



SAPIENZA
UNIVERSITÀ DI ROMA

Analisi di Immagini Mediche: Segmentazione e Classificazione di tumori cerebrali con tecniche di Computer Vision

Facoltà di Ingegneria dell'Informazione, Informatica e Statistica
Laurea Triennale in Informatica

Candidato

Davide Belcastro

Matricola 1962536

Responsabile

Prof. Luigi Cinque

Co-Responsabili

Prof. Daniele Pannone

Prof. Danilo Avola

Anno Accademico 2022/2023

**Analisi di Immagini Mediche: Segmentazione e Classificazione di tumori cerebrali
con tecniche di Computer Vision**

Tesi di Laurea Triennale. Sapienza Università di Roma

© 2023 Davide Belcastro. Tutti i diritti riservati

Questa tesi è stata composta con L^AT_EX e la classe Sapthesis.

Email dell'autore: davide_belcastro@yahoo.it

*"I propose to consider the question, '**Can machines think?**' This should begin with definitions of the meaning of the terms 'machine' and 'think.' The definitions might be framed so as to reflect so far as possible the normal use of the words, but this attitude is dangerous, If the meaning of the words 'machine' and 'think' are to be found by examining how they are commonly used it is difficult to escape the conclusion that the meaning and the answer to the question,..."*

A. M. Turing (1950) *Computing Machinery and Intelligence*. *Mind* 49: 433-460.

Indice

1 INTRODUZIONE	1
1.1 Introduzione al problema	1
1.2 Stato dell'arte	2
1.2.1 Segmentazione di tumori cerebrali	2
1.2.2 Classificazione di tumori cerebrali	2
1.2.3 Sfide e limitazioni attuali	2
1.3 Miglioramento proposto	4
1.3.1 Generalizzazione	4
1.3.2 Incremento dell'interpretabilità attraverso feature extraction	5
2 BACKGROUND	7
2.1 Artificial Intelligence, Machine Learning, Deep Learning e Computer Vision: principali differenze	7
2.2 MRI al cervello: caratteristiche	11
2.2.1 MRI al cervello con tumore	13
2.2.2 Difficoltà riscontrate	13
3 METODO PROPOSTO	17
3.1 Segmentazione del tumore	17
3.1.1 Skull Stripping	18
3.1.2 Spiegazione Filtro Canny nel dettaglio	21
3.1.3 Tumor mean color detection	28
3.1.4 Tumor segmentation	29
3.2 Classificazione del tumore	37
3.2.1 Acquisizione delle features più importanti	37
3.2.2 Modelli di Machine Learning: principali differenze	40
3.2.3 Creazione e allenamento del modello ML Random Forest	43
3.2.4 Altri modelli di Machine Learning, differenze di affidabilità	45
3.3 Gui	46
3.3.1 Scopo della Gui	46
3.3.2 Descrizione	46
4 ESPERIMENTI E RISULTATI OTTENUTI	51
4.0.1 Skull Stripping	51
4.0.2 Segmentazione	52
4.0.3 Classificazione	54

5 CONCLUSIONE ED ASPETTI FUTURI	55
5.1 Verso una ricostruzione 3d	56
Ringraziamenti	59
Bibliografia	59

Capitolo 1

INTRODUZIONE

1.1 Introduzione al problema

L'oggetto della tesi riguarda **Segmentazione e Classificazione di tumori cerebrali attraverso MRI**. Viene presentato un quadro generale del contesto medico e delle sfide che questa ricerca si propone di affrontare, delineando gli obiettivi, la rilevanza e il contributo atteso dalla tesi.

I tumori cerebrali rappresentano una delle patologie più complesse e potenzialmente letali per la salute umana. Le metodologie diagnostiche e terapeutiche si sono notevolmente evolute nel corso degli anni, ma la diagnosi e la gestione dei tumori cerebrali rimangono una sfida critica per i medici e i ricercatori.

L'MRI (Imaging a Risonanza Magnetica) è diventato uno strumento cruciale per la valutazione dei tumori cerebrali, poiché offre immagini dettagliate e non invasive dei tessuti cerebrali. La segmentazione e la classificazione automatica dei tumori cerebrali dalle MRI possono rivestire un ruolo fondamentale nella diagnosi precoce, nella pianificazione dei trattamenti e nel monitoraggio della progressione della malattia. L'obiettivo è sviluppare un algoritmo avanzato per la segmentazione automatica dei tumori cerebrali nelle MRI. Questo algoritmo dovrà essere in grado di individuare con precisione le aree tumorali all'interno delle immagini, fornendo una mappa dettagliata delle regioni interessate.

Inoltre, il nostro progetto mira a sviluppare un sistema di classificazione che permetta di distinguere tra tumori di tipo:

- Meningioma (tipicamente benigno)
- Pituitary (tipicamente benigno)
- Glioma (tipicamente maligno)

Questo sistema dovrà analizzare le caratteristiche delle immagini segmentate e assegnare una classe specifica al tipo di tumore presente. La corretta classificazione dei tumori è essenziale per una diagnosi accurata e per definire il piano di trattamento più appropriato.

1.2 Stato dell'arte

Lo stato dell'arte riguardante gli algoritmi di segmentazione e classificazione di tumori cerebrali tramite MRI è in continua evoluzione grazie alla rapida crescita della ricerca nell'ambito dell'intelligenza artificiale e dell'apprendimento automatico. Ad oggi questi algoritmi hanno fatto importanti progressi ma ci sono ancora sfide da superare per raggiungere una piena maturità e affidabilità.

1.2.1 Segmentazione di tumori cerebrali

Gli algoritmi di segmentazione si concentrano sull'individuazione e l'isolamento delle regioni tumorali nelle MRI. Le tecniche tradizionali di segmentazione, come le soglie e la region growing, hanno limitazioni nell'affrontare la complessità e la variabilità delle immagini cerebrali. Tuttavia negli ultimi anni, sono stati sviluppati approcci basati su algoritmi di apprendimento automatico, come reti neurali convoluzionali (CNN), che hanno dimostrato di essere promettenti per la segmentazione dei tumori cerebrali.

I modelli di CNN, addestrati su grandi dataset di MRI, sono capaci di apprendere automaticamente caratteristiche rilevanti nei dati e di produrre segmentazioni più precise rispetto agli approcci tradizionali. Questi algoritmi possono individuare con accuratezza le aree tumorali.

1.2.2 Classificazione di tumori cerebrali

La classificazione dei tumori cerebrali è un'attività complessa poiché esistono diverse tipologie di tumori con caratteristiche molto diverse. Le tecniche di classificazione possono distinguersi in tecniche tradizionali come l'estrazione manuale di caratteristiche (feature extraction) o tecniche più sofisticate come le CNN.

Questi modelli possono apprendere automaticamente le caratteristiche rilevanti dai dati e classificare le immagini in base al tipo di tumore. La disponibilità di dataset di addestramento adeguati è cruciale per ottenere modelli accurati.

1.2.3 Sfide e limitazioni attuali

Tra i principali problemi incontrati nello stato dell'arte troviamo:

- **Scarsa conoscenza per le reti neurali**

I dataset attualmente disponibili online per le MRI al cervello sono limitati e non riescono a coprire in modo adeguato la vasta gamma di tipi di tumori cerebrali, modalità di acquisizione delle MRI e proiezioni delle immagini. Questa carenza di dati rappresentativi ostacola l'addestramento efficiente delle reti neurali, poiché tali reti richiedono un insieme diversificato e completo di esempi per generalizzare in modo accurato su situazioni diverse. La mancanza di un dataset rappresentativo e completo compromette la capacità delle reti neurali di svolgere analisi diagnostiche e predittive di alta qualità nel contesto dei tumori cerebrali.

Le MRI illustrate precedentemente e quelle che verranno illustrate in seguito sono prese da due dataset che abbiamo scaricato.

Le immagini contenute nei dataset sono state usate per addestrare i modelli di Machine Learning e per analizzare gli output del programma di segmentazione.

- Il primo dataset è stato scaricato da Kaggle, che si trova a questo link, fonte [6]. Questo dataset contiene 3264 immagini suddivisi in due directory: "Testing" e "Training". Ogni directory contiene altre 4 directory: glioma, pituitary, meningioma, no-tumor. Il formato delle immagini è **jpg**.
- Il secondo dataset è stato scaricato da Figshare, che si trova a questo link, fonte [7]. Questa directory contiene 3064 immagini T1-MRI con contrasto eseguite su 233 pazienti con 3 tipi di tumore differenti: meningioma (708), glioma (1426) e pituitary (930).

Il dataset è diviso in 4 directory contenenti ciascuna 766 file.

Ogni file è in formato matlab (**.mat**) e contiene le seguenti informazioni

- * *cjdata.label*: 1 per meningioma, 2 per glioma, 3 per tumore pituitario
- * *cjdata.PID*: ID del paziente
- * *cjdata.image*: dati dell'immagine
- * *cjdata.tumorBorder*: contorno del tumore
- * *cjdata.tumorMask*: un'immagine binaria con in nero il cervello e in bianco la regione del tumore

Vengono mostrare le ultime righe del file README.txt presente nel dataset.

Questi dati sono stati utilizzati nei seguenti articoli scientifici Cheng, Jun, et al. "Enhanced Performance of Brain Tumor Classification via Tumor Region Augmentation and Partition." PloS one 10.10 (2015). Cheng, Jun, et al. "Retrieval of Brain Tumors by Adaptive Spatial Pooling and Fisher Vector Representation." PloS one 11.6 (2016). I codici sorgenti Matlab sono disponibili su GitHub.

- **Interpretabilità**

Molti modelli di apprendimento automatico, in particolare le CNN, possono essere considerati *scatole nere*, il che significa che la loro logica di ragionamento interna non è facilmente interpretabile dagli esseri umani. Ciò solleva questioni riguardanti l'interpretazione delle decisioni dei modelli, ovvero pongono dubbi sulla capacità di comprendere e spiegare il motivo per cui prendono determinate decisioni.

La Fig.1.1 illustra lo schema logico utilizzato dagli algoritmi di classificazione che implementano reti neurali.

E' possibile notare come la classificazione tramite reti neurali utilizzi tecniche non note ai professionisti nel campo della medicina.

Quest'ultimi possono solo inserire una MRI al programma che restituirà in output il presunto tipo di tumore.

In conclusione, gli algoritmi di segmentazione e classificazione dei tumori cerebrali tramite MRI hanno compiuto importanti passi avanti, ma esistono ancora sfide e

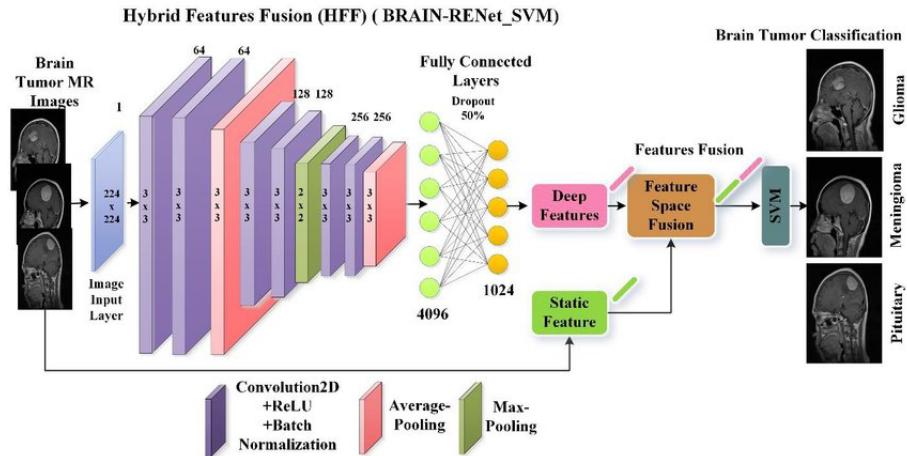


Figura 1.1. Classificazione con reti neurali, fonte [1]

limitazioni da superare. La continua ricerca e lo sviluppo di nuove tecniche sono essenziali per migliorare l'efficacia e l'affidabilità di questi strumenti, rendendoli preziosi ausili per i medici nella diagnosi e nella gestione dei tumori cerebrali.

1.3 Miglioramento proposto

Per affrontare le limitazioni esistenti e migliorare l'accuratezza nella segmentazione e classificazione di tumori cerebrali, abbiamo introdotto un nuovo approccio nell'ambito di questo progetto. L'approccio proposto si basa su un metodo innovativo che sfrutta tecniche di elaborazione d'immagine per analizzare le MRI cerebrali senza l'uso di una rete neurale.

1.3.1 Generalizzazione

La difficoltà delle CNN di allenarsi adeguatamente per mancanza di dataset può causare delle analisi errate in quanto le MRI sono molto diverse tra loro, per questo abbiamo usato una tecnica di *skull stripping* per eliminare il tessuto osseo nelle MRI, indipendentemente dalla vista del cervello. Questo permette di eliminare ogni informazione aggiuntiva inutile nello studio del tumore che potrebbe portare ad un'analisi errata.

In algoritmi di visione artificiale, lo studio di immagini che contengono solo il tessuto cerebrale risulta più affidabile rispetto all'analisi di una MRI completa.

Inoltre, il metodo fa uso di filtri e analisi dell'istogramma per comprendere il range di colori nel tessuto cerebrale, consentendo di identificare se il tumore è **iperintenso** (più chiaro) o **ipointenso** (più scuro) rispetto al tessuto cerebrale circostante. Questo passaggio informativo diventa fondamentale per la successiva fase di segmentazione del tumore.

Il nuovo approccio proposto è stato sperimentato e i risultati ottenuti dimostrano un significativo miglioramento nell'accuratezza della segmentazione dei tumori cerebrali rispetto ai metodi tradizionali basati su CNN. L'eliminazione delle dipendenze

dalle diverse viste e metodi di acquisizione ha dimostrato di contribuire in modo significativo all'ottenimento di risultati più coerenti e affidabili.

1.3.2 Incremento dell'interpretabilità attraverso feature extraction

Nell'ambito della classificazione dei tumori cerebrali, l'interpretabilità delle decisioni del modello è di fondamentale importanza per la fiducia dei medici nell'utilizzo di tali sistemi in contesti clinici.

Al fine di aumentare l'interpretabilità, abbiamo scelto di utilizzare tecniche di feature extraction, consentendo al programmatore e al medico di selezionare manualmente le caratteristiche rilevanti per la classificazione. Questo approccio ha permesso di identificare specifici parametri e attributi nei dati di imaging che possono essere interpretati dai professionisti.

Scalabilità e migliorabilità

Un ulteriore vantaggio di questo approccio consiste nella scalabilità e nella migliorabilità del sistema proposto. L'utilizzo di feature extraction e algoritmi di regolarizzazione, invece di reti neurali complesse, rendono il modello più estendibile con nuove features in futuro.

Ciò conferisce una maggiore agilità al sistema e la possibilità di adattarlo alle nuove sfide emergenti nel campo dell'imaging medico.

In conclusione, il contributo di questa tesi si focalizza sull'introduzione di un approccio innovativo nella segmentazione e classificazione di tumori cerebrali, che supera le limitazioni dei metodi esistenti dovute alla mancanza di dataset ben organizzati e alla diversità delle MRI. I risultati promettenti ottenuti sottolineano l'efficacia dell'approccio proposto e il potenziale impatto positivo che potrebbe avere nella pratica medica, migliorando la precisione delle diagnosi e l'efficacia dei trattamenti. L'intero programma è stato implementato grazie al linguaggio di programmazione Python insieme alla libreria **OpenCV** per sviluppare algoritmi di segmentazione. Nella classificazione abbiamo utilizzato approcci di Artificial Intelligence e Machine Learning grazie all'utilizzo della libreria **Scikit-Learn** per distinguere tra i diversi tipi di tumore.

Capitolo 2

BACKGROUND

2.1 Artificial Intelligence, Machine Learning, Deep Learning e Computer Vision: principali differenze

I termini Artificial Intelligence (intelligenza artificiale), Machine Learning (apprendimento automatico), Deep Learning (apprendimento profondo) e Computer Vision (visione artificiale), negli ultimi anni, hanno avuto un forte impatto mediatico.

Per avere un quadro più chiaro della situazione è opportuno dare una definizione ad ognuno di questi termini e descrivere come sono collegati tra loro.

- Artificial Intelligence (AI):

L'Intelligenza Artificiale è un campo dell'informatica che si occupa di creare sistemi o programmi in grado di svolgere attività che richiedono intelligenza umana. L'obiettivo principale dell'AI è sviluppare algoritmi o modelli che possano apprendere, ragionare, prendere decisioni e risolvere problemi in modo autonomo.

- Machine Learning (ML):

Il Machine Learning è un sottoinsieme dell'Intelligenza Artificiale basato sullo sviluppo di algoritmi che consentono ai computer di apprendere automaticamente dai dati e migliorare le prestazioni nel tempo senza essere esplicitamente programmati. Il ML si basa sull'identificazione di modelli e relazioni nei dati per prendere decisioni o fare previsioni.

Gli algoritmi di Machine Learning utilizzano apprendimenti che possono essere di diversi tipi:

- **Apprendimento supervisionato (Supervised learning)**

Questi algoritmi vengono addestrati utilizzando un dataset in cui ogni record è associato a una corrispondente etichetta o output desiderato.

Gli algoritmi di apprendimento supervisionato costruiscono una funzione che mappa ogni record alla propria etichetta, in modo da poter fare previsioni su nuovi dati non etichettati. Nella Fig.2.1 viene illustrata la logica dell'algoritmo.

L'algoritmo esamina i dati di input cercando caratteristiche (*features*) che sono fondamentali per il tipo di dati che si sta analizzando, nella fig.2.1

l'algoritmo deve distinguere tra immagini contententi mele e banane, sarà di rilevanza analizzare il colore, la forma della figura presente in ogni immagine che ha nel dataset di allenamento. Dopo aver costruito la funzione che associa queste features alla corrispettiva etichetta, l'algoritmo sarà in grado di ricevere in input un nuovo record che non ha mai visto e, dopo aver calcolato le features ed eseguito la funzione, potrà fare una previsione sulla possibile etichetta di appartenenza. Questo tipo di apprendimento viene utilizzato in diversi contesti come per esempio la **Classificazione di immagini**.

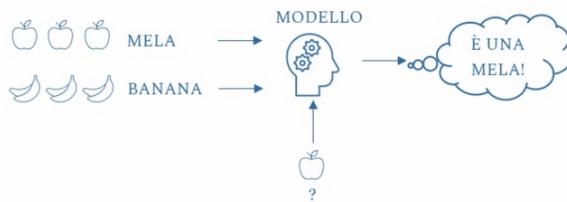


Figura 2.1. Supervisionato, fonte [13]

- Apprendimento non supervisionato(Unsupervised learning)

Questi algoritmi non hanno un dataset etichettato, sono gli algoritmi stessi che, analizzando i dati, cercano delle relazioni tra gli stessi e generano quelli che possono essere possibili raggruppamenti. Il risultato sarà un raggruppamento dei dati secondo caratteristiche simili.

Nella Fig.2.2 è possibile vedere come il modello ha una serie di dati in input con cui viene allenato che non hanno associata un'etichetta. Si limita a raggruppare elementi con caratteristiche affini senza sapere se si tratta di mele, banane o fragole.

Algoritimi di clusterizzazione come il K-Mean Clustering utilizzano questo tipo di apprendimento.

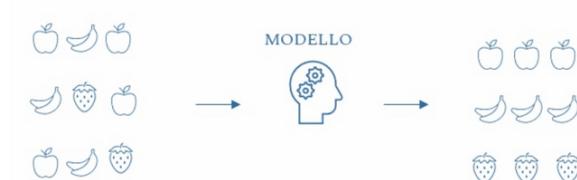


Figura 2.2. Non supervisionato, fonte [13]

– **Apprendimento semi-supervisionato (Semi-supervised learning)**

Questo apprendimento rappresenta una soluzione intermedia tra apprendimento supervisionato e non supervisionato. Questi algoritmi cercano di apprendere utilizzando un dataset etichettato insieme ad un grande dataset non etichettato.

Molto spesso capita di avere a disposizione molti dati non etichettati, tipicamente a causa del fatto che questo lavoro deve essere per forza eseguito manualmente. In questo caso si può pensare di etichettare solo una parte di dati.

– **Apprendimento per rinforzo (Reinforcement learning)**

Questo apprendimento è un approccio più recente in cui si procede *per tentativi*. L'algoritmo comincia avendo scarsissima conoscenza del problema e, di conseguenza, produrrà un risultato pseudo casuale. Andando avanti nella fase di training, continuerà a produrre risultati, se quest'ultimi saranno corretti riceverà un punteggio positivo, in caso contrario un punteggio negativo. Cercando di seguire la strada che porta ad ottenere punti positivi alla fine otterrà un modello ottimale.

- Deep Learning (DL):

Il Deep Learning è una branca del Machine Learning che utilizza reti neurali artificiali per l'apprendimento automatico.

Le reti neurali sono costituite da numerosi strati di neuroni artificiali che

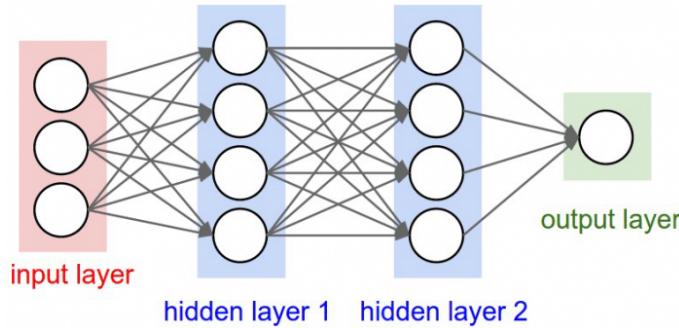


Figura 2.3. Rete Neurale Artificiale

lavorano in modo collaborativo per effettuare previsioni. Il DL è particolarmente efficace nel riconoscimento di immagini, nel riconoscimento del linguaggio naturale e in altre applicazioni che richiedono una comprensione avanzata dei dati.

Esempio di reti neurali sono le CNN (Convolutional Neural Network).

Vedere fig.2.3

- Computer Vision (CV):

La Computer Vision è una sottodisciplina dell'Intelligenza Artificiale che si concentra sull'elaborazione e l'interpretazione delle immagini e dei video da parte dei computer. L'obiettivo della CV è sviluppare algoritmi e modelli che permettano ai computer di *vedere* e comprendere il contenuto visivo come gli esseri umani. La CV viene utilizzata in diverse applicazioni, come il riconoscimento facciale, la segmentazione delle immagini, l'analisi delle scene e molto altro.

Può utilizzare tecniche di Machine Learning e Deep Learning per raggiungere l'obiettivo.

In fig.2.4 è possibile vedere a livello grafico quanto appena descritto

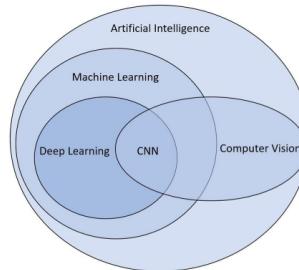


Figura 2.4. Gerarchia AI, fonte [4]

2.2 MRI al cervello: caratteristiche

La risonanza magnetica (MRI, Magnetic Resonance Imaging) è una tecnica di imaging medico che utilizza campi magnetici e onde radio per creare immagini dettagliate degli organi e dei tessuti all'interno del corpo. Nella MRI al cervello viene applicata questa tecnica per ottenere immagini dettagliate del cervello e delle sue strutture. Le caratteristiche principali della MRI al cervello includono:

- **Differenze di proiezione/vista del cervello**

Il cervello può essere scansionato tra 3 viste differenti, ognuna di queste permette di vedere meglio il cervello da una determinata angolazione.

Le MRI che si generano saranno diverse a seconda dell'angolazione.

Le diverse angolazioni possono far variare per il referto restituito, quantità visiva di tessuto osseo, intensità luminosa ed altro.

Le tre proiezioni sono:

- **Sagittale**

La vista sagittale nelle immagini MRI del cervello fornisce una visione laterale del cervello. Le immagini sono acquisite in modo tale che l'asse antero-posteriore sia parallelo al piano verticale del corpo, passando attraverso i due emisferi cerebrali. In questa vista è possibile osservare le strutture cerebrali dal lato, come il corpo calloso, il tronco cerebrale e il cervelletto. È particolarmente utile per valutare le anomalie laterali o asimmetriche nel cervello e per studiare l'anatomia delle strutture mediali.

- **Coronale**

La vista coronale nelle immagini MRI del cervello fornisce una sua visione frontale. Le immagini sono acquisite in modo tale che l'asse antero-posteriore sia perpendicolare al piano verticale del corpo, passando da fronte a nuca. In questa vista è possibile osservare le strutture cerebrali da una prospettiva frontale, come i lobi frontali, parietali, temporali e occipitali. Si possono anche visualizzare le strutture del sistema ventricolare, come i ventricoli laterali. La vista coronale è particolarmente utile per valutare le lesioni cerebrali, come i tumori o le emorragie, e per identificare la loro posizione e l'estensione all'interno del cervello.

- **Assiale**

La vista assiale nelle immagini MRI del cervello fornisce una sua visione orizzontale. Le immagini sono acquisite in modo tale che gli assi antero-posteriore e destra-sinistra siano paralleli al piano orizzontale del corpo, passando da sotto il mento verso l'alto. In questa vista è possibile osservare le strutture cerebrali da una prospettiva orizzontale, con un taglio che attraversa il cervello in modo trasversale. Si possono visualizzare le strutture come il corpo calloso, il tronco cerebrale, il cervelletto e i ventricoli cerebrali. La vista assiale è particolarmente utile per valutare l'anatomia delle strutture cerebrali nella direzione orizzontale e per identificare eventuali lesioni o anomalie.

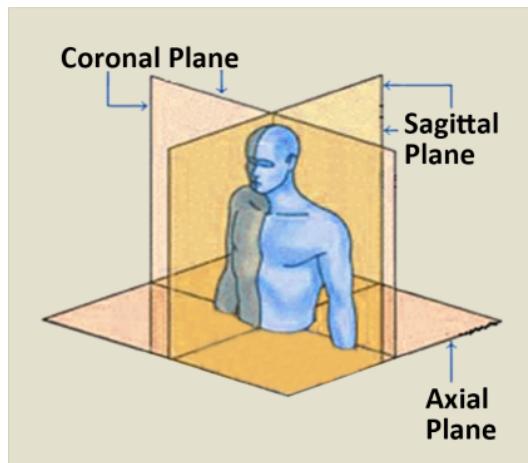


Figura 2.5. MRI views, fonte [8]

- **Differenze di intensità**

Le MRI al cervello possono presentare differenze di intensità luminosa basate sulla tecnologia utilizzata. Le sequenze T1 e T2 sono due sequenze comuni utilizzate nella MRI al cervello e possono fornire informazioni contrastanti sulla composizione dei tessuti.

- **Sequenza T1**

Le immagini T1 sono caratterizzate da un buon contrasto tra i tessuti molli, come la sostanza grigia e bianca del cervello. In queste immagini, la sostanza grigia appare più scura rispetto alla sostanza bianca. Le immagini T1 sono utili per valutare l'anatomia del cervello, come la forma e la posizione delle strutture cerebrali.

- **Sequenza T2**

Le immagini T2 forniscono un migliore contrasto tra i tessuti patologici o alterati. Nelle immagini T2, la sostanza grigia appare più chiara rispetto alla sostanza bianca. Queste immagini sono utili per individuare ed evidenziare lesioni, infiammazioni, edemi e altre anomalie.

- **Mezzo di contrasto**

In alcuni casi può essere utilizzato un mezzo di contrasto nella MRI al cervello per evidenziare ulteriormente determinate strutture o patologie. Il mezzo di contrasto viene iniettato nel paziente e può evidenziare vasi sanguigni, tumori o lesioni che possono non essere chiaramente visibili senza il contrasto.

La scelta tra le sequenze T1 e T2 dipende dal tipo di patologia sospettata e dagli obiettivi specifici dell'esame.

2.2.1 MRI al cervello con tumore

Le MRI al cervello con tumore possono presentare una varietà di caratteristiche che differiscono da un caso all'altro.

Ecco alcune delle differenze che possono essere riscontrate:

- **Intensità dei pixel**

I tumori cerebrali possono variare nell'intensità dei pixel nelle MRI. Questa variazione dipende dalla composizione del tumore, dalla sua vascularizzazione, dal tipo di tumore e da altri fattori. Alcuni tumori possono apparire **iperintensi**, cioè più luminosi rispetto al tessuto circostante, mentre altri possono apparire **ipointensi**, cioè più scuri rispetto al tessuto circostante.

- **Dimensione**

I tumori cerebrali possono variare notevolmente in termini di dimensione, dalla presenza di piccole lesioni a tumori di dimensioni più grandi.

- **Forma**

I tumori cerebrali possono assumere diverse forme, come rotonda, ovale, irregolare o lobulata. La forma del tumore può influenzare la sua identificazione e caratterizzazione.

- **Posizione**

I tumori cerebrali possono svilupparsi in varie regioni del cervello, come il lobo frontale, parietale, temporale, occipitale o nel tronco cerebrale.

2.2.2 Difficoltà riscontrate

Creare un algoritmo di computer vision che funzioni per qualsiasi tipo di MRI al cervello, indipendentemente dal tipo di sequenza utilizzata o dalla vista dell'immagine, ha presentato diverse problematiche. Ecco alcune delle problematiche che sono sorte durante lo sviluppo dell'intero progetto:

- **Variazioni di intensità luminosa**

Le MRI del cervello possono avere una vasta gamma di intensità luminose a causa di diversi fattori come il tipo di sequenza utilizzata (T1, T2, FLAIR, etc.), le impostazioni di acquisizione e la presenza di tessuto osseo. Questa variazione di intensità può rendere difficile stabilire una soglia di segmentazione universale o definire regole fisse per l'analisi della MRI attraverso un algoritmo.

- **Dimensione e forma del cervello**

Le dimensioni del cervello possono variare da individuo a individuo e ciò può influire sulla scala e sulla proporzione delle strutture cerebrali nelle MRI. Inoltre la forma del cervello può presentare variazioni individuali che rendono complessa l'elaborazione automatica delle immagini.

- **Presenza di tessuto osseo**

Nelle MRI del cervello il tessuto osseo circostante può creare artefatti e influire sull'intensità dei pixel nelle vicinanze. Questo può causare confusione nella segmentazione dei tumori, in quanto anche il tessuto tumorale, come il tessuto osseo, può apparire con diverse intensità luminose rispetto al tessuto cerebrale.

- **Variazioni nell'aspetto dei tumori**

I tumori cerebrali possono presentare una vasta gamma di aspetti visivi, come variazioni di forma, margini sfumati o irregolari, presenza di cisti o necrosi. Questa diversità di aspetto può rendere difficile l'identificazione e la classificazione automatica dei tumori.

- **Presenza di strutture anatomiche**

Il cervello è costituito da alcune strutture anatomiche più o meno visibili nella MRI a seconda della proiezione. Queste strutture hanno un'intensità di pixel differente dal tessuto cerebrale e questo comporta una difficoltà per l'implementazione di un programma di computer vision che sia in grado di distinguere strutture anatomiche da tumori. Nelle MRI generate da una vista assiale è visibile il *corpo calloso* ovvero una struttura anatomica centrale del cervello che il programma di computer vision potrebbe scambiare per un tumore. Lo stesso corpo calloso, inoltre, può essere iperintenso o ipointenso rispetto al tessuto cerebrale.

Scrivere un programma che sia in grado di distinguere in maniera autonoma queste strutture anatomiche dai tumori è stato molto complesso.

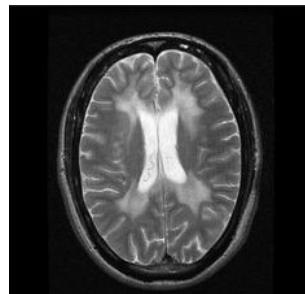


Figura 2.6. Esempio di MRI con il corpo calloso iperintenso

- **Variabilità dei dataset**

I dataset disponibili per l'addestramento di algoritmi di computer vision per l'analisi presentano una grande varietà di MRI che non vengono classificati in base alle loro caratteristiche comuni (come per esempio il tipo di proiezione). Questa variazione può rendere complesso l'adattamento del modello a diversi tipi di immagini, richiedendo una maggiore generalizzazione dell'algoritmo.

Per affrontare queste problematiche è stato necessario sviluppare algoritmi di computer vision che fossero in grado di adattarsi e gestire la diversità delle MRI al cervello. Nella figura in basso è possibile vedere le tre differenti MRI generate dalla tre viste. E' possibile notare come tutte e tre hanno un'intensità di pixel differente.

La vista Sagittale è più scura rispetto alle altre mentre la vista Coronale presenta punti di luce molto chiari ai bordi (tessuto osseo), punti molto scuri all'interno (tessuto cerebrale). La vista Assiale, infine, ha un colore abbastanza omogeneo all'interno di tutta la scansione dove è ben visibile anche il corpo calloso ipointenso.

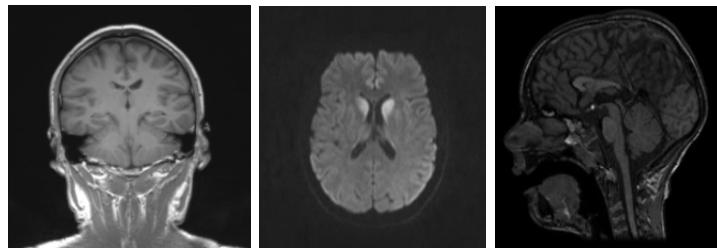


Figura 2.7. Da sx a dx: coronale, assiale, sagittale.

Capitolo 3

METODO PROPOSTO

3.1 Segmentazione del tumore

Primo task del progetto: **Segmentare il tumore.**

Per ottenere la segmentazione del tumore abbiamo diviso il problema in 3 fasi.

La prima fase, denominata **Skull Stripping**, consiste nell'estrare il tessuto cerebrale eliminando il tessuto osseo.

Un programma che sia in grado di trovare un tumore all'interno del cervello deve basarsi sulla variazione di intensità luminosa che genera il tumore stesso rispetto al resto del tessuto cerebrale.

Il tumore può essere più scuro (ipointenso) o più chiaro (iperintenso) rispetto al tessuto cerebrale.

La presenza del tessuto osseo può ingannare il programma in quanto anche il tessuto osseo genera variazioni di intensità luminose. Una variazione di intensità luminosa dei pixel, presente all'interno di un'immagine con solo il tessuto cerebrale, se unita ad una determinata grandezza e circolarità dei pixel può indicare, con una certa probabilità, la presenza del tumore.

Per questo motivo il processo di analisi di segmentazione comincia con l'eliminazione del tessuto osseo.

La seconda fase, denominata **Tumor mean color detection**, consiste nel restituire il colore medio del tumore (qualora sia presente).

Questa fase riceve in input l'immagine ritornata dalla prima fase e analizza zone con intensità di luce differenti rispetto al colore medio del tessuto cerebrale.

Se il tumore non è presente, questa fase non troverà grandi differenze di intensità luminosa e ritornerà un'intensità di pixel molto vicina al colore medio del tessuto cerebrale. La fase successiva sarà in grado di capire che non ci sono tumori presenti nell'immagine prendendo in input questo valore ed unito ad altri fattori.

La terza fase, denominata **Tumor Segmentation**, riceve in input l'immagine ritornata dalla prima fase e il colore medio ritornato dalla seconda fase. Si occupa di trovare il contorno che rappresenta il tumore.

Se non trova nessun contorno anomalo, in termini di differenza di intensità luminosa, grandezza, circolarità e altro, specifica che non ha trovato il tumore.

3.1.1 Skull Stripping

Estrazione del tessuto cerebrale da una MRI

Descrizione algoritmo

Illustriamo nel dettaglio l'algoritmo attraverso diversi step.

- **Acquisizione della MRI**

L'algoritmo riceve in input l'immagine iniziale, che viene trasformata in scala di grigi (un solo canale) in modo da ridurre la dimensionalità dei dati, semplificando così l'elaborazione e riducendo il costo computazionale.

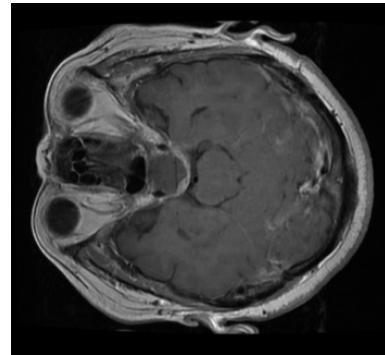


Figura 3.1. Acquisizione MRI

- **Modifica pixel**

Tutti i pixel con valore minore o uguale a 15 vengono impostati a 255.

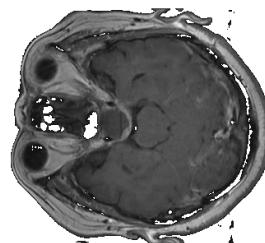


Figura 3.2. Modifica pixel

- **Applicazione del filtro di threshold**

Tutti i pixel con un'intensità uguale a 255 vengono impostati a 0, mentre tutti gli altri vengono impostati a 255. Questo passaggio serve per creare una maschera bianca (pixel uguali a 255), che rappresenta l'intera sagoma presente nella MRI, su sfondo nero (pixel uguali a 0). In questo modo è possibile trovare il contorno più esterno della sagoma, da cui è possibile calcolarne le dimensioni.

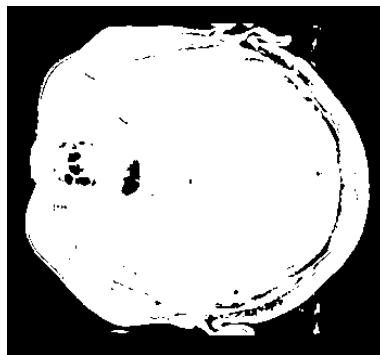


Figura 3.3. Output thresold

Definizione del filtro di thresold

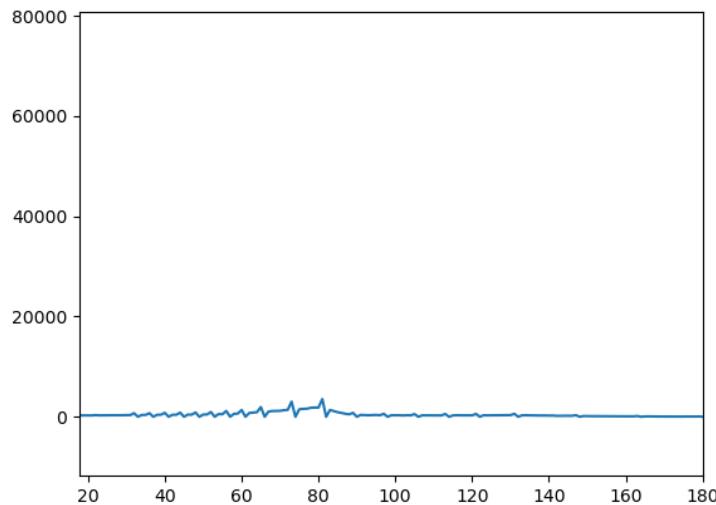
Il filtro di threshold è un metodo di segmentazione dell'immagine che assegna un valore fisso (solitamente 0 o 255) ai pixel dell'immagine in base a una soglia di intensità (valore di thresold). L'obiettivo principale del filtro di threshold è separare le regioni di interesse (ad esempio, i contorni) dallo sfondo.

Il filtro di threshold utilizza una semplice operazione matematica: confronta il valore di intensità di ogni pixel con una soglia predefinita. Se il valore dell'intensità supera la soglia, il pixel viene assegnato al valore massimo (ad esempio 255); altrimenti viene assegnato al valore minimo (ad esempio 0).

- **Colore medio del cervello**

Viene creato un istogramma analizzando i pixel che si trovano dentro il contorno della sagoma trovata in precedenza. Non consideriamo i pixel troppo scuri (molto vicini allo 0) o troppo chiari (molto vicini a 255). Nell'istogramma in basso è possibile vedere come sia presente un numero maggiore di pixel di intensità intorno ai 70/80.

Possiamo assumere che il colore medio del cervello sia circa 75 essendo un valore intermedio tra i pixel che sono maggiormente presenti all'interno dell'immagine.



- **Applicazione del filtro di Canny**

Metodo sofisticato per il rilevamento dei contorni, la funzione Canny prende due parametri: soglia minima e soglia massima che sono utilizzati per determinare quali pixel dell'immagine vengono considerati bordi. Nel caso specifico, alla funzione Canny, gli vengono passati come soglia minima: il valore del colore medio del cervello sottratto di 10, come soglia massima il valore del colore medio del cervello aumentato di 10. Viene creato questo range in cui tutto ciò che non è compreso in questo range viene considerato bordo e, nell'immagine di output ritornata dal filtro di Canny questi pixel saranno uguali a 255, tutti gli altri uguali a 0. Vedere Fig. 3.4

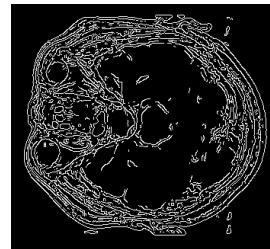


Figura 3.4. Output filtro Canny

3.1.2 Spiegazione Filtro Canny nel dettaglio

Il metodo di Canny produce bordi *connessi*.

L'approccio prevede le seguenti fasi:

- **Smoothing Gaussiano dell'immagine**

Un filtro Gaussiano viene utilizzato per applicare un effetto di sfocatura all'immagine, i risultati del filtro Gaussiano dipendono dalla scelta di σ . La scelta del valore di σ dipende dal livello di rumore presente nell'immagine. Se l'immagine è molto rumorosa, è necessario un valore di σ più grande per ottenere una sfocatura maggiore e ridurre il rumore. Al contrario, se l'immagine ha poco rumore, un valore di σ più piccolo può essere sufficiente. Questo perché σ , influenza la forma e la dimensione della matrice filtro (kernel) utilizzata nel filtro Gaussiano.

Il processo di sfocatura è molto semplice:

La matrice filtro(in questo esempio 3x3) contiene dei pesi. L'algoritmo inizia analizzando ogni pixel dell'immagine che vuole sfocare. Per ogni pixel, posiziona il centro della matrice filtro sul pixel stesso e moltiplica i valori dei pixel circostanti per i rispettivi pesi nella matrice filtro. Infine somma tutti i risultati. Questa somma rappresenta il nuovo valore del pixel di destinazione nell'immagine filtrata che verrà sostituito dal pixel corrente che sta analizzando nell'immagine di input.

La dimensione della matrice del filtro è determinata dalla scelta dell'utente e dipende dalla scala della sfocatura desiderata. Una dimensione comune per la matrice del filtro è 3x3 o 5x5, ma può variare a seconda delle esigenze specifiche dell'applicazione.

Per generare la matrice del filtro, si applica la formula della distribuzione gaussiana

$$G(x, y) = \frac{1}{2\pi\sigma^2} \cdot \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \quad (3.1)$$

dove $G(x, y)$ rappresenta il peso assegnato al pixel nella posizione (x, y) nella matrice del filtro, σ è il valore di sigma scelto dall'utente ed \exp è la costante di Nepero (2.71828).

Vedere un esempio in Fig. 3.5

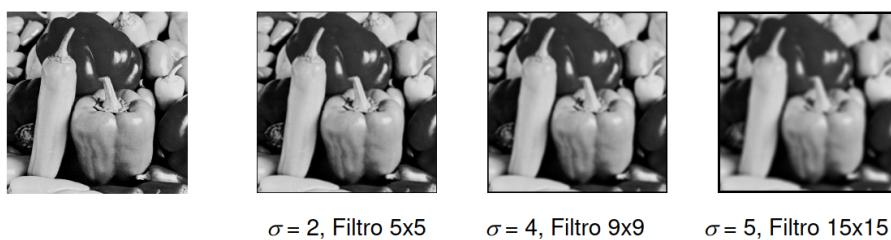


Figura 3.5. Esempio di filtro Gaussian,fonte [12].

– Calcolo del gradiente

Il calcolo del gradiente è un’operazione fondamentale nell’elaborazione delle immagini ed è utilizzato per identificare i cambiamenti di intensità dei pixel all’interno di un’immagine. Il gradiente di un’immagine misura l’intensità del cambiamento dei valori dei pixel, indicando le regioni in cui vi sono transizioni o bordi.

In generale, il calcolo del gradiente coinvolge il calcolo delle derivate parziali dell’immagine rispetto alle coordinate spaziali (x e y). Esistono diversi operatori che possono essere utilizzati per calcolare il gradiente, tra cui gli operatori di Roberts, di Prewitt e di Sobel.

L’obiettivo del calcolo del gradiente è identificare i punti in cui l’intensità dell’immagine cambia rapidamente. Questi punti corrispondono a bordi o transizioni significative tra diverse regioni dell’immagine. Il gradiente viene quindi utilizzato come base per l’individuazione dei contorni o per altre operazioni di analisi dell’immagine, come il rilevamento dei bordi, la segmentazione delle immagini o l’estrazione di caratteristiche.

In pratica, il calcolo del gradiente coinvolge la convoluzione dell’immagine originale con un operatore di derivata, che misura la variazione dell’intensità lungo le direzioni orizzontale e verticale. Questo produce due mappe del gradiente, una per la direzione orizzontale (derivata rispetto a x) e una per la direzione verticale (derivata rispetto a y). Queste mappe possono quindi essere combinate per ottenere una mappa del modulo del gradiente (rappresentata con $|\nabla|$), che rappresenta l’intensità complessiva dei cambiamenti di intensità nell’immagine.

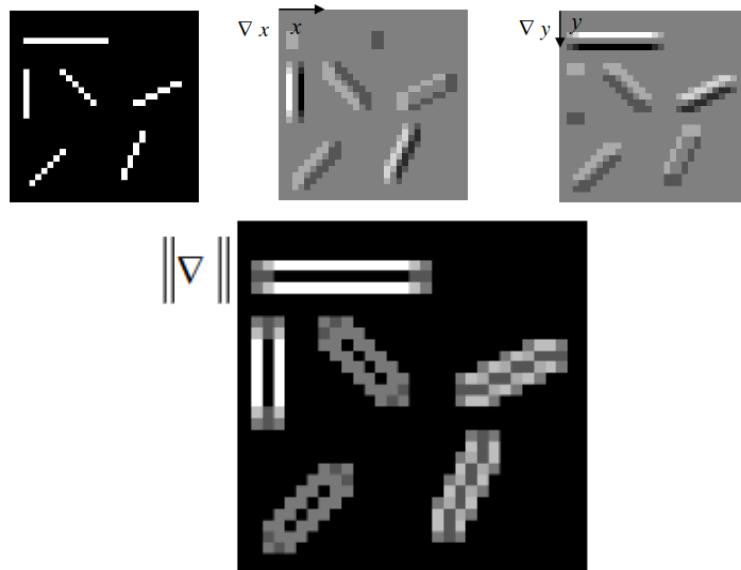


Figura 3.6. Calcolo del gradiente, fonte [12].

– **Soppressione dei non-massimi in direzione ortogonale all'edge**

La soppressione dei non massimi è una tecnica utilizzata nell'algoritmo di rilevamento dei bordi di Canny per sottolineare i punti di massimo nel gradiente dell'immagine e sopprimere tutti gli altri punti che non sono considerati punti di bordo. Questa tecnica aiuta a ottenere contorni sottili e ben definiti riducendo il numero di punti di bordo rilevati. Consente di conservare solo i punti che sono localmente massimi lungo la direzione del gradiente, eliminando così i punti che potrebbero essere parte di un bordo più grande. Questo contribuisce a ottenere una rappresentazione più accurata dei contorni nell'immagine.

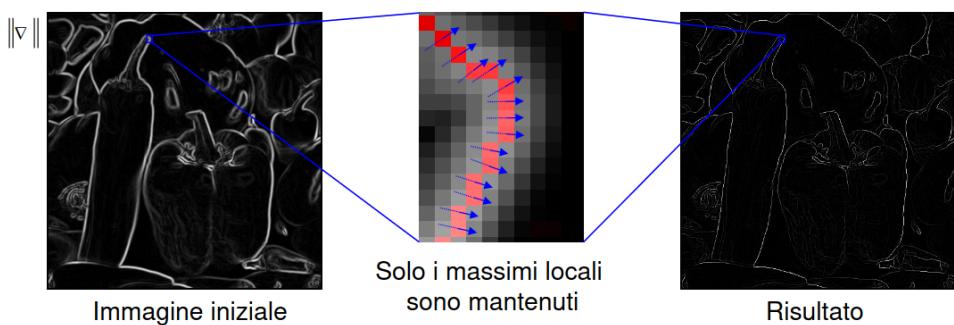


Figura 3.7. fonte [12].

– **Selezione degli edge significativi mediante isteresi**

Al fine di selezionare solo gli edge significativi (tralasciando edge “spuri”), ma evitando allo stesso tempo la frammentazione, si utilizza il concetto di isteresi: vengono impiegate due soglie:

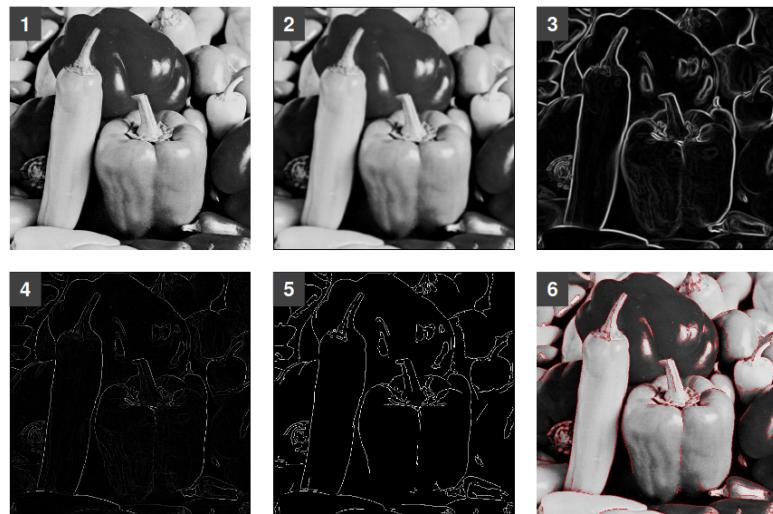
-**Soglia minima**

Questo valore rappresenta la soglia più bassa per l'intensità del gradiente. I pixel con intensità di gradiente al di sotto di questa soglia vengono considerati non bordi e vengono scartati.

-**Soglia massima**

Questo valore rappresenta la soglia più alta per l'intensità del gradiente. I pixel con intensità di gradiente al di sopra di questa soglia vengono considerati bordi forti.

Durante il processo di *isteresi*, i pixel con intensità di gradiente compresa tra la soglia minima e la soglia massima vengono considerati bordi deboli. Tuttavia, questi bordi deboli vengono mantenuti solo se sono collegati a bordi forti. In altre parole, i pixel deboli che sono adiacenti a pixel forti vengono considerati parte dei bordi, tutto il resto viene scartato. Esempio completo in fig.3.8



1) Immagine originale – 2) Smoothing gaussiano – 3) Modulo del gradiente
4) Soppressione non-massimi – 5) Selezione edge – 6) Edge sovrapposti a immagine originale

Figura 3.8. Esempio finale filtro di Canny,fonte [12].

- **Applicazione filtro dilate**

Il filtro di dilatazione, od operazione di dilatazione, è un'operazione morfologica utilizzata nell'elaborazione delle immagini per espandere o ingrandire le regioni luminose o i punti di interesse nell'immagine.

La dilatazione viene eseguita utilizzando un elemento strutturale, che può essere una forma geometrica predefinita come un quadrato, un cerchio o un rettangolo. L'elemento strutturale viene posizionato su ogni pixel dell'immagine e viene confrontato con la regione circostante. Se almeno un pixel nell'area di copertura dell'elemento strutturale ha un pixel di interesse, il pixel corrispondente nell'immagine di output viene impostato sul valore di interesse.

In pratica, la dilatazione *espande* i punti di interesse nell'immagine, aumentando la loro dimensione e connettività.

L'operazione di dilatazione può essere iterata più volte per ottenere un effetto di espansione più marcato. In ogni iterazione, i punti di interesse vengono dilatati ulteriormente.

In questo preciso contesto, i punti d'interesse sono tutti i pixel che il filtro di Canny, nello step precedente, considerava bordi.

L'obiettivo è quello di *dilatare* questi punti per creare delle regioni omogenee e collegare segmenti interrotti.

E' stato creato un algoritmo che sia in grado di automatizzare la scelta del numero di iterazioni che deve eseguire la funzione, andando a controllare l'espansione ad ogni iterazione, finché ci sono pixel d'interesse vicini e non collegati continua ad iterare. Vedere Fig.3.9

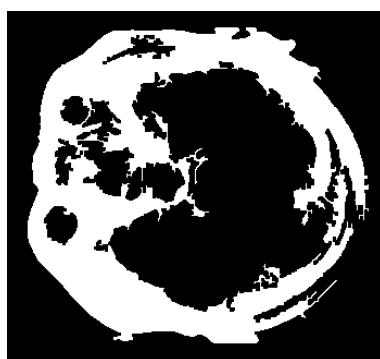


Figura 3.9. Dilate output

- **Eliminazioni di pixel non d'interesse**

Vengono impostati a 255 tutti i pixel che si trovano fuori dal contorno della sagoma trovata all'inizio. In questo modo isolo tutte le zone di pixel interamente nere (valore 0). Vedere Fig. 3.10.



Figura 3.10. Output

- **Scansione contorni**

Vengono analizzati i contorni delle zone nere, il contorno più grande che sia anche abbastanza circolare potrebbe essere il tessuto cerebrale.

- **Estrazione del contorno più grande e circolare**

Una volta confrontati tutti i contorni e scelto quello più *interessante*, applichiamo dei confronti, se il contorno trovato è troppo piccolo oppure ha dimensione maggiore o uguale al contorno della sagoma trovata in precedenza, con buona probabilità ci sarà stato un errore e il processo di skull stripping ritorna l'immagine iniziale (senza nessuna estrazione); se il contorno è più piccolo del contorno della sagoma ma non è abbastanza circolare potrebbe esserci stato un errore e anche in questo caso il programma ritorna l'immagine iniziale; se il contorno è circolare, è più piccolo del contorno iniziale possono succedere due casi.

-Il contorno è più grande di un threshold (valore soglia calcolato in base alla dimensione della sagoma), in questo caso, con buona probabilità, il programma ha trovato la zona cerebrale con una parte della zona ossea.

-Il contorno è più piccolo del threshold, in questo caso il programma potrebbe aver trovato solo il contorno cerebrale. Vedere Fig.3.11

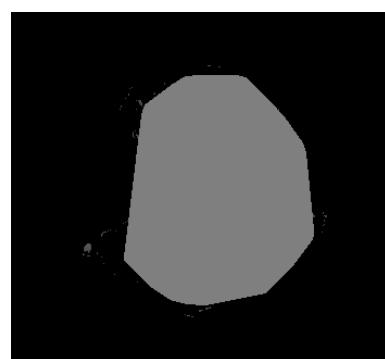


Figura 3.11. Output

- **Applicazione maschera**

Nei casi in cui il programma trova il tessuto cerebrale (con o senza una parte del tessuto osseo), viene applicata una maschera per prendere i pixel dentro questo contorno che si trovano nell'immagine iniziale. Vedere Fig. 3.12

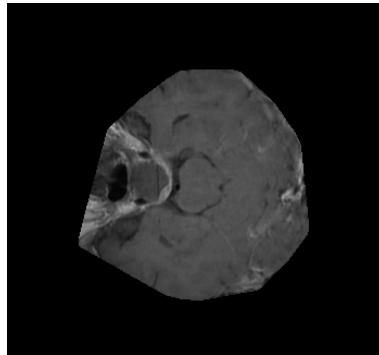


Figura 3.12. Output

Test effettuati

Per avere una stima in termini di affidabilità del processo di skull stripping, abbiamo effettuato dei test che mirano a controllare se il processo di estrazione del tessuto cerebrale funziona correttamente oppure no.

Abbiamo utilizzato l'intero dataset contenente le MRI dei pazienti e, per ogni MRI abbiamo eseguito il programma di skull stripping e, sulla base delle condizioni spiegate nella penultima fase, abbiamo calcolato la percentuale di immagini la cui estrazione avveniva

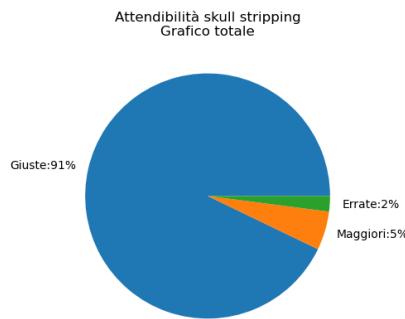
- Correttamente: il programma trova solo il tessuto cerebrale
- Maggiore: il programma trova tutto il tessuto cerebrale + parte del tessuto osseo
- Errata: il programma non è stato in grado di estrarre il tessuto cerebrale

La proprietà del programma stesso di *autovalutarsi*, con una certa probabilità, è fondamentale per gli step successivi durante il processo di segmentazione. Il programma applica filtri e funzioni differenti a seconda dei tre casi in cui crede di trovarsi (Giusta, Maggiore, Errata).

Quando il processo non è in grado di estrarre il tessuto cerebrale, il programma ritorna l'immagine iniziale che verrà elaborata, successivamente, in maniera differente (si tiene conto del fatto che è presente ancora il tessuto osseo).

Risultati ottenuti

Possiamo vedere il grafico che illustra i risultati ottenuti in questa fase. Il grafico è stato creato con un programma Python che va a leggere da un file csv in cui sono riportati i dati calcolati durante il test di affidabilità.



3.1.3 Tumor mean color detection

Questa fase ha l'obiettivo di analizzare l'immagine ritornata dalla prima fase (Skull Stripping) e di ritornare il colore medio del tumore.

Descrizione algoritmo

Più nel dettaglio, l'algoritmo tiene conto del colore medio del cervello calcolato nelle fasi precedenti, analizza l'immagine e, attraverso l'uso di un istogramma, seleziona le zone di pixel (sufficientemente grandi) che hanno un colore medio differente rispetto al colore medio del cervello.

Per ognuna di queste zone, calcola la differenza assoluta tra il colore medio del cervello e il colore medio della zona *i-esima*; Il colore medio del contorno *i-esimo* che ha la maggior differenza assoluta con il colore medio del cervello sarà il valore ritornato in output da questa fase.

La differenza assoluta è importante in quanto il tumore può essere più chiaro o più scuro rispetto al tessuto cerebrale, di conseguenza non possono essere fatte assunzioni sull'intensità dei pixel.

Se l'immagine non presenta nessun tumore, il programma ritornerà un valore che sarà molto vicino al colore medio del cervello.

Test effettuati

Per *testare* l'affidabilità di questa fase abbiamo preso il dataset che contiene, per ogni record, il contorno del tumore.

Per ogni MRI del dataset abbiamo calcolato il vero colore medio del tumore calcolato dal segmento del tumore già presente, ed abbiamo eseguito il programma con in input l'immagine corrente, se il valore restituito dal programma era abbastanza simile all'effettivo colore medio del tumore, l'esecuzione aveva avuto successo, fallimento in

caso contrario.

Nel caso in cui il tumore non fosse presente, abbiamo confrontato il colore medio del cervello con il valore ritornato dal programma.

Lo pseudocodice è il seguente:

- Esegue la funzione *get_color*
- Aumenta la variabile *totale* di uno
- Calcola da differenza assoluta tra il colore ritornato dalla funzione e il colore medio (vero) del tumore
- Confronta la differenza assoluta
- Se la differenza assoluta è minore o uguale a 10 aumenta il numero di test corretti
- Aumenta il numero di test errati in caso contrario
- In entrambi i casi scrive su un file csv il colore trovato e quello vero.

Risultati ottenuti

Nel grafico mostrato in Fig.3.13 possiamo vedere l'attendibilità di questa parte di programma che prende i dati dal file csv creato durante l'esecuzione dei test.

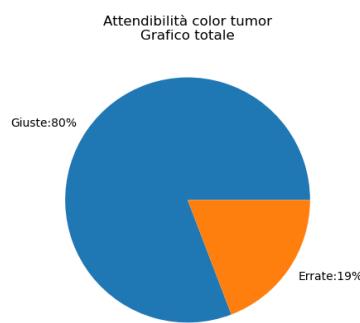


Figura 3.13. Grafico color tumor

3.1.4 Tumor segmentation

L'ultima fase del processo di segmentazione del tumore consiste nell'andare a trovare il tumore e segmentarlo, questa fase prende in input l'output delle due fasi precedenti. Utilizza l'algoritmo **K-Mean Clustering** per *clusterizzare* l'immagine.

Il cluster più *anomalo* ovvero che ha tutte le caratteristiche per essere considerato un tumore, viene segmentato e il programma ritorna l'immagine iniziale con il tumore segmentato.

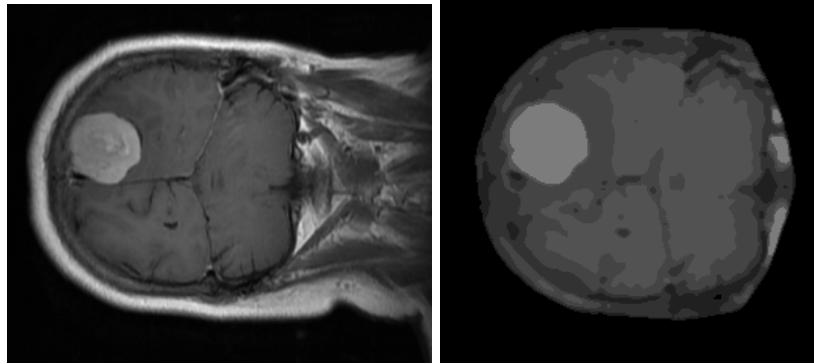
Se non sono presenti cluster *anomali*, il programma ritorna che non ha trovato tumori.

Descrizione algoritmo

L'algoritmo procede per step

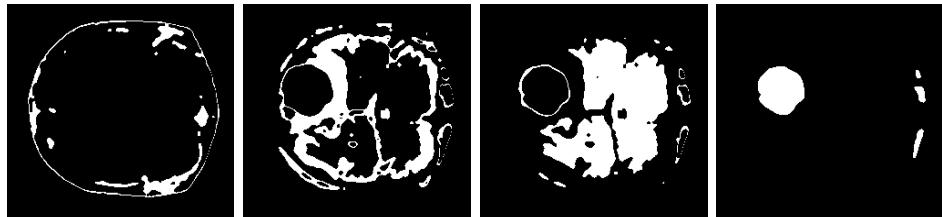
- **Step1**

Esegue K-Mean Clustering che ritorna un'immagine in cui tutti i pixel di una certa intensità (simili tra loro) vengono considerati pixel di uno stesso cluster, l'output dell'algoritmo è un'immagine in cui ogni cluster ha un colore che è diverso da tutti gli altri cluster.



- **Step2**

Il programma itera su ogni cluster, e per ognuno di essi crea una maschera totalmente nera in cui di bianco sono presenti solo i pixel del cluster *i-esimo*. Per ogni maschera creata, trova tutti i contorni al suo interno.



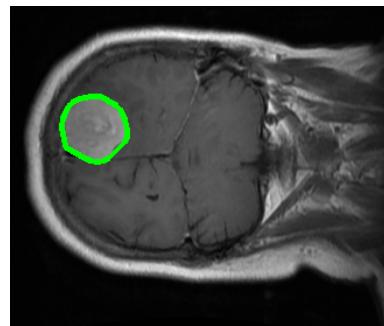
- **Step3**

Tutti i contorni che trova possono essere un possibile tumore, per trovare il vero tumore viene associato ad ogni contorno un valore che è il risultato di una media pesata tra diversi attributi: area, circolarità, differenza assoluta tra il colore medio del contorno e il colore medio del tumore calcolato nello step precedente (più la differenza è piccola e meglio è), e tanti altri.

Viene considerato il contorno che ha la media pesata più elevata, questa media viene confrontata con un valore soglia, se la media è sufficientemente elevata allora il programma ritorna il contorno, altrimenti vuol dire che non ha trovato tumori.

Questo perchè se il tumore non è presente, tutti i contorni che ha trovato avranno una media pesata molto bassa che non deve essere ritornata.

Limitazioni: non è in grado di trovare due o più tumori all'interno di una stessa MRI.



Approfondimento sull'algoritmo K-Mean Clustering

L'algoritmo K-Means è un algoritmo di clustering non supervisionato utilizzato per raggruppare un insieme di dati in cluster basati sulla loro similarità. A differenza degli algoritmi di apprendimento supervisionato, in cui i dati di addestramento sono etichettati con le risposte desiderate, l'algoritmo K-Means lavora solo sulla base dei dati di input senza alcuna informazione di output predefinita.

L'obiettivo dell'algoritmo K-Means è suddividere i dati in K cluster, dove K è un numero predefinito fornito in input. L'algoritmo cerca di minimizzare la somma dei quadrati delle distanze tra i punti di dati e i centroidi dei rispettivi cluster.

Il funzionamento dell'algoritmo K-Means è il seguente

- **Inizializzazione**

Seleziona casualmente K punti come centroidi iniziali dei cluster.

- **Assegnazione dei punti al cluster**

Per ogni punto nel dataset, calcola la sua distanza da ciascun centroide e assegna il punto al cluster il cui centroide è più vicino.

- **Aggiornamento dei centroidi**

Calcola il nuovo centroide di ogni cluster come il punto medio di tutti i punti assegnati a quel cluster.

- **While**

Ripete i passaggi 2 e 3 fino a quando non si raggiunge una condizione di convergenza (ad esempio, quando non ci sono ulteriori cambiamenti nella configurazione dei centroidi o quando viene raggiunto un numero massimo di iterazioni).

L'algoritmo K-Means non richiede etichette o supervisione esterna per creare i cluster. Invece, utilizza solo la struttura intrinseca dei dati per determinare la loro suddivisione in gruppi omogenei. Questo rende l'algoritmo K-Means molto versatile e ampiamente utilizzato in vari campi, come l'analisi dei dati, il riconoscimento di pattern, l'analisi delle immagini e altro ancora.

Nell'algoritmo di segmentazione quando viene chiamato l'algoritmo K-Mean Clustering non si sa a priori quale k scegliere, di conseguenza abbiamo implementato un metodo per automatizzare la scelta di k .

L'algoritmo usato per automatizzare la scelta di k è stato il *metodo del gomito*.

Elbow method

Il metodo del gomito (**elbow method**) è una tecnica utilizzata per determinare il numero ottimale di cluster da utilizzare nell'algoritmo K-Means o in altri algoritmi di clustering. Il nome *gomito* deriva dalla forma del grafico generato durante l'applicazione del metodo.

Il metodo del gomito coinvolge i seguenti passaggi

- Eseguire l'algoritmo K-Means su un dataset per un range di valori di K (numero di cluster) desiderati.
- Per ogni valore di K , calcola la somma dei quadrati delle distanze dei punti dai centroidi dei rispettivi cluster (detta anche varianza intra-cluster).
- Traccia un grafico che mostra la varianza intra-cluster in funzione del numero di cluster (K).
- Osserva il grafico e cerca il punto in cui la varianza intra-cluster inizia a diminuire in modo meno significativo. Questo punto corrisponde al "gomito" del grafico.
- Il numero di cluster corrispondente al punto del gomito è considerato come il numero ottimale di cluster da utilizzare per il dataset.

L'idea alla base del metodo del gomito è che l'aggiunta di cluster aggiuntivi inizialmente porterà a una significativa riduzione della varianza intra-cluster, poiché i cluster diventano più specifici. Tuttavia, a un certo punto, l'aggiunta di ulteriori cluster avrà un impatto sempre minore sulla riduzione della varianza, e il grafico formerà una sorta di *gomito* o *curva di flessione*. Questo punto rappresenta un equilibrio tra complessità del modello (numero di cluster) e la sua capacità di spiegare la varianza dei dati.

Possiamo vedere un esempio in Fig.3.14 in questo specifico esempio, il k trovato è 4, questo vuol dire che l'algoritmo suddividerà in 4 cluster l'immagine. Con buona probabilità per questa immagine 4 è il numero ideale per clusterizzare. Dopo aver creato questo grafico abbiamo implementato una funzione che fosse in grado di leggere l'istogramma e restituire il k ideale, per risolvere questo compito abbiamo analizzato le differenze sull'asse verticale presenti tra due k adiacenti, ad esempio in questo grafico il programma comincia analizzando la differenza tra $k=2$ e $k=3$, prosegue analizzando tra $k=3$ e $k=4$, e così via, ad ogni confronto memorizza la differenza. Quando confronta $k=4$ con $k=5$ noterà una riduzione sostanziale della differenza rispetto al valore precedente che andrà a stabilizzarsi nelle differenze successive. Sulla base di questa logica il programma ritornerà $k=4$ essendo il punto in cui la differenza comincia a stabilizzarsi.

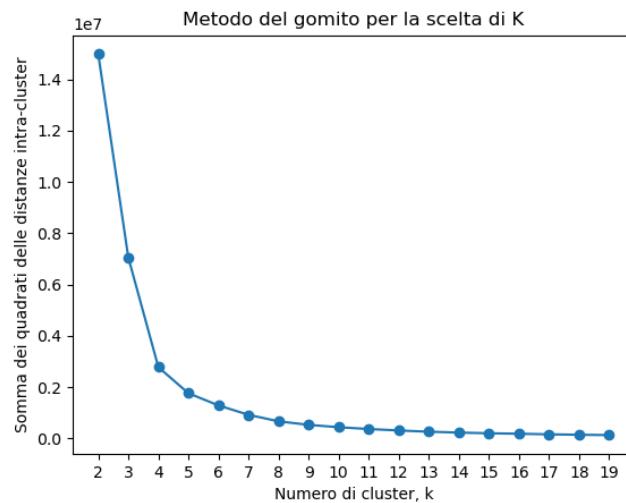


Figura 3.14. Elbow method

Test effettuati

Per avere una stima di quanto l'algoritmo di segmentazione fosse corretto abbiamo effettuato dei test.

Abbiamo eseguito l'algoritmo su ogni MRI presente nel dataset.

Ad ogni iterazione abbiamo utilizzato una funzione di OpenCV per confrontare due immagini.

La prima immagine è totalmente nera con dentro il tumore vero(preso dal record in questione già salvato nel dataset).

La seconda immagine anche è nera con dentro il tumore trovato dal programma. In Fig.3.15 è possibile vedere il confronto che viene effettuato tra il tumore presente nel dataset e il tumore trovato dal programma.

La funzione openCV ritorna un numero che va da 0 ad 1 e indica l'indice di somiglianza tra le due immagini, se questo valore è maggiore o uguale a 0.97 l'esecuzione è corretta, errata altrimenti.

Nelle MRI in cui non è presente il tumore l'immagine sarà una foto totalmente

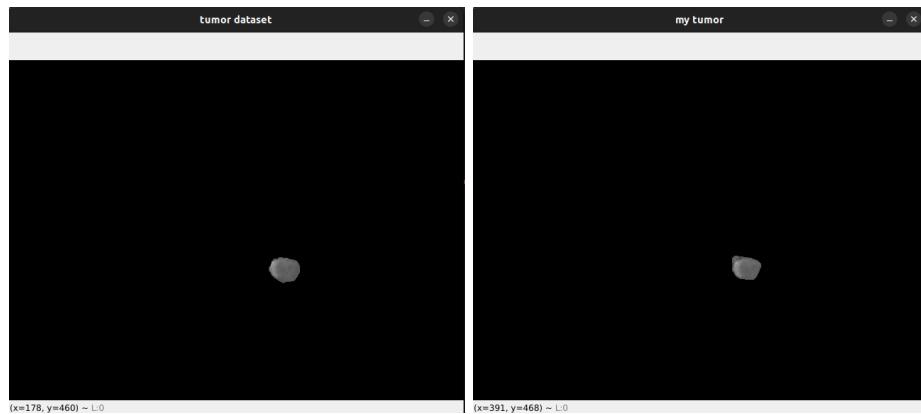


Figura 3.15. Confronto tra tumore presente nel dataset e tumore trovato dal programma

nera, se anche il programma non trova il tumore verrà eseguito un confronto fra due immagini uguali che porterà ad un esito positivo.

Risultati ottenuti

Il grafico generato dall'esecuzione del test di affidabilità possiamo vederlo in Fig.3.16.

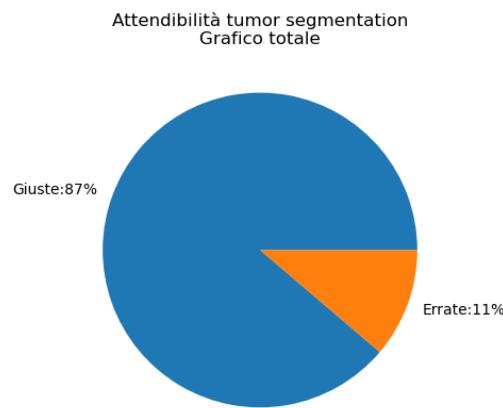


Figura 3.16. Grafico tumor segmentation

L'11% di tumori non trovati correttamente può indicare che il programma non ha rilevato il tumore oppure che il programma non ha trovato tutto il tumore.

Nel dataset scaricato sono presenti alcune MRI con tumori estremamente piccoli che anche ad occhio nudo è difficile rilevarli, in questi casi il programma non è stato in grado di rilevarli e rientrano in quell'11%.

Un esempio è presente in Fig.3.17



Figura 3.17. Tumore non trovato dal programma

3.2 Classificazione del tumore

Secondo task del progetto: **Classificare il tumore**.

Questa fase prende in input il contorno che ha trovato la prima fase e studia le features più importanti che può trovare all'interno del contorno.

La classificazione del tumore viene eseguita grazie ad un modello di ML che viene precedentemente addestrato su un dataset di features calcolate.

Se il programma non trova tumori questa seconda fase non viene eseguita.

3.2.1 Acquisizione delle features più importanti

La difficoltà di questa fase è stata quella di cercare quali fossero le caratteristiche che distinguono un tumore dall'altro e che siano simili tra tumori dello stesso tipo (Glioma,Meningioma,Pituitary).

Una prima caratteristica visibile anche ad occhio nudo è il colore del tumore, il Glioma tende ad essere più scuro rispetto al Meningioma, questo purtroppo non è sufficiente per classificare e distinguere tra questi due tumori in quanto non è così in tutti i casi. Una volta trovate più caratteristiche possibili, abbiamo dovuto

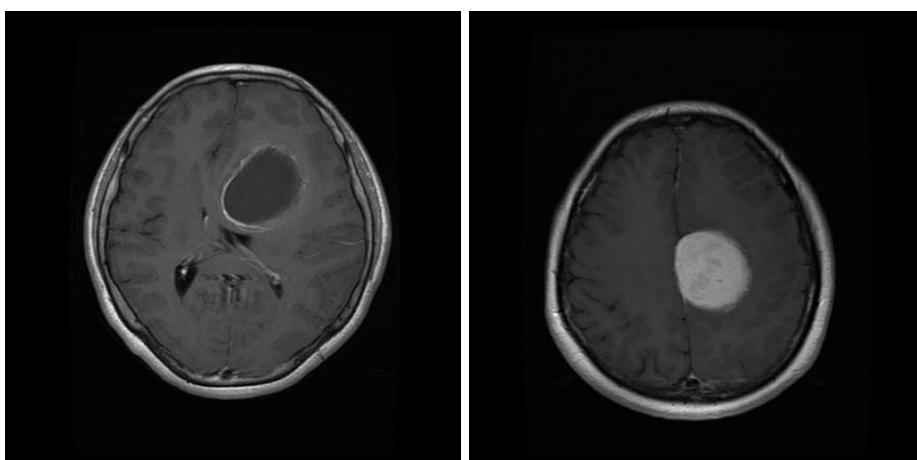


Figura 3.18. Glioma e Meningioma

studiare come andare a calcolare e trovare queste features da un tumore tramite codice Python.

Le features che abbiamo trovato sono:

- **Area**

L'area rappresenta la dimensione totale del tumore, cioè l'estensione spaziale occupata dal tumore nel cervello. L'area può variare tra i diversi tipi di tumori

- **Perimetro**

Il perimetro indica la lunghezza del contorno esterno del tumore. Il perimetro può essere influenzato dalla forma e dalla struttura del tumore e può essere diverso tra i tipi di tumori.

- **Solidità**

La solidità è una misura della compatezza del tumore. Rappresenta la propor-

zione di tessuto solido rispetto al tessuto non solido all'interno del tumore. La solidità può essere un indicatore della densità del tumore e può variare tra i diversi tipi di tumori.

- **Aspect ratio**

L'aspect ratio è il rapporto tra la lunghezza e la larghezza del tumore. Può fornire informazioni sulla forma e l'allungamento del tumore.

- **Eccentricità**

L'eccentricità è una misura della deviazione del tumore dalla forma circolare. Un valore di eccentricità più vicino a 0 indica una forma più circolare, mentre un valore più vicino a 1 indica una forma più ellittica o allungata.

- **7 Hu moment**

Gli Hu moment sono un insieme di sette momenti di Hu, che sono descrittori numerici utilizzati per rappresentare la forma del tumore. Questi momenti possono catturare caratteristiche come l'orientazione, l'allungamento e la simmetria del tumore.

- **Colore medio**

Il colore medio rappresenta il valore medio del colore all'interno del tumore. Questa caratteristica può essere ottenuta calcolando la media dei valori di colore dei pixel all'interno del tumore. Le differenze nel colore medio possono essere indicative di diversi tipi di tumori.

- **Gradiente medio**

Il gradiente medio rappresenta la variazione media dell'intensità dei pixel all'interno del tumore. Il gradiente può fornire informazioni sulla transizione delle intensità dei pixel e può essere utilizzato per rilevare i confini del tumore.

- **Spectral ratio**

Lo spectral ratio si riferisce al rapporto tra le intensità di due bande spettrali specifiche all'interno del tumore. Questa caratteristica può essere utile per distinguere i diversi tipi di tessuti tumorali sulla base delle loro proprietà spettrali.

- **Texture**

Le caratteristiche di texture rappresentano le proprietà spaziali dei pattern di intensità all'interno del tumore. Esistono diverse metriche di texture, come l'energia, la contrasto, l'omogeneità, che possono essere calcolate per valutare le proprietà texture del tumore.

- **Tipo di tumore(etichetta)**

Glioma,Meningioma,Pituitary

Un esempio di una riga del file csv generata è la seguente.

Area	348.5
Perimetro	72.04163
Solidità	0.96006
Aspect Ratio	1.04762
Eccentricità	0.92125
Hu Moment1	0.16137
Hu Moment2	0.00018
Hu Moment3	3.91e-05
Hu Moment4	4.18e-07
Hu Moment5	1.69e-12
Hu Moment6	5.39e-09
Hu Moment7	-4.89e-14
Int.Med	0.12233
Int.MedGrad.	0.22858
Rap.Spet	0.17480
Tex1	0.99733
Tex2	1.36021
Tex3	0.94567
Tex4	12.51522
Tex5	0.99884
Tex6	0.24534
Tex7	48.70066
Tex8	0.02415
Tex9	0.02676
Tex10	0.00678
Tex11	0.02016
Tex12	-0.73180
Tex13	0.17437
Tipo	meningioma

Tabella 3.1. Descrizione delle caratteristiche

3.2.2 Modelli di Machine Learning: principali differenze

Illustriamo alcuni tra i principali modelli di Machine Learning dandone una definizione.

- **SVM: Support Vector Machine**

SVM è un algoritmo di apprendimento supervisionato utilizzato per la classificazione e la regressione. Cerca di trovare un iperpiano ottimale che separi i punti dei dati delle diverse classi nello spazio delle caratteristiche. SVM punta a massimizzare la distanza tra gli esempi di addestramento più vicini (support vector) e il confine decisionale.

Fig. 3.19

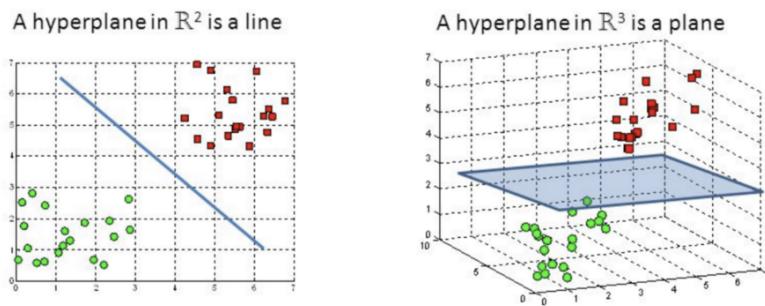


Figura 3.19. SVM, fonte: [11]

- **NB: Naive Bayes**

NB è un algoritmo di apprendimento supervisionato utilizzato per la classificazione. Si basa sull'applicazione del teorema di Bayes con l'assunzione di indipendenza condizionale tra le features. NB è veloce e richiede meno dati per l'addestramento rispetto ad altri algoritmi. È particolarmente utile quando le dimensioni dello spazio delle caratteristiche sono elevate.

Fig.3.20

$$P(A|E) = \frac{P(E|A) \cdot P(A)}{P(E)}$$

probabilità condizionata dell'evento E noto A probabilità a priori dell'evento A

↓ ↓

↑

probabilità condizionata dell'evento A noto E probabilità a priori dell'evento E

Figura 3.20. NB, fonte: [9]

- **RF: Random Forest**

RF è un algoritmo di apprendimento supervisionato utilizzato principalmente per la classificazione e la regressione. Si basa sulla costruzione di un insieme di alberi decisionali. Ogni albero viene addestrato su un sottoinsieme casuale dei dati di addestramento e utilizza una selezione casuale delle caratteristiche. L'output finale è ottenuto attraverso la votazione o la media degli output dei singoli alberi.

Nel caso della classificazione con RF, ogni albero nell'insieme emette una previsione di classe per un dato campione. L'output finale viene ottenuto selezionando la classe più frequente tra le previsioni di tutti gli alberi (votazione a maggioranza). Ad esempio, se ci sono 10 alberi nel Random Forest e 7 alberi predicono la classe A, mentre 3 alberi predicono la classe B, allora l'output finale sarà la classe A.

Nel caso della regressione con RF, ogni albero fornisce una stima numerica per un dato campione. L'output finale è ottenuto calcolando la media delle stime fornite da tutti gli alberi. Ad esempio, se ci sono 10 alberi nel Random Forest e ognuno fornisce una stima di un valore numerico, l'output finale sarà la media di tutte queste stime.

La votazione o la media degli output dei singoli alberi nel Random Forest permette di ottenere una previsione o una stima più robusta e stabile, riducendo l'effetto del rumore o delle previsioni errate generate da singoli alberi. Questo approccio rende il Random Forest un algoritmo potente e ampiamente utilizzato in diverse applicazioni di machine learning.

Fig.3.21

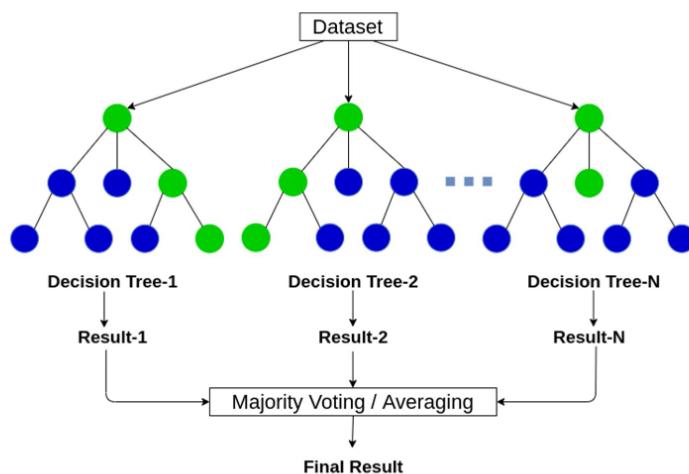


Figura 3.21. RF, fonte: [10]

- **KNN: K-Nearest Neighbors**

KNN è un algoritmo di apprendimento supervisionato che utilizza la prossimità per effettuare classificazioni. Sebbene possa essere utilizzato per problemi di regressione o classificazione, viene generalmente utilizzato come algoritmo di classificazione, basandosi sul concetto di vicinanza: punti simili possono essere trovati l'uno vicino all'altro.

Fig.3.22

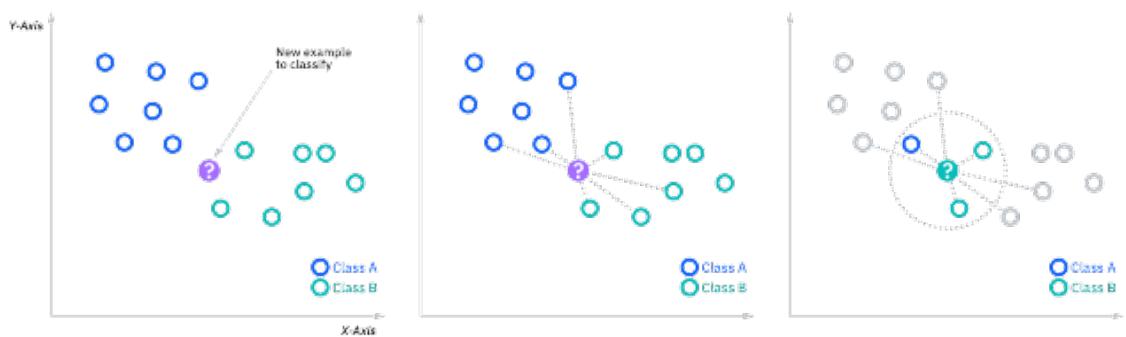


Figura 3.22. KNN, fonte: [5]

3.2.3 Creazione e allenamento del modello ML Random Forest

In questa fase abbiamo analizzato ogni tumore presente nel dataset scaricato ed abbiamo eseguito la funzione del calcolo delle features per ogni tumore, andando a scrivere all'interno di un file csv tutte le features del tumore *i-esimo* calcolate con la funzione sopra descritta.

Ogni riga del file quindi rappresenta un tumore e viene identificata con l'insieme delle features a cui si aggiunge l'etichetta del tipo di tumore (presente in ogni tumore del dataset scaricato).

I dati presenti nel file sono stati inseriti nei seguenti vettori:

- **X_train**

Vettore contenente le features di ogni record che verrà utilizzato durante l'allenamento, non sono presenti l'etichette.

- **X_testing**

Vettore contenente le features di ogni record che verrà utilizzato durante il testing, non sono presenti l'etichette.

- **y_train**

Vettore contenente gli stessi record presenti nel vettore X_train ma con l'etichetta e basta, non sono presenti tutte le features (viene mantenuto l'ordine degli elementi presenti in X_train e y_train).

- **y_testing**

Vettore contenente gli stessi record presenti nel vettore X_testing ma con l'etichetta e basta, non sono presenti tutte le features.

secondo la seguente percentuale: 80% dei dati sono stati inseriti nel vettore *train* e il restante 20% nel vettore *testing*.

Vediamo queste tabelle riguardanti un dataset di classificazione di fiori, per comprendere meglio quanto detto.

La tabella 3.23a contiene l'intero dataset, ogni entry contiene l'identificativo univoco di quel record(*id*), tutte le features + l'etichetta(*species*).

Le tabelle 3.23b, 3.23c, 3.23d, 3.23e raffigurano quanto descritto in precedenza.

Successivamente abbiamo creato il modello ML che è stato addestrato con i vettori X_train e y_train.

Abbiamo utilizzato il vettore X_test e y_test per testare l'affidabilità del modello. Nel codice seguente è possibile vedere come sono stati creati i 4 vettori a partire dai file csv generati precedentemente.

La funzione *trainRF* crea un modello ML di tipo Random Forest che allena il modello con i vettori di training e salva i parametri in un file (per evitare di allenare il modello ogni volta che deve essere eseguito).

La funzione *Testing* carica i parametri dal file generato dalla funzione di Training ed effettua la *predizione* sul vettore di testing e infine calcola l'accuratezza.

Il perchè è stato scelto l'algoritmo Random Forest è spiegato nel prossimo paragrafo.

(a) Dataset Originale

id	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	5.4	3.9	1.3	0.4	setosa
3	7.9	3.8	6.4	2.0	virginica

(b) X_Train

id	sepal_length	sepal_width	petal_length	petal_width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2

(c) y_Train

id	species
0	setosa
1	setosa

(d) X_Test

id	sepal_length	sepal_width	petal_length	petal_width
2	5.4	3.9	1.3	0.4
3	7.9	3.8	6.4	2.0

(e) y_Test

id	species
2	setosa
3	virginica

Figura 3.23. Tabelle vettori

3.2.4 Altri modelli di Machine Learning, differenze di affidabilità

La scelta del modello Random Forest è dovuta al fatto che è stato l'algoritmo che ha generato più accuratezza.

Non esiste un modello che sia migliore rispetto ad un altro in un qualsiasi contesto, ma a seconda del problema che bisogna risolvere un algoritmo può risultare più efficiente rispetto ad un altro.

Illustriamo i grafici in Fig.3.24 e 3.25 che abbiamo generato dopo avere effettuato la predizione con i seguenti algoritmi

- **Random Forest**
- **Support Vector Machine**
- **Naive Bayes**
- **K-Nearest Neighbors**

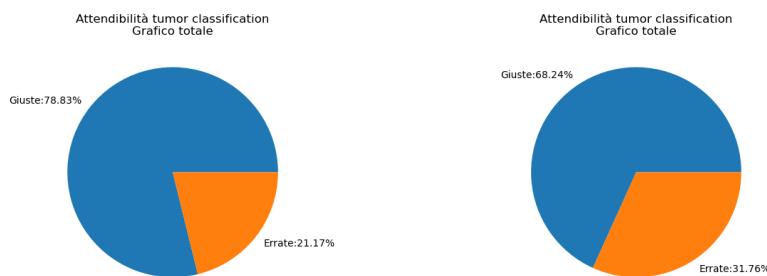


Figura 3.24. da sx a dx: RF e KNN

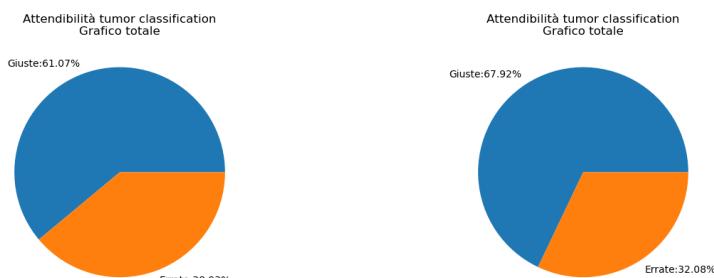


Figura 3.25. da sx a dx: NB e SVM

Come possiamo vedere, l'algoritmo RF è quello che ha generato più accuratezza. L'algoritmo NB ha dato un accuratezza molto bassa poichè, per sua natura NB si basa sull'assunzione di indipendenza condizionale tra le caratteristiche. Di conseguenza l'assunzione di indipendenza potrebbe non essere adeguata per le features calcolate in questo specifico compito in quanto alcune dipendono da altre.

3.3 Gui

3.3.1 Scopo della Gui

L'implementazione di un interfaccia grafica ha come scopo quello di rendere l'applicazione *user-friendly*.

L'applicazione deve poter essere usata da chiunque (non solo da informatici) di conseguenza l'utente che vuole usarla non deve interagire con linea di comando/IDE o altro che possa indurlo a compiere azioni errate.

3.3.2 Descrizione

La GUI è stata scritta tramite l'uso della libreria *flet* di Python.

E' presente un pulsante *Upload File* che permette di scegliere il file presente nel computer che vogliamo inserire. Gestisce tutti i casi in cui l'utente prova ad inserire formati non accettati, infatti l'applicazione accetta solo formati immagine come jpg,jpeg,png; ma anche formato matlab (.mat)

Una volta inserita l'immagine, questa viene mostrata sulla sinistra sotto la sezione *Before*, ora l'utente può premere il pulsante *Start* e il programma inizia.

Se l'utente preme start prima ancora di effettuare l'upload del file, il programma gestisce l'errore invitando l'utente a caricare un file.

E' possibile salvare l'immagine segmentata premendo il pulsante *Save Output Image* e scegliere la directory dove salvarla.

Infine il pulsante *Clean* cancella tutte le immagini di output che il programma salva sempre nella directory *tmp*.

Nella Fig.3.26 si può vedere un esempio, da notare come viene mostrata anche l'area del tumore e la probabilità che il contorno trovato sia realmente un tumore, per ottenere questo valore in percentuale il programma analizza la media ponderata del contorno (se è molto elevata allora con un'elevata probabilità il contorno sarà un tumore, e così via a scendere finché non raggiunge un valore minimo oltre il quale non viene più considerato più tumore).

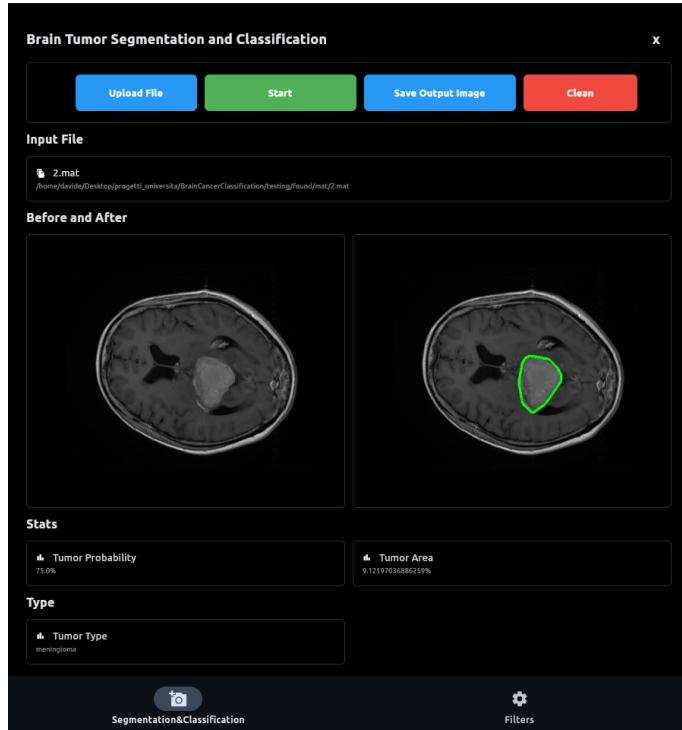


Figura 3.26. Esempio GUI e classification

L'output generato dal programma quando non trova il tumore è possibile vederlo nella Fig.3.27

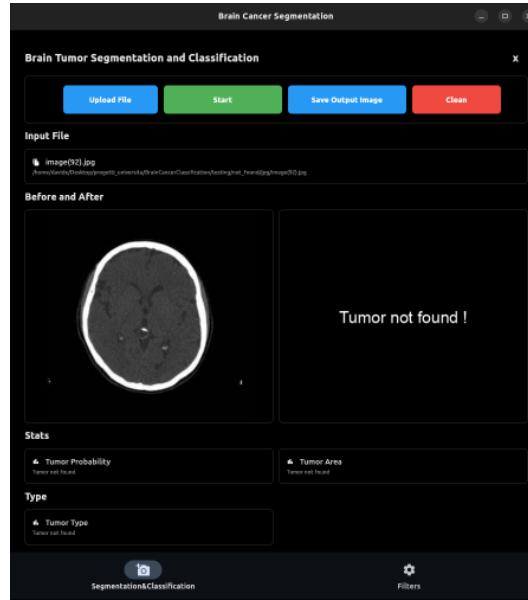


Figura 3.27. Tumore non rilevato

Filtri

L'utente può scegliere anche di selezionare il button in basso a destra *filters*.

Qui l'utente può caricare un immagine segmentata (restituita dal programma) e applicare una serie di filtri che vengono eseguiti all'interno o all'esterno del contorno segmentato. Può decidere anche qui di salvare l'immagine o svuotare la directory tmp.

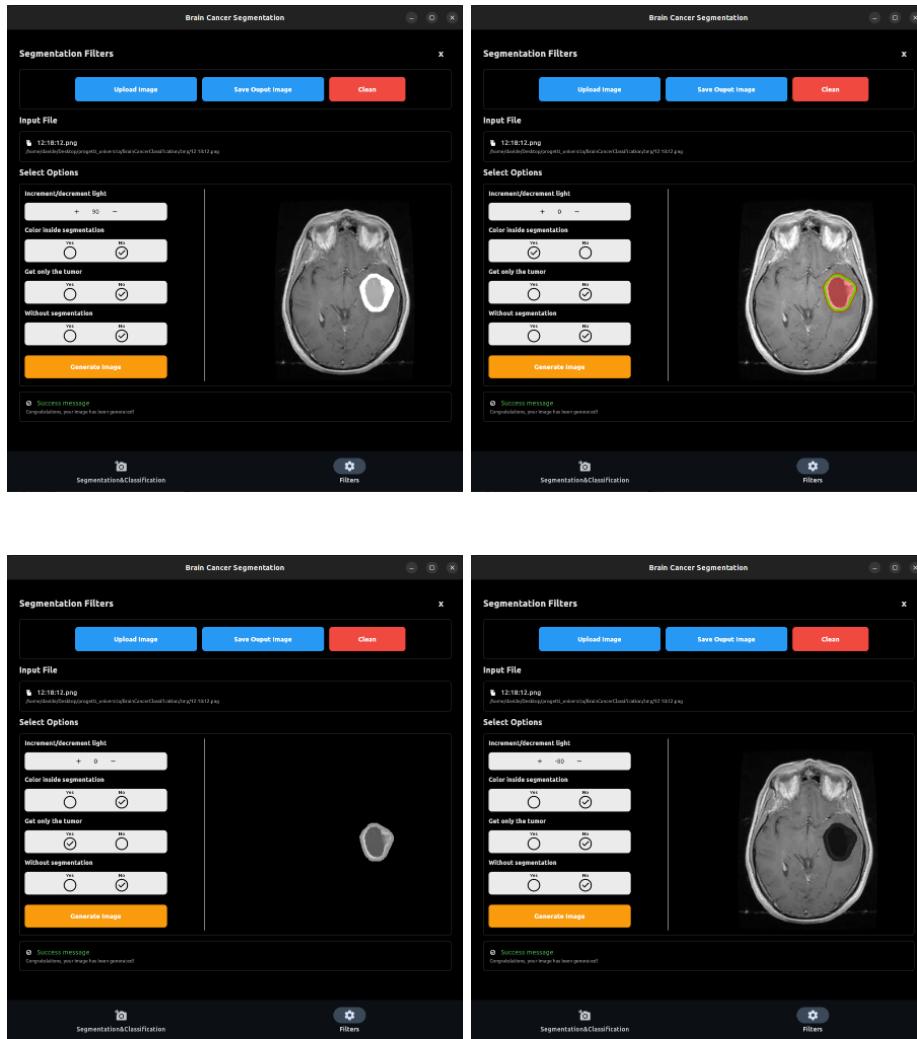


Figura 3.28. Filtri

Capitolo 4

ESPERIMENTI E RISULTATI OTTENUTI

In questa sezione vengono illustrati i risultati ottenuti durante l'intero svolgimento del progetto, confrontandoli con lo stato dell' arte attuale mostrando le differenze e il miglioramento proposto.

4.0.1 Skull Stripping

Illustriamo alcuni esempi di risultati ottenuti in questo passaggio.

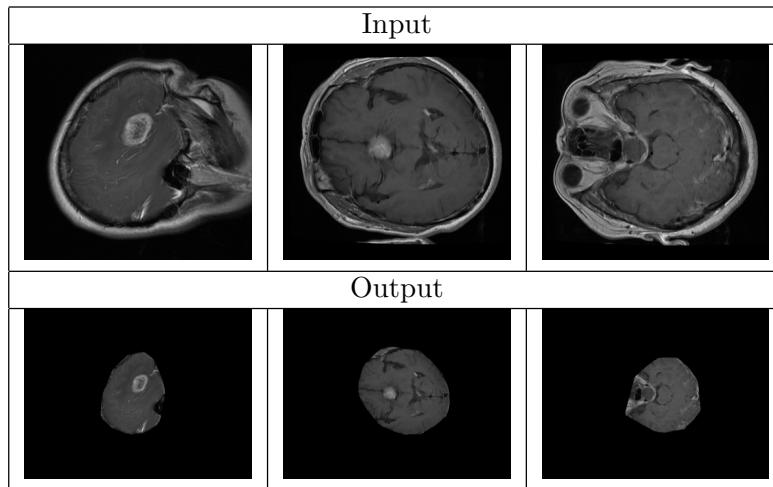


Tabella 4.1. Esempi skull stripping

4.0.2 Segmentazione

Illustriamo un esempio di segmentazione effettuata con algoritmi noti utilizzati allo stato dell'arte e confrontiamo il risultato ottenuto con l'output del progetto. Nella

Confronto metodi		
Input	CNN method	Our method
		

Tabella 4.2. Tabella confronto metodi

tabella 4.2 possiamo vedere il confronto tra l'analisi della MRI attraverso algoritmi di reti neurali utilizzati allo stato dell'arte, e l'analisi della MRI attraverso il nostro progetto.

L'immagine centrale rappresenta l'output allo stato dell'arte, questo algoritmo genera un'immagine totalmente nera lasciando in bianco solo i contorni *anomali*, ovvero i contorni che potrebbero rappresentare un tumore.

L'immagine a destra rappresenta l'output del nostro programma.

E' possibile notare come nell'immagine centrale non è stato trovato il contorno del tumore ma altri contorni che non rappresentano dei tumori.

L'immagine a sinistra rappresenta l'input, in questo caso un processo di skull stripping iniziale che vada ad eliminare tutto ciò che non riguarda il tessuto cerebrale può essere di grande aiuto nella segmentazione, se questo passaggio non viene effettuato o viene svolto in modo non corretto, il processo può portare ad una segmentazione sbagliata.

Vengono illustrate in tabella 4.3 ulteriori esempi di segmentazioni svolte dal nostro progetto.

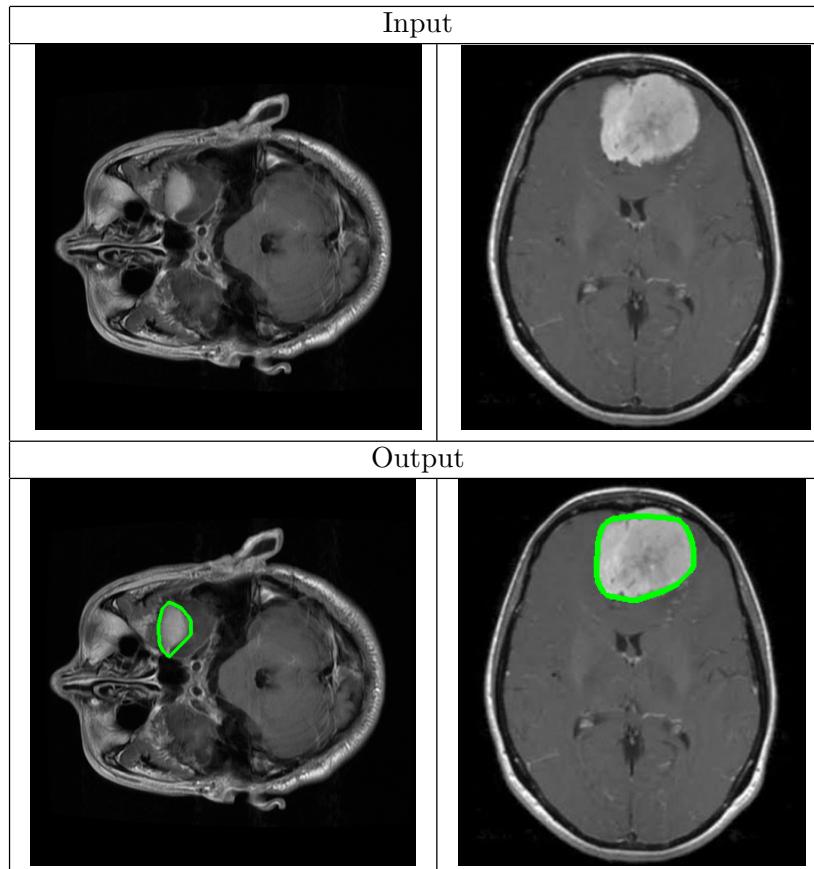


Tabella 4.3. Esempi skull stripping

4.0.3 Classificazione

Il nostro progetto ha riscontrato ulteriori passi in avanti anche nel processo di classificazione, specialmente nelle MRI molto complesse.

E' possibile vedere un esempio in Fig. 4.1 che rappresenta l'output di un algoritmo



Figura 4.1. CNN classification

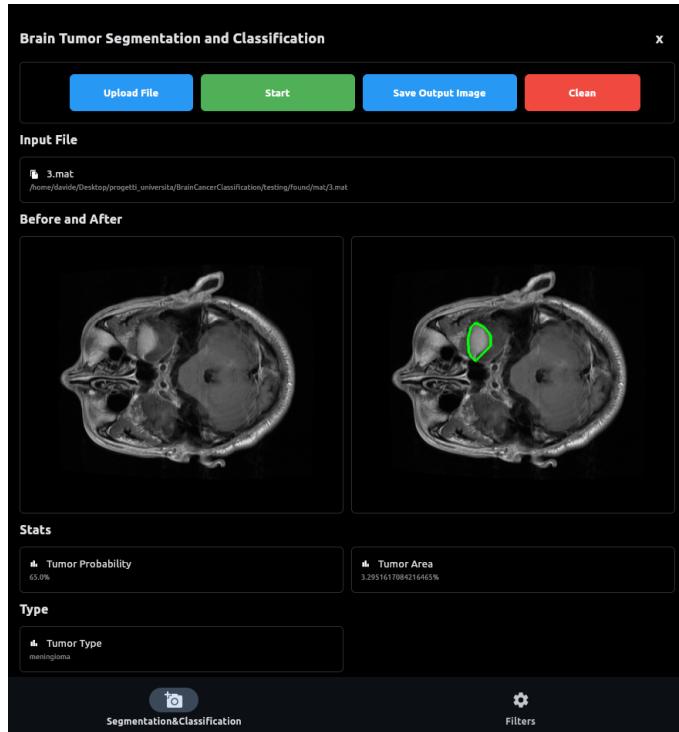


Figura 4.2. Our Classification

di classificazione che utilizza le CNN, mentre in Fig.4.2 è presente l'output mostrato dal nostro programma.

Le CNN classificano il tumore in esame come un *Glioma* mentre il nostro programma lo classifica come un *Meningioma*.

Capitolo 5

CONCLUSIONE ED ASPETTI FUTURI

Il programma che abbiamo sviluppato per la segmentazione e classificazione dei tumori presenta una struttura modulare e scalabile, in cui ogni fase dipende dai risultati delle fasi precedenti. Questa architettura modulare offre numerosi vantaggi in termini di flessibilità e miglioramenti futuri.

Nel processo di segmentazione, il programma inizia eseguendo l'operazione di skull stripping per rimuovere le informazioni non desiderate dall'immagine. Successivamente, cerca di trovare il colore medio del tumore, e infine si procede alla segmentazione del tumore stesso. Questa metodologia progressiva consente di ottenere risultati più accurati e affidabili, poiché ogni fase si basa sui risultati delle fasi precedenti.

Nella fase di classificazione, il programma utilizza il contorno estratto dalla segmentazione come input per l'estrazione delle caratteristiche. Queste caratteristiche, vengono quindi utilizzate per addestrare un modello di Random Forest (RF) per la classificazione dei tumori. Il modello addestrato può quindi effettuare previsioni sul tipo di tumore sulla base delle caratteristiche estratte dal contorno.

Un aspetto chiave è la scalabilità. Poiché ogni fase è indipendente, il miglioramento di una fase può essere implementato senza dover modificare l'intero programma. Questa caratteristica rende il programma flessibile e adattabile a futuri progressi tecnologici o nuove metodologie di segmentazione o classificazione dei tumori.

Ad esempio, se vengono introdotte nuove tecnologie o metodologie per la segmentazione o la classificazione dei tumori, è possibile integrarle facilmente nelle fasi corrispondenti del programma. Ciò potrebbe portare a un miglioramento complessivo dell'affidabilità del programma e alla possibilità di ottenere risultati ancora più precisi e accurati nel rilevamento dei tipi di tumore.

5.1 Verso una ricostruzione 3d

Un primo miglioramento nello studio e nella classificazione del tumore può essere fatto andando a ricostruire il cervello (e il tumore) in 3d.

Una ricostruzione in 3d del tumore permetterebbe sicuramente uno studio più ampio della forma, dimensione ed estensione del tumore per una classificazione più accurata. Cos'è l' **Echo Planar Imaging**

L'imaging echo planare è una tecnica di imaging di risonanza magnetica "ultra veloce" in grado di produrre immagini tomografiche a frequenza video (da 15 a 30 immagini al secondo). La sequenza di scansione più comunemente usata negli studi di fMRI dell'encefalo è detta GE-EPI (Gradient-Echo Echo Planar Imaging). La sequenza EPI è un procedimento di acquisizione delle immagini molto veloce e richiede macchine sofisticate in grado di applicare i tre gradienti di impulso

descritti sopra in successione molto rapida e quindi consente una scansione molto veloce con TE molto basso: l'intero volume, infatti, è acquisito in circa 50-100 ms. Il prezzo da pagare per questa rapidità di scansione e la presenza di artefatti, legati al movimento, alla suscettività..., che necessitano di essere aboliti durante la post-produzione delle immagini. Nell'MRI, lo spazio-k equivale allo spazio definito dalle direzioni della codifica di frequenza e di fase. Le sequenze convenzionali registrano una linea dello spazio-k ad ogni passo della codifica di fase. Poiché si ha un passo della codifica di fase ogni TR (tempo di ripetizione della sequenza) secondi, il tempo richiesto per la produzione dell'immagine è dato dal prodotto di TR per il numero di passi della codifica di fase. L'imaging echo planare registra tutte le linee dello spazio-k in un singolo TR.

Fonte: [3]



Figura 5.1. fonte: [3]

Dopo aver scaricato un set di 68 immagini in cui non è presente il tumore effettuate con l'EPI ad un paziente (fonte: [2]) dall'alto verso il basso della nuca, abbiamo inserito le seguenti immagini in un plot 3d per ricostruire il volto del paziente.

Questo è solo un primo passo verso la vera e propria ricostruzione 3d del cervello che andrebbe eseguita tramite il processo di *triangolazione*.

Nella figura 5.2 illustro i risultati ottenuti fin'qui.

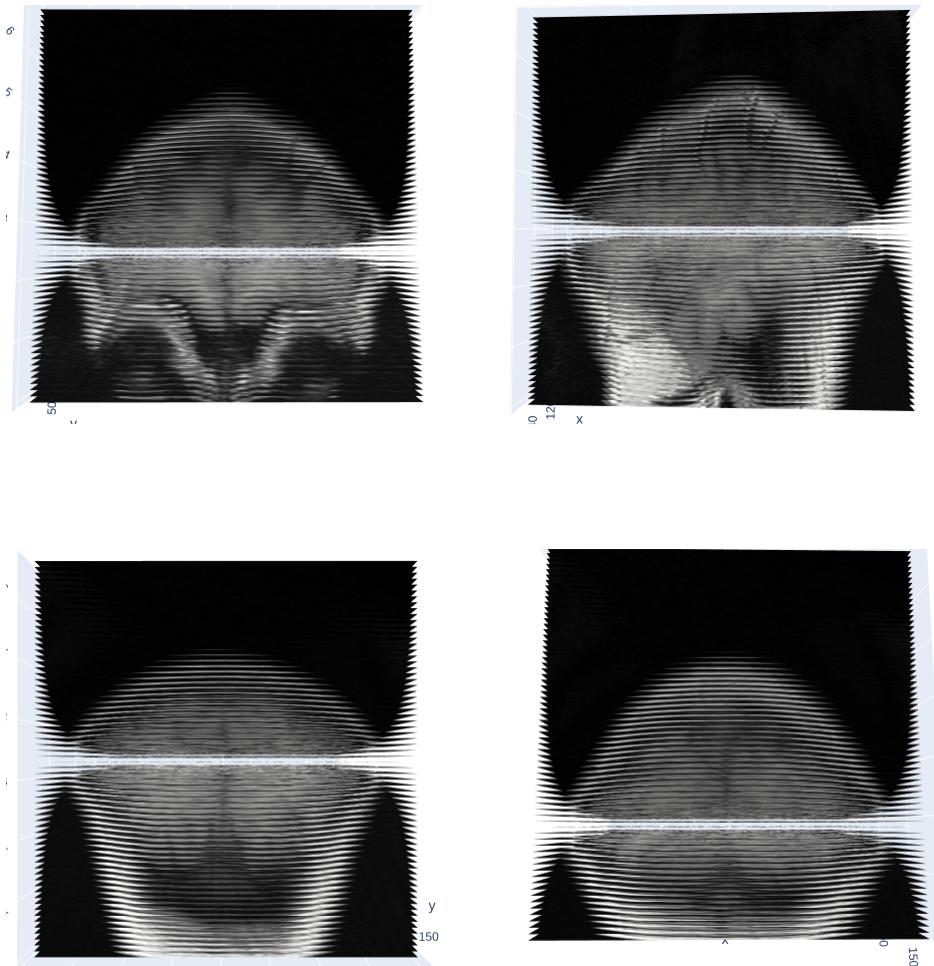


Figura 5.2. Da sinistra a destra vista a 360 gradi della nuca.

Ringraziamenti

Vorrei ringraziare il Prof. **Daniele Pannone** (Co-Responsabile) per avermi seguito e per avermi dato la possibilità di svolgere questo lavoro.
E inoltre ringrazio la mia famiglia, che mi ha sostenuto lungo tutto il mio percorso di studio.

Bibliografia

- [1] Cnn classification. Available from: https://www.researchgate.net/figure/The-proposed-BRAIN-RENet-deep-CNN-for-brain-tumor-classification-The-proposed-BRAIN-RENet_fig3_357874994.
- [2] Dataset epi. Available from: https://s3.amazonaws.com/assets.datacamp.com/blog_assets/attention-mri.tif.
- [3] Descrizione epi. Available from: http://arpg-serv.ing2.uniroma1.it/arpg-site/images/ARPG_MEDIA/Tesi/tesi_finale_Andellini.pdf.
- [4] Gerarchia ai. Available from: <https://www.frontiersin.org/articles/10.3389/fpsyg.2023.992541/full>.
- [5] Knn. Available from: <https://www.ibm.com/topics/knn>.
- [6] Mri dataset1. Available from: <https://www.kaggle.com/datasets/sartajbhuvaji/brain-tumor-classification-mri>.
- [7] Mri dataset2. Available from: https://figshare.com/articles/dataset/brain_tumor_dataset/1512427.
- [8] Mri views. Available from: <https://www.ipfradiologyrounds.com/hrct-primer/image-reconstruction/>.
- [9] Nb. Available from: <https://www.eage.it/machine-learning/naive-bayes>.
- [10] Rf. Available from: <https://anasbrital98.github.io/blog/2021/Random-Forest/>.
- [11] Svm. Available from: <https://www.marktechpost.com/2021/03/25/introduction-to-support-vector-machines-svms/>.
- [12] CAPPELLI, R. Fondamenti di elaborazione di immagini estrazione dei bordi e segmentazione. Available from: <http://bias.csr.unibo.it/fei/Dispense/3%20-%20FEI%20-%20Estrazione%20dei%20Bordi%20e%20Segmentazione.pdf>.
- [13] DI NUZZO, M. *Deep Learning con Keras e Tensorflow: Progetti di Intelligenza Artificiale* (2022).