x_i \right) = f\left(\sum_{i=1}^{k} p_i x_i \right). \tag{A.34}$$

In definitiva, come volevasi dimostrare, si è ottenuto che:

$$\sum_{i=1}^{k} p_i f(x_i) \geq f\left(\sum_{i=1}^{k} p_i x_i \right). \tag{A.35}$$

Il teorema, ora dimostrato per il caso discreto, può essere esteso al caso continuo per mezzo di argomenti sulla continuità.

A.2.2 Teorema della diseguaglianza dell'informazione

Il teorema della diseguaglianza dell'informazione [18] afferma che, date due funzioni di probabilità di massa $p_X(x)$ e $q_X(x)$ con $x \in \Re$ allora:

$$D(p\|q) \geq 0 \;\; \forall x \in \Re,$$

e

$$D(p\|q) = 0 \iff p_X(x) = q_X(x) \;\; \forall x \in \Re.$$

Infatti, sia S lo spazio campionario di $p_X(x)$ tale che:

$$S = \{x \mid p_X(x) > 0\},$$

allora si ha che:

$$D(p\|q) = \sum_{x \in S} p_X(x) \log_2 \frac{p_X(x)}{q_X(x)}. \tag{A.36}$$

Si noti che limitare la sommatoria nella formula A.36 ai valori di $x \in S$ anziché ai valori di $x \in \Re$ non modifica il valore dell'entropia relativa, in quanto:

$$\forall x \in \Re \wedge x \notin S \;\; p_X(x) = 0,$$

e quindi:

$$\forall x \in \Re \wedge x \notin S \;\; p_X(x) \log_2 \frac{p_X(x)}{q_X(x)} = 0.$$

Dalla formula A.36 segue anche:

$$-D(p\|q) = -\sum_{x \in S} p_X(x) \log_2 \frac{p_X(x)}{q_X(x)}; \tag{A.37}$$

per le proprietà del logaritmo:

$$-\sum_{x \in S} p_X(x) \log_2 \frac{p_X(x)}{q_X(x)} = \sum_{x \in S} p_X(x) \log_2 \frac{q_X(x)}{p_X(x)}. \tag{A.38}$$

Ora, applicando la diseguaglianza di Jensen

$$\sum_{x \in S} p_X(x) \log_2 \frac{q_X(x)}{p_X(x)} \leq \log_2 \sum_{x \in S} p_X(x) \frac{q_X(x)}{p_X(x)}; \tag{A.39}$$

semplificando:

$$\log_2 \sum_{x \in S} p_X(x) \frac{q_X(x)}{p_X(x)} = \log_2 \sum_{x \in S} q_X(x); \tag{A.40}$$

considerando di nuovo tutte le $x \in \Re$ si può affermare:

$$\log_2 \sum_{x \in S} q_X(x) \leq \log_2 \sum_{x \in \Re} q_X(x); \tag{A.41}$$

questa minorazione deriva dal fatto che è possibile che:

$$\exists x \in \Re \wedge x \notin \Re \;\; q_X(x) > 0.$$

È ora noto, in quanto proprietà della funzione di probabilità di massa, che:

$$\sum_{x \in \Re} f_X(x) = 1;$$

quindi:

$$\log_2 \sum_{x \in \Re} q_X(x) = \log_2 1 = 0. \tag{A.42}$$

In definitiva, si è ottenuto che:

$$-D(p\|q) \leq 0, \tag{A.43}$$

ovvero, come volevasi dimostrare:

$$D(p\|q) \geq 0. \tag{A.44}$$

Si noti, inoltre, che nella formula A.39, applicando la diseguaglianza di Jensen e avendo che $\log x$ è una funzione strettamente concava di x, è possibile sostituire la minorazione con un'uguaglianza se è solo se:

$$\frac{q_X(x)}{p_X(x)} = 1 \ \ \forall x \in \Re,$$

e quindi:

$$D(p\|q) = 0 \Longleftrightarrow p_X(x) = q_X(x) \ \ \forall x \in \Re. \tag{A.45}$$

Risulta così provata anche la seconda parte del teorema della diseguaglianza dell'informazione.

A.2.3 Teorema della distribuzione della massima entropia

Per dimostrare il teorema della distribuzione della massima entropia espresso in 5.2.5, si parta dall'entropia di g:

$$h(g) = -\int_S g \ln g. \tag{A.46}$$

Si introduca ora anche $f*$:

$$-\int_s g \ln g = -\int_S g \ln \left(\frac{g}{f*} f* \right); \tag{A.47}$$

usando le proprietà del logaritmo e la definizione di entropia relativa si ha:

$$-\int_S g \ln \left(\frac{g}{f*} f* \right) = -\int_S g \ln \frac{g}{f*} - \int_S g \ln f* = -D(g\|f*) - \int_S g \ln f*; \tag{A.48}$$

per la non negatività dell'entropia relativa si può scrivere:

$$-D(g\|f*) - \int_S g \ln f* \leq -\int_S g \ln f*; \tag{A.49}$$

usando la definizione di $f*$data nella formula (5.10) si ottiene:

$$-\int_S g \ln f* = -\int_S g \left(\lambda_0 + \sum_i \lambda_i r_i \right). \tag{A.50}$$

Ora, siccome sia g che $f*$ soddisfano i vincoli (5.7), (5.8) e (5.9) si ha:

$$-\int_S g \left(\lambda_0 + \sum_i \lambda_i r_i \right) = -\int_S f* \left(\lambda_0 + \sum_i \lambda_i r_i \right); \tag{A.51}$$

ricorrendo ancora una volta alla definizione di $f*$ data nella formula (5.10) si ottiene:

$$-\int_S f* \left(\lambda_0 + \sum_i \lambda_i r_i \right) = -\int_S f* \ln f*, \tag{A.52}$$

ovvero, per la definizione di entropia:

$$-\int_S f* \ln f* = h(f*). \tag{A.53}$$

In definitiva, come volevasi dimostrare:

$$h(g) \leq h(f*), \tag{A.54}$$

dove il caso limite di uguaglianza risulta vero se e solo $g(x) = f*(x)$ per ogni x; in quest'ultimo caso risulterebbe $d(g \| f*) = 0$ e sarebbe quindi possibile usare il segno di uguaglianza nella formula (A.49).

Bibliografia

[1] *Brain Facts - A primer on the brain and nervous system.*

[2] AA.VV. *Enciclopedia Medica Italiana.* Sansoni Edizioni Scientifiche, 1952.

[3] AA.VV. *The Clinical Neurophysiology Primer.* Humana Press, 2007.

[4] Brendan Allison. *P300 or not P300: Toward a Better P300 BCI.* PhD thesis, University of California, San Diego, 2003.

[5] Marcia Barinaga. Turning thoughts into actions. last access: July 2009, 1999.

[6] Niels Birbaumer, Andrea Kubler, Thilo Hinterberger, Nimir Ghanayim, Jouri Perelmouter, Jochen Kaiser, Iver Iversen, Boris Kochoubey, Nicola Neumann, and Herta Flor. The thought translation device (ttd) for completely paralyzed patients. *IEEE Transactions on Rehabilitation Engineering*, 8:190–193, 2000.

[7] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[8] Christopher M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 2006.

[9] Sergio Bittanti. *Identificazione dei Modelli e Sistemi Adattativi.* Pitagora Editrice, 2004.

[10] Gail A. Carpenter. Distributed activation, search and learning by art and artmap neural networks. Technical report, Boston University, 1996.

[11] Gail A. Carpenter and Stephen Grossberg. *The Handbook of Brain Theory and Neural Networks, Second Edition*, chapter Adaptive Resonance Theory, pages 79–82. MIT Press, 2002.

[12] Gail A. Carpenter, Stephen Grossberg, Natalya Markuzon, John H. Reynolds, and David B. Rosen. Fuzzy artmap: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks*, 3:698–712, 1992.

[13] Gail A. Carpenter, Stephen Grossberg, and John H. Reynolds. Artmap: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. *Neural Networks*, 4:565–588, 1991.

[14] Gail A. Carpenter, Stephen Grossberg, and David B. Rosen. A neural network realization of fuzzy art. Technical report, Boston University, 1991.

[15] Angela Cattini. Interfacce cervello-computer per la classificazione dell'attività immaginativa dell'uomo. Master's thesis, Università degli Studi La Sapienza, Roma, 2000-2001.

[16] Ohio State University Medical Centre. Brainstem stroke. last access: July 2009, 2008.

[17] Andrew Chipperfield, Peter Fleming, Hartmut Pohlheim, and Carlos Fonseca. *Genetic Algorithm Toolbox for Use with Matlab*. Department of Automatic Control and Systems Engineering of the University of Sheffield. Version 1.2.

[18] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley and Sons, 1991.

[19] Tiziano D'Albis. A predictive speller for a brain-computer interface based on motor imagery. Master's thesis, Politecnico di Milano, 2009.

[20] Janez Demsar. Statistical comparisons of classifers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[21] Jean-Louis Dessalles. Notes from the course 'collective intelligence'. last access: July 2009, 2009.

[22] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley and Sons, 2000.

[23] Chris Eliasmith and Charles H. Anderson. *Neural Engineering*. MIT Press, 2003.

[24] Ilenia Epifani. Appunti dal corso di 'statistica'. last access: July 2009, 2009.

[25] Georg E. Fabiani, Dennis J. McFarland, Jonathan R. Wolpaw, and Gert Pfurtscheller. Conversion of eeg activity into cursor movement by a brain-computer interface (bci). *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 12:331–338, 2004.

[26] Max Fogiel. *REA's Problem Solvers: Statistics*. Research And Education Association, 1991.

[27] I. I. Goncharova, Dennis J. McFarland, Theresa M. Vaughan, and Jonathan R. Wolpaw. Emg contamination of eeg: spectral and topographical characteristics. *Clinical neurophysiology*, 114:1580–1593, 2003.

[28] Bernhard Graimann, George Townsend, and Gert Pfurtscheller. Brain-computer communication - a brief introduction. last access: February 2009, 2005.

[29] B. Greenstein and A. Greenstein. *Color Atlas of Neuroscience*. Thieme, 2000.

[30] R. Gutierrez-Osuna. Notes from the course 'introduction to pattern analysis'. last access: July 2009, 2008.

[31] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh. *Feature extraction: foundations and applications*. Springer, 2006.

[32] Todd C. Handy. *Event-related Potentials: A Methods Handbook*. MIT Press, 2005.

[33] Bin He and Mitra Dutta. *Neural Engineering*. Springer, 2005.

[34] Leigh R. Hochberg, Mijail D. Serruya, Gerhard M. Friehs, Jon A. Mukand, Maryam Saleh, Abraham H. Caplan, Almut Branner, David Chen, Richard D. Penn, and John P. Donoghue. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. *Nature*, 442:164–171, 2006.

[35] Ulrich Hoffmann, Jean-Marc Vesin, and Touradj Ebrahimi. Recent advances in brain-computer interfaces. last access: July 2009, 2007.

[36] Daniel Kantor. Muscular dystrophy. last access: July 2009, 2006.

[37] Dean J. Krusienski. Brain-computer interface research and technology. last access: February 2009, 2005.

[38] Dean J. Krusienski, Dennis J. McFarland, and Jonathan R. Wolpaw. An evaluation of autoregressive spectral estimation model order for brain-computer interface applications. *International Conference of the IEEE Engineering in Medicine and Biology Society*, 1:1323–1326, 2006.

[39] Dean J. Krusienski, Gerwin Schalk, Dennis J. McFarland, and Jonathan R. Wolpaw. Tracking of the mu rhythm using an empirically derived matched filter. *International Conference of the IEEE Engineering in Medicine and Biology Society*, 1:v–viii, 2005.

[40] Emanuele Lauricella. *Dizionario Medico*. USES Edizioni Scientifiche Firenze, 1976.

[41] H. Liu and H. Motoda. *Computational Methods of Feature Selection*. Chapman & Hall / CRC, 2008.

[42] A Longstaff. *Neuroscience*. Taylor & Francis., 2005.

[43] Joaquim P. Marques de Sa. *Pattern Recognition*. Springer, 2001.

[44] Matteo Matteucci. Appunti dal corso di 'soft computing'. last access: July 2009, 2009.

[45] Marco Mattiocco. Stima dell'attività corticale nell'uomo correlata all'immaginazione motoria con metodi di stima lineare inversa. Master's thesis, Università degli Studi La Sapienza, Roma, 2002-2003.

[46] Dennis J. McFarland, Charles W. Anderson, Klaus-Robert Muller, Alois Schlogl, and Dean J. Krusienski. Bci meeting 2005 - workshop on bci signal processing: Feature extraction and translation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 14:135–138, 2006.

[47] Dennis J. McFarland, Lynn M. McCane, Stephen V. David, and Jonathan R. Wolpaw. Spatial filter selection for eeg-based communication. *Electroencephalography and clinical neurophysiology*, 103:386–394small laplacian filter, 1997.

[48] Dennis J. McFarland, Lynn M. McCane, and Jonathan R. Wolpaw. Eeg-based communication and control: Short-term role of feedback. *IEEE Transactions on Rehabilitation Engineering*, 6:7–11, 1998.

[49] Dennis J. McFarland, William A. Sarnacki, Theresa M. Vaughan, and Jonathan R. Wolpaw. Brain-computer interface (bci) operation: signal and noise during early training sessions. *Clinical neurophysiology*, 116:56–62, 2005.

[50] Dennis J. McFarland, William A. Sarnacki, and Jonathan R. Wolpaw. Brain-computer interface (bci) operation: optimizing information transfer rate. *Biological Psychology*, 63:237–251, 2003.

[51] Dennis J. McFarland and Jonathan R. Wolpaw. Eeg-based communication and control: Speed-accuracy relationships. *Applied Pychophysiology and Biofeedback*, 28:217–231, 2003.

[52] Dennis J. McFarland and Jonathan R. Wolpaw. Sensorimotor rhythm-based brain-computer interface (bci): Feature selection by regression improves performance. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 13:372–379, 2005.

[53] G. A. Mihailoff and C. Briar. *Nervous System*. Elsevier Health Sciences, 2005.

[54] Laurie A. Miner, Dennis J. McFarland, and Jonathan R. Wolpaw. Answering questions with electroencephalogram-based brain-computer interface. *Arch Phys Med Rehabil*, 79:1029–1033, 1998.

[55] Douglas C. Montgomery and George C. Runger. *Applied statistics and probability for engineers*. John Wiley and Sons, 1999.

[56] Klaus-Robert Muller, Charles W. Anderson, and Gary E. Birch. Linear and nonlinear methods for brain-computer interfaces. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 11:165–169, 2003.

[57] Gregory W. Neat, Dennis J. McFarland, Catherine A. Forneris, and Jonathan W. Wolpaw. Eeg-based brain-to-computer communication: system description. *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 12:2298–2300, 1990.

[58] Frank H. Netter, John A. Craig, James Perkins, John T. Hansen, and Bruce M. Koeppen. *Atlas of Neuroanatomy and Neurophysiology.* Icon Custom Communications, 2002.

[59] Ernst Niedermeyer and Fernando Lopes da Silva. *Electroencephalography: Basic Principles, Clinical Applications, and Related Fields.* Lippincott Williams and Wilkins, 2004.

[60] Jorge B. Ochoa. *EEG Signal Classification for Brain Computer Interface Applications.* PhD thesis, Ecole Polytechnique Federale de Lausanne, 2002.

[61] Vernon J. Odom, Michael Bach, Colin Barber, Mitchell Brigell, Michael F. Marmor, Alma P. Tormene, and Graham E. Holder & Vaegan. Visual evoked potentials standard (2004). *Documenta Ophthalmologica*, 108:115–123, 2004.

[62] National Institute of Neurological Disorders and Stroke. Ninds cerebral palsy information page. last access: July 2009, 2008.

[63] Brain Injury Association of Queensland Inc. About brain injury. last access: July 2009, 2008.

[64] Canadian Paraplegic Association Ontario. About spinal cord injury. last access: July 2009.

[65] Brad G. Osgood. Notes from the course 'the fourier transform and its applications'. last access: July 2009, 2007.

[66] Micheal O'Shea. *The Brain: a very short introduction.* Oxford University Press, 2005.

[67] Gert Pfurtscheller, C. Neuper, C. Guger, W. Harkam, Herbert Ramoser, Alois Schlogl, B. Obermaier, and Pregenzer M. Current trends in graz brain-computer interface (bci) research. *IEEE Transactions on Rehabilitation Engineering*, 8:216–219, 2000.

[68] Gert Pfurtscheller, C. Neuper, G. R. Muller, B. Obermaier, G. Krausz, Alois Schlogl, Scherer R., Bernhard Graimann, C. Keinrath, D. Skliris, M. Wortz, G. Supp, and C. Schrank. Graz-bci: State of art and clinical applications. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 11:177–180, 2003.

[69] L. Portinale and L. Saitta. Feature selection. last access: July 2009, 2002.

[70] Claudio Prati. *Segnali e sistemi per le telecomunicazioni.* McGraw-Hill, 2003.

[71] D. Purves, G. J. Augustine, D. Fitzpatrick, W. C. Hall, A. S. Lamantia, J. O. McNamara, and S. M. Williams. *NEUROSCIENCE: Third Edition.* Sinauer Associates, Inc., 2004.

[72] Gerwin Schalk. *Towards a clinically practical brain-computer interface.* PhD thesis, Rensselaer Polytechnic Institute, Troy, New York, 2006.

[73] Gerwin Schalk, Peter Brunner, Lester A. Gerhardt, H. Bischof, and Jonathan R. Wolpaw. Brain-computer interfaces (bcis): Detection instead of classification. *Journal of Neuroscience Methods*, 167:51–62, 2008.

[74] Gerwin Schalk, Eric C. Leuthardt, Peter Brunner, Jeffrey G. Ojemann, Lester A. Gerhardt, and Jonathan R. Wolpaw. Real-time detection of event-related brain activity. *Neuroimage*, 43:245–249, 2008.

[75] Gerwin Schalk, Dennis J. McFarland, Thilo Hinterberger, Niels Birbaumer, and Jonathan R. Wolpaw. Bci2000: A general-purpose brain-computer interface (bci) system. *IEEE Transactions on Biomedical Engineering*, 51:1034–1043, 2004.

[76] Thomas D. Schneider. Information theory primer. last access: July 2009, 2007.

[77] S. Theodoridis and K. Koutroumbas. *Pattern Recognition Second Edition.* Elsevier Academic Press, 2003.

[78] E. M. Tzanakou. *Supervised and Unsupervised Pattern Recognition: Feature Extraction and Computational Intelligence.* CRC, 1999.

[79] Tad J. Ulrych and Thomas N. Bishop. Maximum entropy spectral analysis and autoregressive decomposition. *Reviews of Geophysics and Space Physics*, 13:183–200, 1975.

[80] Theresa M. Vaughan, Dennis J. McFarland, Gerwin Schalk, William A. Sarnacki, Dean J. Krusienski, Eric W. Sellers, and Jonathan R. Wolpaw. The wadsworth bci research and development program: At home with bci. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 14:229–233, 2006.

[81] Theresa M. Vaughan, Laurie A. Miner, Dennis J. McFarland, and Jonathan R. Wolpaw. Eeg-based communication: analysis of concurrent emg activity. *Electroencephalography and clinical neurophysiology*, 107:428–433, 1998.

[82] Jacques J. Vidal. Toward direct brain-computer communication. *Annual Review of Biophysics and Bioengineering*, 2:157–180, 1973.

[83] Sa Wang, Cheng-Lin Liu, and Lian Zheng, editors. *Feature Selection by Combining Fisher Criterion and Principal Feature Analysis*, volume Proceedings of the Sixth International Conference on Machine Learning and Cybernetics, 2007.

[84] Eric W. Weisstein. Fast fourier transform. last access: July 2009, 2008.

[85] Ingrid Wickelgreen. Tapping the mind. *Science*, 299:496–499, 2003.

[86] Chris J. Wild and George A. F. Seber. *Chance Encounters: A First Course in Data Analysis and Inference*. John Wiley and Sons, 1999.

[87] Jonathan R. Wolpaw, Niels Birbaumer, William J. Heetderks, Dennis J. McFarland, P. Hunter Peckam, Gerwin Schalk, Emmanuel Donchin, Louis A. Quatrano, Charles J. Robinson, and Theresa M. Vaughan. Brain-computer interface technology: A review of the first international meeting. *IEEE Transactions on Rehabilitation Engineering*, 8:164–173, 2000.

[88] Jonathan R. Wolpaw, Niels Birbaumer, Dennis J. McFarland, Gert Pfurtscheller, and Theresa M. Vaughan. Brain-computer interfaces for communication and control. *Clinical neurophysiology*, 13:767–791, 2002.

[89] Jonathan R. Wolpaw and Dennis J. McFarland. Multichannel eeg-based brain-computer communication. *Electroencephalography and clinical neurophysiology*, 90:444–449, 1994.

[90] Jonathan R. Wolpaw and Dennis J. McFarland. Control of a two-dimensional movement signal by a human noninvasive brain-computer interface in humans. *PNAS*, 21:17849–17854, 2004.

[91] Jonathan R. Wolpaw, Dennis J. McFarland, Gregory W. Neat, and Catherine A. Forneris. An eeg-based brain-computer interface for cursor control. *Electroencephalography and clinical neurophysiology*, 78:252–259, 1991.

[92] Jonathan R. Wolpaw, Dennis J. McFarland, and Theresa M. Vaughan. Brain-computer interface research at the wadsworth center. *IEEE Transactions on Rehabilitation Engineering*, 8:222–226, 2000.

[93] Jonathan R. Wolpaw, Dennis J. McFarland, Theresa M. Vaughan, and Gerwin Schalk. The wadsworth center brain-computer (bci) research and development program. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 11:204–207, 2003.

[94] Jonathan R. Wolpaw, Herbert Ramoser, Dennis J. McFarland, and Gert Pfurtscheller. Eeg-based communication: Improved accuracy by response verification. *IEEE Transactions on Rehabilitation Engineering*, 6:326–333, 1998.