Image and Video Analysis: Sistema di riconoscimento automatico del livello di dolore utilizzando Random Forest Regressor

Lorenzo Gainasssi, Francesco Gigli
Università degli Studi di Firenze
lorenzo.gianassi@stud.unifi.it, francesco.gigli@stud.unifi.it

Abstract

L'analisi di sequenze di dati è un compito impegnativo al giorno d'oggi. Ci sono molti modelli in grado di gestire questo compito, uno dei più famosi è il Random Forest Regressor. L'obiettivo di questo progetto è prevedere il valore VAS, una misura utilizzata per le caratteristiche soggettive. Nel nostro ambito misura il dolore provato da un paziente sottoposto ad impulsi di vario tipo. Le previsioni devono essere fatte sulla base di un video, composto da una sequenza di fotogrammi. Il Framework che in questo progetto svolge questo lavoro è composto da una Combinazione di Gaussian Mixture Model (GMM) e Random Forest Regressor.

1. Introduzione

Negli ultimi anni il riconoscimento automatico del dolore è diventato di grande interesse in tutte quelle situazioni in cui la valutazione soggettiva del paziente non è applicabile, come nel caso di incoscienza o quando è affetto da problemi cognitivi o verbali, oppure può essere utilizzato come supporto agli autorilevamenti del paziente, fornendo un ulteriore strumento di analisi per il medico. Al contempo l'applicativo può essere sottoposto in contesti diversi, come per la valutazione del dolore percepito dagli animali, mediante il riconoscimento di espressioni, dove quest'ultime spesso non sono di facile classificazione come nel caso degli esseri umani. Questo ha portato alla necessità di avere una stima automatizzata del dolore per un sempre maggior numero di lavori, in quanto risulta un obiettivo di estrema sensibilità. L'elaborato in oggetto alla presente relazione è stato realizzato con l'obiettivo di estendere il lavoro svolto da un nostro collega che aveva come risultato la creazione di un sistema capace di predire il livello di dolore provato da una persona per mezzo dell'analisi di video che ne mostrano le varie espressioni. Questo progetto è stato implementato in linguaggio Python e prevede l'utilizzo del Dataset *UNBC-McMaster* per l'apprendimento supervisionato dell'indice di dolore come nel paper precedente. Il Dataset in questione per ogni frame di ciascuna sequenza contenuta al suo interno, associa alla posizione dei landmarks facciali del soggetto raffigurato nel video il corrispondente indice *VAS (Visual Analog Scale)*.

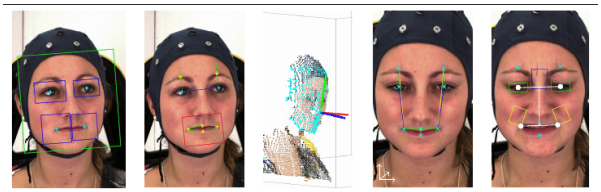

Figure 1. Un esempio dal Biovid Dataset.

Per quanto riguarda l'apprendimento supervisionato sono state utilizzate le **Random Forest** con estimatore di tipo **Regressor**. L'obbiettivo di questo progetto sarà quello di comparare i risultati ottenuti in precedenza utilizzando Support Vector Machine con i risultati ottenuti con Random Forest Regressor nella speranza di andare a migliorare i risultati.

2. Datasets

Ai fini dello svolgimento del progetto, è stata utilizzata l'ultima versione del Dataset *UNBC-McMASTER Shoulder Pain Expression*, il quale contiene sequenze video dei volti dei pazienti quando stavano attivamente e passivamente muovendo le loro spalle a seguito di impulsi dolorosi.
Le sequenze video sono 200 associate a 25 soggetti differenti. Quest'ultime a loro volta composte da un arbitrario numero di frames ed hanno associato un indice denominato *VAS (Visual Analog Scale)*, il quale rappresenta il livello di dolore percepito dal soggetto rappresentato nel video in una scala compresa tra 0 e 10. Ogni frame è caratterizzato dalle posizioni di 66 landmarks facciali estratti dal volto del soggetto, espresse per mezzo delle loro coordinate x ed y. Questo Dataset è stato usato nel lavoro svolto nel paper citato precedentemente [1].
Per un'analisi comparativa abbiamo svolto i calcoli sul medesimo Dataset, introducendo però un estimatore differente (Random Forest Regressor), per paragonare i risul-

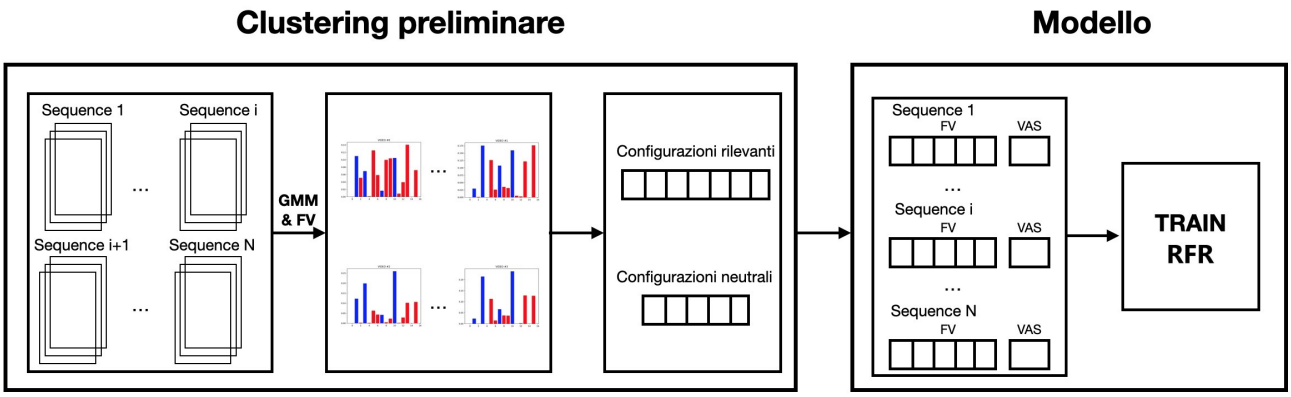

Figure 2. Schema del framework proposto

tati. Data però la dimensione limitata del Dataset *UNBC-McMASTER Shoulder Pain Expression*, per una analisi completa è stato utilizzato un Dataset aggiuntivo di dimensione maggiore, *BioVid Heat Pain Database (BioVid)*, che presenta caratteristiche differenti.
BioVid Heat Pain Database (BioVid) è un Dataset recente creato per migliorare l'affidabilità e obiettività nella misurazione del dolore. Per migliorare i sistemi automatizzati di riconoscimento del dolore, Biovid presenta oltre a video che mostrano l'espressione del dolore (segnali visivi), anche un insieme di segnali biologici.
BioVid raffigura un numero totale di 87 soggetti, suddivisi in tre fasce di età con uguale numero di rappresentanti maschili e femminili:

1. Età 18-35 (30 soggetti);
2. Età 36-50 (30 soggetti);
3. Età 51-65 (27 soggetti).

Tutti i video relativi agli 87 soggetti sono caratterizzati con la corrispondente intensità dello stimolo del dolore. I video sono etichettati con nessun dolore o con una delle altre quattro intensità del dolore: basso dolore (PA1, soglia del dolore), dolore intenso (PA4, tolleranza al dolore) e due intensità intermedie (PA2 e PA3). Per ciascun soggetto, ci sono 20 campioni video per classe di dolore e altri 20 campioni della condizione base in cui il soggetto non subisce impulsi dolorosi (pain-free baseline condition), ottenendo così un numero di 100 campioni per persona per un totale di 8.700 video.
Andremo quindi a valutare i risultati su questo Dataset nella speranza di migliorare quelli ottenuti in precedenza.

3. Framework per il riconoscimento automatico dell'indice VAS

L'architettura del sistema, del quale è riportato uno schema in figura 2, prevede la presenza di due fasi principali da eseguire in cascata l'una di seguito all'altra. Inizialmente, viene eseguita una prima parte dedita allo svolgimento di un'operazione di clustering preliminare, all'interno della quale sono in primo luogo estratte le features scalate atte a descrivere singolarmente ogni frame delle sequenze. Tali descrittori vengono quindi successivamente clusterizzati utilizzando i Fisher Vectors, in modo da individuare così quelle configurazioni che essendo associate all'espressione neutrale non sono da considerarsi come caratterizzanti per la classificazione dell'indice VAS .
Nel progetto del nostro collega utilizzando le sole configurazioni rilevanti vengono estratti i descrittori delle sequenze, i quali sono infine posti in input ad un modello SVRM che una volta addestrato si occupa di restituire l'indice VAS di una sequenza a partire dal vettore multidimensionale che la descrive.
Il lavoro svolto in questo paper sarà quello di modificare l'ultima fase del framework sostituendo al modello SVRM un modello Random Forest Regressor che analizzeremo nel paragrafo successivo, relativo al modello.

3.1. Clustering preliminare

La fase di clustering preliminare è composta dalle seguenti fasi:

1. Estrazione delle features atte a caratterizzare ciascun frame delle sequenze appartenenti al training set.
2. Scaling delle features estratte al fine di renderle maggiormente robuste rispetto alla presenza di outliers.
3. Rappresentazione di ciascun frame a partire dal suo vettore di features mediante l'utilizzo di *Fisher Vectors (FV)*.
4. Estrazione delle configurazioni rilevanti rappresentatando ogni sequenza di train attraverso un istogramma atto ad indicare la ricorrenza delle configurazioni rilevate al suo interno.

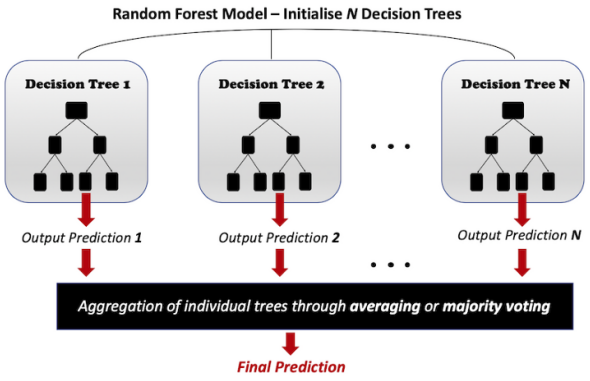

Figure 3. Funzionamento della Random Forest.

Per ottenere un'analisi più completa e dettagliata della fase di Clustering preliminare è possibile consultare il paper citato [1].

3.2. Modello RFR

3.2.1 Features delle sequenze

Una volta terminata la fase di clustering preliminare, le configurazioni delle features considerate rilevanti per la classificazione dell'indice VAS vengono in primo luogo utilizzate per generare i FV atti a descrivere interamente le sequenze a partire da quelli dei frames che le compongono. Nello specifico, il FV associato ad una generica sequenza j, che denotiamo come FV_j, è dato dalla somma delle righe correlate a media e varianza delle configurazioni rilevanti, dei FV associati ai frames che compongono la sequenza stessa. Posto quindi D il numero di landmarks considerati e P il numero di configurazioni rilevanti estratte durante il clustering preliminare, ciascuna sequenza viene descritta da un FV di dimensioni $2 \times D \times P$. A tali vettori multidimensionali viene poi applicata un' operazione di flatten, attraverso la quale ciascun video viene descritto per mezzo di un vettore unidimensionale di lunghezza pari a $2DP$.

3.2.2 Random Forest

Una *Random Forest* è innegabilmente uno dei migliori modelli per ottenere una soluzione rapida e ragionevole alla maggior parte dei problemi di dati strutturati. Possono adattarsi sia a problemi di regressione (come nel caso analizzato in questo paper) che di classificazione, sono resistenti all'over-fitting (con stimatori sufficienti) e possono funzionare senza alcuna standardizzazione dei dati o creazione di variabili fittizie. La chiave per formare un modello di foresta casuale è un albero decisionale: ne combiniamo molti insieme per ottenere una migliore prestazione complessiva. Gli alberi decisionali utilizzati conducono una serie di divisioni binarie ricorsive *greedy*, ciascuna delle quali è calcolata in base ad una funzione obiettivo per il guadagno di informazioni. Per gli alberi di regressione, le funzioni obiettivo comuni includono errore quadratico medio (MSE), errore medio assoluto (MAE) o deviazione standard. I singoli alberi soffrono di una varianza elevata, ma la riduciamo in modo significativo e contemporaneamente aumentiamo le prestazioni predittive calcolando la media di molti alberi insieme.
Come panoramica, un algoritmo di foresta casuale esegue le seguenti operazioni:

1. Crea n campioni bootstrap dai dati di addestramento, che vengono utilizzati per addestrare n alberi decisionali. Un campione bootstrap è un campione randomizzato dei dati di addestramento originali con sostituzione, il che significa che abbiamo alcuni punti che possono riapparire in campionamenti futuri, inoltre possiamo avere anche campioni mancanti del tutto, noti come $out - of - bag$.

2. Sviluppa n alberi decisionali utilizzando gli n campioni con bootstrap.

3. Per ogni divisione nei singoli alberi, sceglie a caso un campione di elementi e seleziona la divisione migliore utilizzando solo quelle colonne. La scelta di effettuare questa suddivisione si basa sulla funzione obiettivo utilizzata (es. Entropia, indice di Gini, errore di classificazione, errore quadratico medio).

4. Aggrega i risultati per formare una previsione generale. Per la regressione, questa è solitamente la media, mentre per la classificazione, ciò avverrà spesso attraverso il voto a maggioranza.

3.2.3 Hyperparameter Tuning

Per comprendere l'*Hyperparameter Tuning* dobbiamo prima sottolineare la differenza tra parametri e Iperarametri. I parametri del modello vengono appresi durante l'addestramento, gli iperparametri invece devono essere impostati dal data scientist prima dell'addestramento. Nel caso di una Random Forest, gli iperparametri includono ad esempio, il numero di alberi decisionali nella foresta o il numero di caratteristiche considerate da ciascun albero quando si divide un nodo.
L'ottimizzazione degli iperparametri si basa più sui risultati sperimentali che sulla teoria, quindi il metodo migliore per determinare le impostazioni ottimali è provare molte combinazioni diverse per valutare le prestazioni di ciascun modello. Tuttavia, valutare ogni modello solo sul training set può portare al problema dell'**overfitting**, il quale può essere gestito mediante l'utilizzo di tecniche di **Cross Validation**. Abbiamo solo una vaga idea dei migliori iperparametri e quindi l'approccio migliore per restringere la

nostra ricerca è valutare un'ampia gamma di valori per ciascun iperparametro. Per rappresentare e gestire la *grid* di parametri su cui testare il modello utilizziamo il metodo di Scikit-Learn, **RandomizedSearchCV**. Utilizzando questo metodo, viene definita una griglia di intervalli di iperparametri dalla quale vengono scelti dei campioni casuali. Viene infine eseguito un K-Fold CV con ogni combinazione di valori.
Risultati ancora più ottimizzati possono essere ottenuti combinando a **RandomizedSearchCV** un altro tipologia di tuning degli HyperParamenter 3.2.3. Questo metodo è chiamato **GridSearchCV**, il quale invece di campionare casualmente da una distribuzione, valuta tutte le combinazioni definite a priori dal data scientist. Per utilizzare *GridSearch*, creiamo un'altra griglia basata sui migliori valori forniti dalla *RandomizedSearch* fatta in precedenza. I risultati ottenuti saranno quindi frutto della concatenazione delle due tecniche di ottimizzazione descritte in precedenza.

3.3. Valutazione delle performance

Il modello RFR consente di predire l'indice VAS associato ad una certa sequenza a partire dal FV che la descrive.
Nel processo di inferenza quindi, per ogni sequenza appartenente al test set, viene generato il vettore multidimensionale che la caratterizza esattamente come già visto per le sequenze di train. Vengono cioè calcolati i FV dei frames che la compongono e successivamente questi sono sommati nelle sole componenti associate alle configurazioni rilevanti delineate dal clustering preliminare per generare il descrittore della sequenza stessa. I FV dei video di test sono quindi posti in input al metodo di predizione del modello, il quale ne restituisce l'indice VAS espresso come un valore decimale.
La bontà del modello, viene quindi in primo luogo valutata calcolando il **mean absolute error (MAE)** sulle sequenze di test. A seguito di ogni predizione, il risultato viene mappato all'interno dell'intervallo compreso tra 0 e 10, per poi essere arrotondato al numero intero più vicino. Il mse medio è infine calcolato come la media dei singoli errori ricavati su ogni sequenza di test.
Seguendo l'approccio e i test svolti dal nostro collega nel paper [1] viene anche prodotta in output una **matrice di confusione** con l'obbiettivo di mostrare in modo chiaro e riassuntivo le prestazioni del modello durante la predizione delle varie classi. Le matrici di confusione M sono composte da 11 righe e da altrettante colonne (una per ogni classe) nel caso del *UNBC-McMASTER Shoulder Pain Expression*, mentre nel secondo dataset (*The BioVid Heat Pain Database (BioVid)*) il numero di righe e colonne è di 5, corrispondenti al numero di classi di VAS. Un suo generico elemento m_{ij} indica la quantità di volte in cui il modello predice un esempio di classe i con la classe j. La matrice in questione, una volta calcolata viene poi normalizzata. Idealmente quindi, un modello che performa perfettamente ha tutti elementi pari ad 1 sulla diagonale e valori nulli al di fuori di essa mentre, viceversa, più i valori tendono a concentrarsi al di fuori della diagonale e peggiori saranno le sue performance.
Infine per eseguire dei test per valutare un possibile overfit del modello è stato utilizzato **neg mean absolute error (NMAE)** che, come suggerisce il nome, è semplicemente il **MAE** valutato sull'asse negativo.

4. Implementazione

Il framework descritto nel paragrafo 3 è stato implementato con il linguaggio di programmazione Python. Il progetto si basa su due script denominati **PreliminaryClustering.py** e **ModelRFR.py**, i quali hanno lo scopo di implementare, come meglio descritto di seguito, rispettivamente la fase atta all'estrazione delle configurazioni rilevanti e quella relativa alla gestione della Random Forest Regressor.
In questo progetto ci basiamo sul codice fornito dal nostro collega, al quale sono apportate le modifiche per andare a utilizzare un modello differente (Random Forest Regressor) e un dataset aggiuntivo di dimensioni maggiori.
Andiamo ad analizzare le differenze nello specifico nel seguente Sottoparagrafo.

4.1. Modifiche Implementative

La principale modifica implementativa rispetto al progetto precedente è l'introduzione del modello **Random Forest Regressor** che va a sostituire **Support Vector Machine Regressor**. Di conseguenza la classe principale relativa al modello RFR che era *ModelSVR.py* è stata modificata per gestire e creare il modello di Random Forest Regressor e prenderà il nome di *ModelRFR.py*. In questa classe inoltre sono stati modificati i metodi per il training e il testing del modello.
Per quanto riguarda il training è stato modificato il metodo *__train_maximizing_score()* in quanto è stato utilizzato il protocollo di ottimizzazione **RandomizeSearchCV** per ottenere un tuning sui parametri che meglio sarà descritto nel paragrafo riguardante i test. Per quanto riguarda il testing del modello è stato aggiunto un metodo per valutare il possibile overfit, *evaluate_overfit()*, che utilizza la griglia di valori ottenuta da **RandomizeSearchCV** per eseguire un **GridsearchCV** mediante il metodo *__gridsearch()*.
Inoltre, dovendo gestire un nuovo Dataset, sono state estratte le sequenze prese in considerazione dall'insieme di samples totali, e rispetto a queste sono state calcolate le coordinate nel formato consono alla classe *PreliminaryClustering.py*. Per eseguire tali operazioni sono stati implementati rispettivamente *generate_biovid_sequence.py* e *generate_biovid_coords.py*.
Eseguito ciò, creiamo i file *BioVid_sequence.csv* e *BioVid_coord.csv* rappresentanti rispettivamente:

- le sequenze, una per ogni riga del file, associate ai vari soggetti;
- le coordinate x e y dei vari landmarks frame per frame.

Inoltre è stato implementato *test_regression.py* che si occupa di eseguire dei test più specifici che verranno descritti nel paragrafo successivo.

5. Tests

I test svolti sul Dataset *UNBC-McMASTER Shoulder Pain Expression* hanno l'obbiettivo di replicare quelli svolti dal nostro collega con il fine di mostrare possibili miglioramenti rispetto ai risultati ottenuti con l'utilizzo delle **Support vector Machine**. Lo script utilizzato per eseguire tali test è **test_regression.py**, il cui scopo è quello di poter confrontare i risultati ottenuti al variare del valore utilizzato come soglia per le configurazioni neutrali e al contempo valutare i diversi raggruppamenti di landmarks. Questo viene eseguito mediante lo scorrimento su 4 gruppi di landmarks, rappresentanti in ordine gli occhi, il naso, la bocca e la configurazione migliore, denominata "standard" formata da 15 landmark specifici, per quanto riguarda il Dataset *UNBC-McMASTER Shoulder Pain Expression*, mentre solo sul gruppo di 67 landmarks totali per il Dataset *The BioVid Heat Pain Database (BioVid)*. Inoltre ogni gruppo sarà analizzato in base al variare del valore della soglia per le configurazioni neutrali ottenendo così le matrici di confusione, sia di train che di test, un grafico relativo al MAE e la rappresentazione grafica di un albero di regressione scelto randomicamente tra quelli presenti nella foresta, tutto ciò eseguito soglia per soglia. In particolar modo i test sono stati eseguiti utilizzando 16 Kernels, che rappresentavano e rappresentano, anche nel nostro caso, la configurazione che portava ad un miglior risultato. Inoltre sono stati utilizzati tre protocolli differenti Leave-One-Subject-Out, 5-fold-cross-validation e Leave-One-Sequence-Out.
Per valutare la qualità del modello inoltre, sono stati svolti dei test sulle performance del RFR all'aumentare del numero di aberi presenti nella foresta randomica, valutando la differenza tra i risultati ottenuti tra training set e test set. Successivamente sono stati svolti i test sul Nuovo dataset per osservare un possibile miglioramento dato dalla dimensione magiore del dataset.

5.1. Parametri di Random Forest Regressor

Il modello Random Forest Regressor utilizza una serie di parametri che ne vanno ad individuare caratteristiche e resa dei risultati. I parametri utilizzati dal Random Forest Regressor sono i seguenti:

- $n\_estimators$: il numero di alberi nella foresta. Maggiore è il numero di alberi, maggiore sarà la precisione in quanto si avrà più campioni su cui fare una media, ma al contempo maggiore sarà il tempo necessario per l'elaborazione del risultato. Nel caso non ottimizzato sono stati utilizzati 10 estimatori, mentre nel caso ottimizzato vi era un limite superiore di 50 estimatori, ma generalmente ne vengono utilizzati tra i 10 e i 20 dal *RandomizedSearchCV*. Un esempio di albero ottenuto nel processo è visionabile nella Fig. 4.

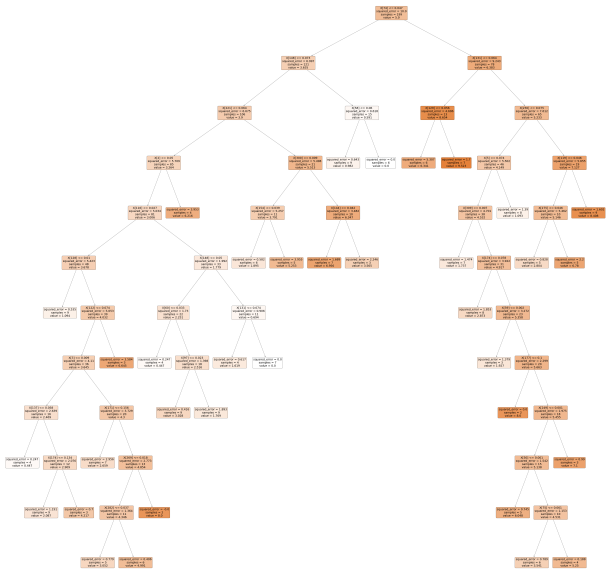

Figure 4. Esempio di albero di decisione creato dal Random Forest Regressor.

- $criterion$: questa funzione misura la qualità dello split dell'albero di decisione. I criteri più utilizzati sono: *mean_squared_error*, *mean_absulute_error* e *poisson*. Nel caso non ottimizzato è stato usato mean squared error (MAE) eseguito tra i valori predetti e quelli effettivi, con formula descritta nell'equazione sottostante.

$$I(t) = MAE(t) = \frac{1}{N_t} \sum_{i \in D_t} |y^i - \hat{y}_t|$$

  Mentre nel caso ottimizzato veniva scelto dal *RandomizedSearchCV* il miglior criterio tra quelli sopra citati.

- max_depth: la massima profondità che può essere raggiunta dall'albero. Se non viene espressa, questo continuerà ad espandersi fino a che tutte le foglie non risultano pure o fino a che tutte le foglie non contengano un numero minore o uguale a quello definito nel parametro $min\_samples\_split$. Questo parametro rappresenta uno dei più importanti per l'ottimizzazione in quanto più profondo è l'albero, più divisioni ha e acquisisa ce più informazioni sui dati. Una depth molto

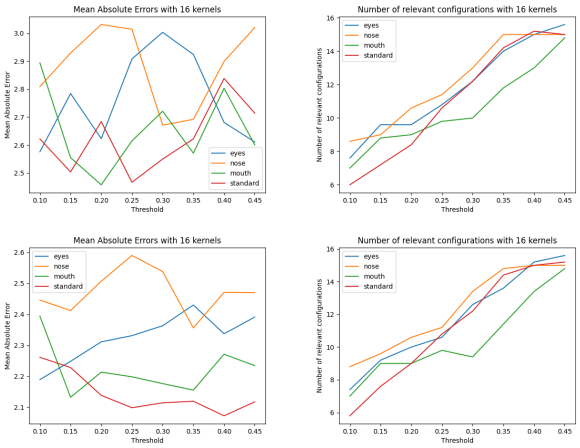

Figure 5. Valore del Mean Absolute Error al variare della Threshold per i gruppi di landmark relativi a occhi, naso, bocca e standard. Numero di configurazioni rilevanti dei vari gruppi di landmark sempre al variare del valore della Threshold. Nella prima riga sono riportati i risultati che non utilizzano l'ottimizzazione degli Hyperparameters.Nella seconda riga viene applicata l'ottimizzazione.

elevata però potrebbe portare a un overfit, portando il modello a una perdita di generalità.

- *bootstrap*: Il bootstrap aggregation (bagging) è un tecnica generale utilizzata prevalentemente per ridurre la varianza di un metodo di apprendimento. Questo avviene estraendo casualmente B diversi sottoinsiemi di osservazione dal training set, per simulare la variabilità dei dati. Da ogni sottoinsieme avviene l'apprendimento mediante un regressore f_i per ciascun sottoinsieme i seperatamente, successivamente vengono aggregate le predizioni sulle nuove istanze x di test come descritto nell'equazione:

$$f_{\text{bag}}(x) = \frac{1}{B}\sum_{i=1}^{B} f_i(x)$$

Come parametro è stato messo a True nel caso non ottimizzato, quindi sono stati utilizzati sottoinsiemi del Dataset, mentre nel caso ottimizzato veniva scelto a discrezione dell'algoritmo di *RandomizedSearchCV* se porlo a True o False, dove nel secondo caso viene utilizzato l'intero albero.

5.2. Test sul Dataset "UNBC-McMASTER Shoulder Pain Expression"

Andiamo adesso ad analizzare i primi test svolti sul Dataset *UNBC-McMASTER Shoulder Pain Expression* in cui usiamo un numero fisso di kernel pari a 16, configurazione con cui sono stati ottenuti i migliori risultati, e in

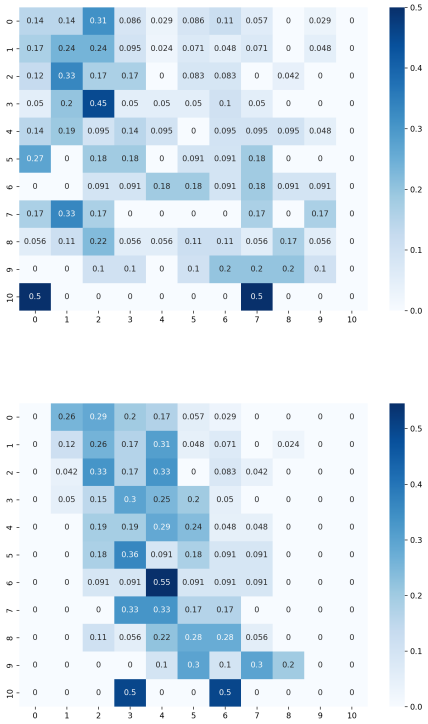

Figure 6. Matrici di Confusione sul Dataset *UNBC-McMASTER Shoulder Pain Expression*. La prima riga rappresenta il fit non ottimizzato con threshold 0.25. La seconda riga rappresenta il caso ottimizzato con threshold 0.40.

cui utilizziamo il protocollo *5-fold-cross-validation*. In particolar modo riportiamo nella Fig. 5 i risultati ottenuti in questa configurazione senza l'ausilio della funzione di ottimizzazione che sfrutta il metodo RandomizedSearchCV nella prima riga dell'immagine, mentre nella seconda riga dell'immagine riportiamo i valori ottenuti ottimizzando. Inanzitutto notiamo che il valore del MAE non scende mai sotto la soglia del 2.4, risultati ben peggiori di quelli riportati dal nostro collega utilizzando come modello SVMR. Per quanto riguarda invece il numero di configurazioni rilevanti notiamo, per ogni gruppo di landmark, che il numero di quest'ultime cresce all'aumentare della soglia di threshold che utilizziamo, risultato in linea con quanto era possibile predire.Il prossimo passo è quello di analizzare i risultati ottenuti applicando l'ottimizzazone degli *Hyperparameters*. Si può notare che l'andamento del grafico che rappresenta il numero di configurazioni rilevanti segue quello ottenuto dai risultati non ottimizzati. Al contempo, per quanto riguarda i valori del MAE, si ha un notevole miglioramento delle prestazioni.
Adesso il valore più basso ottenuto da un gruppo di landmarks rispetto a una threshold si attesta intorno a 2.1. Con questa configurazione siamo riusciti ad avvicinarci ai risul-

tati del nostro collega, comunque migliori, sebbene abbiano dei valori paragonabili. Per scendere nel dettaglio si analizzerà nella Fig. 6 le matrici di confusione delle due versioni (ottimizzata e non). In modo specifico le matrici di confusione considerate sono relative ai risultati migliori ottenuti.
La prima matrice che consideriamo rappresenta il risultato dell'operazione di fit non ottimizzato sul Dataset con threshold 0.25. La seconda matrice invece rappresenta il caso ottimizzato associato alla threshold 0.40. Si può notare un'immediata differenza sulla distribuzione degli elementi all'interno delle due matrici. Nel caso non ottimizzato, performando in modo peggiore, i valori delle previsioni si distribuiscono in modo omogeneo su di essa. Nel caso ottimizzato, invece, i valori sono più raggrupati vicino alla diagonale, testimonianza di una capacità migliore del modello nel prevedere i risultati. Infatti nello scenario migliore la matrice di confusione dovrebbe rapprentare una diagonale.
Per rendere completa la comparazione con i risultati ottenuti dal nostro collega sul Dataset *UNBC-McMASTER Shoulder Pain Expression* sono riportati i risultati relativi agli altri protocolli, in particolar modo *Leave-One-Subject-Out* e *Leave-One-Sequence-Out*.

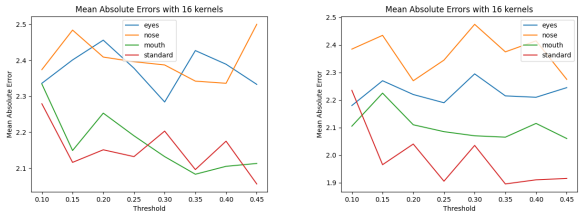

Figure 7. Valore del Mean Absolute Error al variare della Threshold per i gruppi di landmark relativi a occhi, naso, bocca e standard. La prima Immagine è relativa al protocollo *Leave-One-Sequence-Out* la seconda a *Leave-One-Subject-Out*, entrambi con il protocollo ottimizzato.

A partire dai due grafici è possibile confrontare i risultati con il protocollo usato in precedenza. In particolar modo *Leave-One-Sequence-Out* produce dei risultati paragonabili a *5-fold-cross-validation*, quindi non rappresenta un miglioramento effettivo, mentre *Leave-One-Subject-Out* riesce ad arrivare ad un MAE pari a 1.9 per le threshold 0.25 e 0.35 relative al gruppo di landmark denominato "standard" che viene rappresentato in rosso nei grafici dell'immagine 7. In generale, però, i risultati che si ottengono sui vari gruppi di landmarks sono i medesimi con andamenti dei grafici paragonabili fra i vari protocolli.
Per *Leave-One-Subject-Out* ad ogni round vengono selezionate le sequenze di un certo soggetto per il testing, mentre quelle di tutti i rimanenti 24 sono usate per il training. Al round successivo viene scelto un altro soggetto per il testing e la procedura viene così iterata fino a quando tutti i soggetti non sono stati usati per valutare le performance. Stesso concetto per *Leave-One-Sequence-Out*, ad ogni round viene selezionata una sola sequenza di test e tutte le altre vengono utilizzate per eseguire il training (199).
Per come sono costituiti, il problema principale di questi due nuovi protocolli sono il numero di cicli di esecuzione, che è maggiore del protocollo *5-fold-cross-validation*. Infatti avremo 25 cicli per *Leave-One-Sequence-Out* e 200 per *Leave-One-Subject-Out*. Quindi sebbene i risultati possano essere migliori, i tempi di esecuzione si dilatano. Sarà necessaria una valutazione del tradeoff tra tempi di esecuzione e risultati ottenuti.

5.2.1 Valutazione Overfitting

Prima di passare a valutare i risultati ottenuti sul nuovo Dataset, consideriamo la possibilità di essere andati incontro al fenomeno dell'overfitting. Analizziamo quindi il valore del Negative Mean Absolute Error all'aumentare del numero di alberi presenti nella foresta.
Il problema dell'overfit si poteva presentare in quanto la dimensione del Dataset, e di conseguenza quello del Training Set, utilizzato risultava essere limitata portando alla perdita di generalizzazione. Dalla Fig. 8 si nota che il gap fra i risultati ottenuti su Train Set e sul Test Set rimane costante e non eccessivo, quindi possiamo affermare che non vi è la presenza di overfitting nel modello. Per quanto riguarda il tempo di esecuzione può essere interessante notare che il tempo necessario cresce linearmente fino a 250 alberi, rimanendo poi stabile dopo tale soglia.

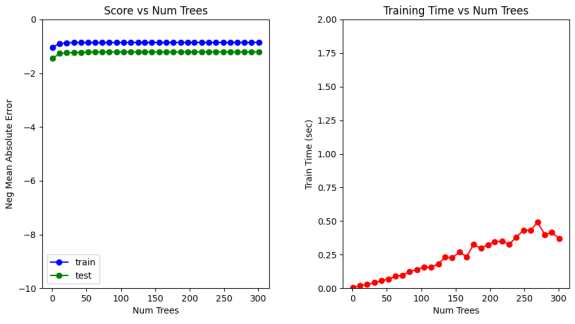

Figure 8. Valore del Negative Mean Absolute Error all'aumentare del numero di alberi nella foresta e tempo di esecuzione necessario. A sinistra si osserva come dopo solo 20 alberi circa il valore di Negative Absulute Error rimane costante. Mentre a destra notiamo la crescita lineare del tempo di esecuzione al crescere del numero di alberi.

5.3. Test sul Dataset "BioVid Heat Pain Database (BioVid)"

Procediamo adesso con l'analisi dei risultati ottenuti sul Dataset *The BioVid Heat Pain Database (BioVid)*. L'introduzione di questo Dataset all'interno del progetto è dovuta alla dimensione limitata del precedente. L'approccio della valutazione dei risultati sarà lo stesso applicato nel caso di *UNBC-McMASTER Shoulder Pain Expression*, infatti, sono riportati i valori di *Mean Absolute Error* nel caso in cui vengano scelti tutti i landmark (in questo caso 67), utilizzando 16 Kernels e come protocollo 5-Fold. Questa configurazione rappresenta il miglior compromesso tra efficacia del framework e tempi di esecuzione, che negli altri due protocolli si dilatano (come spiegato nel paragrafo 5.2). Nell'immagine a sinistra è rappresentato il caso senza ottimizzazione, a destra invece il caso con ottimizzazione degli *Hyper Parameter*. Inoltre rispetto al caso precedente non sono stati riportati i grafici relativi al numero di configurazioni rilevanti al variare della Threshold. Questo perchè i risultati ottenuti hanno un andamento pressochè identico a quello del Dataset *UNBC-McMASTER Shoulder Pain Expression*. Già a partire dal caso non ottimizzato è possibile notare un netto miglioramento delle presazioni rispetto al Dataset precedente, con risultati che si aggirano in un range tra 1.30 e 1.50 di *Mean Absolute Error*. Per quanto riguarda il caso ottimizzato otteniamo un ulteriore miglioramento con risultati si attestano tra 1.215 e 1.195.

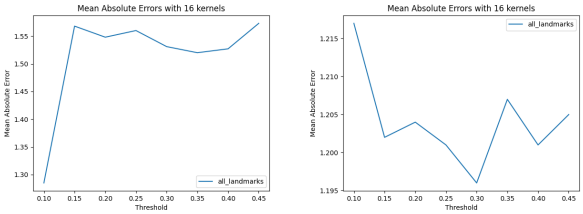

Figure 9. Valore del Mean Absolute Error al variare della Threshold per tutti i landmark con protocollo *5-fold-cross-validation*. Nella prima immagine sono riportati i risultati che non utilizzano l'ottimizzazione degli Hyperparameters. Nella seconda, invece, viene applicata l'ottimizzazione.

Probabilmente l'utilizzo di uno degli altri due protocolli potrebbe portare a un miglioramento dei risultati, ma data la dimensione elevata del Dataset e dell'utilizzo di tutti e 67 landmarks i tempi di esecuzione (che già erano lunghinel caso del dataset *UNBC-McMASTER Shoulder Pain Expression*), il tempo per eseguire il ciclo del framework diventa proibitivo.
Il prossimo passo è l'analisi delle matrici di confusione rispettive alla threshold che ha portato ad ottenere il miglior risultato di di *Mean Absolute Error*, rispettivamente 0.35 e 0.3 per il caso non ottimizzato e ottimizzato.

Nella Fig. 10 già ad una prima vista le differenze tra le due matrici sono notevoli. Partendo dal caso senza ausilio di ottimizzazione notiamo un' assegnazione dei VAS più sparsa rispetto alla diagonale sperata. Ma questi risultati sono comunque migliori del caso non ottimizzato visto in precedenza nel capitolo 5.2, grazie al fatto che il Database risulta essere più grande, quindi il numero di configurazioni rilevanti su cui eseguire i test è maggiore. Nel caso ottimizzato invece notiamo che i valori predetti si accentrano sempre di più ottenedo una retta di previsione verticale relativa al valore 2 di VAS. Benchè questo grafico sia associato al miglior risultato ottenuto, la matrice di confusione così riportata non rappresenta un risultato sano. Ciò potrebbe essere dovuto ad un numero minore di indici associati al dolore rispetto al Dataset *UNBC-McMASTER Shoulder Pain Expression*. Abbiamo quindi che i valori che sono vicini al VAS mediano tendono ad essere assegnati al VAS centrale, dovuti anche alla regressione attuata dal modello.

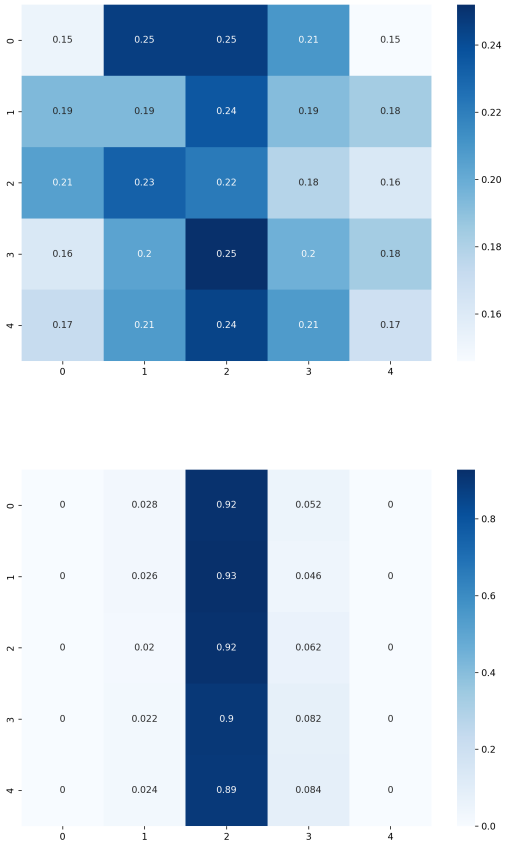

Figure 10. Matrici di Confusione delle migliori configurazioni del protocollo *5-fold-cross-validation*. Nella prima riga caso non ottimizzato con threshold 0.35, nella seconda riga caso ottimizzato con threshold 0.3. Rappresentano le soglie con i risultati migliori.

5.4. Confronto tra i risultati dei due Dataset

UnBC Dataset			
5-Fold Cross Validation			
	SVM	RFR	
Threshold	Opt	Not-Opt	Opt
0.1	2.41	**2.62**	**2.261**
0.15	1.98	**2.50**	**2.228**
0.2	1.95	**2.68**	**2.138**
0.25	1.87	**2.46**	**2.098**
0.3	1.84	**2.55**	**2.114**
0.4	1.98	**2.83**	**2.072**
0.5	2.02	**2.71**	**2.117**

BioVid Dataset			
5-Fold Cross Validation			
	SVM	RFR	
Threshold	Opt	Not-Opt	Opt
0.1	2.41	**1.25**	**1.21**
0.15	1.98	**1.56**	**1.20**
0.2	1.95	**1.54**	**1.20**
0.25	1.87	**1.56**	**1.20**
0.3	1.84	**1.53**	**1.19**
0.4	1.98	**1.52**	**1.20**
0.5	2.02	**1.57**	**1.20**

Per evidenziare i miglioramenti ottenuti sono riportati i valori di MAE per ogni threshold sia nel caso di utilizzo di SVM che di Random Forest Regressor. In verde sono evidenziati i valori che ottengono un MAE minore rispetto al caso SVM. Nella seconda tabella sono riportati i risultati, invece, relativi al secondo Dataset utilizzato *BioVid Heat Pain Database (BioVid)*. In questa sezione si può notare come il framework performi molto meglio rispetto al Dataset *UNBC-McMASTER Shoulder Pain Expression* a testimonianza del fatto che la dimensione del Dataset influisca molto sulle prestazioni del framework utilizzato. Questo confronto non è atto a mostrare i miglioramenti ottenuti in modo generico, ma bensì, a dimostrare che il framework possa ottenere dei risultati ottimi utilizzando un dataset di dimensione sufficienti da permettere al modello di generalizzare in modo corretto.

6. Conclusioni e Sviluppi Futuri

Esaminando i risultati ottenuti su entrambi i Datasets, si può affermare di aver migliorato in media i valori ottenuti dal nostro collega nel suo paper [1] utilizzando come modello di apprendimento Random Forest Regressor, come mostrato nel paragrafo precedente. Però analizzando nel particolare i valori ottenuti nella matrice di confusione (con ottimizzazione) si nota che questo risultato di media sia solo apparente, in quanto il modello assegnerà frequentemente il VAS mediano. Questo risultato attesta la non capacità del modello di apprendere e generalizzare in modo completo. Per possibili sviluppi futuri potrebbe essere interessante andare a eseguire i test sul Dataset *BioVid Heat Pain Database (BioVid)* utilizzando gli altri due protocolli: Leave-One-Subject-Out e Leave-One-Sequence-Out. Questi due protocolli potrebbero richiedere tempi di esecuzione più lunghi dato che il numero di cicli di esecuzione aumentano.
Inoltre anche l'utilizzo di un numero differente di Kernel per le GMM potrebbe portare a dei valori migliori rispetto a quelli ottenuti, in questa relazione è stato considerato solamente il caso 16 kernel che rappresentava la soluzione migliore per [1]. Per concludere i risultati sono stati ottenuti eseguendo il train e il test su tutti i landmark a disposizione del Dataset *BioVid Heat Pain Database (BioVid)* a differenza di *UNBC-McMASTER Shoulder Pain Expression* che presentava dei sottoinsiemi ottimali di landmarks. Questo implica che potrebbe essere possibile migliorare i risultati di previsione ulteriormente ed accelerare anche i tempi di esecuzione, non dovendo scorrere su tutti i landmarks.

References

[1] A. Arezzo, "Image and video analysis: Sistema di riconoscimento automatico del livello di dolore percepito," 2019.