Alma Mater Studiorum — Università di Bologna
Campus di Cesena

Dipartimento di Informatica — Scienza e Ingegneria
Corso di Laurea Magistrale in Ingegneria e Scienze Informatiche

Progetto e implementazione di un software per la segmentazione assistita di immagini mediche e predizione di biomarkers

Presentata da:
Pagnini Lorenzo
Matricola 942265

Relatore:
Prof. Alessandro Bevilacqua

Correlatori:
Ing. Alessandro Gherardi
Ing. Margherita Mottola

III Sessione di Laurea
Anno Accademico 2021–2022

Alla mia famiglia e a chi ha sempre creduto in me.
Oggi ho realizzato il mio sogno.

Indice

Parole chiave

Imaging Medico

CAD

Segmentazione Assistita

Machine Learning

Sommario

La segmentazione prevede la partizione di un'immagine in aree strutturalmente o semanticamente coerenti. Nell'*imaging* medico, è utilizzata per identificare, contornandole, Regioni di Interesse (ROI) clinico, quali lesioni tumorali, oggetto di approfondimento tramite analisi semiautomatiche e automatiche, o bersaglio di trattamenti localizzati. La segmentazione di lesioni tumorali, assistita o automatica, consiste nell'individuazione di pixel o voxel, in immagini o volumi, appartenenti al tumore. La tecnica assistita prevede che il medico disegni la ROI, mentre quella automatica è svolta da software addestrati, tra cui i sistemi *Computer Aided Detection* (CAD). Mediante tecniche di visione artificiale, dalle ROI si estraggono caratteristiche numeriche, *feature*, con valore diagnostico, predittivo, o prognostico.

L'obiettivo di questa Tesi è progettare e sviluppare un software di segmentazione assistita che permetta al medico di disegnare in modo semplice ed efficace una o più ROI in maniera organizzata e strutturata per futura elaborazione ed analisi, nonché visualizzazione. Partendo da Aliza Medical Software, applicativo *open-source*, visualizzatore di esami radiologici in formato DICOM, è stata estesa l'interfaccia grafica per gestire disegno, organizzazione e memorizzazione automatica delle ROI. Inoltre, è stata implementata una procedura automatica di elaborazione ed analisi di ROI disegnate su lesioni tumorali prostatiche, per predire, di ognuna, la probabilità di cancro clinicamente non-significativo e significativo (con prognosi peggiore). Per tale scopo, è stato addestrato un classificatore lineare basato su *Support Vector Machine*, su una popolazione di 89 pazienti con 117 lesioni (56 clinicamente significative), ottenendo, in test, accuratezza = 77%, sensibilità = 86% e specificità = 69%. Il sistema sviluppato assiste il radiologo, fornendo una *seconda opinione*, non vincolante, adiuvante nella definizione del quadro clinico e della prognosi, nonché delle scelte terapeutiche.

1 Introduzione

1.1 Contesto e Problema

Grazie all'evoluzione digitale, qualsiasi dato (inteso come un numero, un'immagine e un video) può essere manipolato ed interpretato da un software per svolgere diversi compiti. Tutto ciò è possibile anche grazie alla visione artificiale mediante l'utilizzo di tecniche complesse che simulano la vista umana. Tra i tanti problemi che questi sistemi riescono a rispondere, vi sono anche a quelli associati al riconoscimento, localizzazione e tracciamento di oggetti all'interno di un'immagine [1]. Gli ambiti di riferimento sono svariati: possono riguardare l'intrattenimento (ad esempio i foto-ritocchi), il settore industriale (ad esempio il riconoscimento dei difetti sul prodotto finito), la video-sorveglianza e l'ambito medico. É proprio quest'ultimo caso, il contesto trattato dall'elaborato.

Nel panorama *healthcare*, la digitalizzazione ha permesso lo sviluppo di nuove strumentazioni in grado di produrre, in formato digitale, esami medici in due, tre e quattro dimensioni. A partire da questi esami, è possibile esplorare, esaminare e monitorare una specifica area, non visibile dall'esterno, attraverso processi di *imaging* medico. Una tecnica che riveste particolare importanza è la segmentazione in quanto permette di estrarre da un'immagine, le ROI separate rispetto alla parte di *background*. Tali regioni dovranno essere successivamente analizzate dal medico al fine di determinare se la Regione evidenziata è significativa dal punto di vista clinico. Questo processo può essere anche affiancato da un software in grado di estrarre dalle regioni individuate, un insieme di *features* per eseguire una predizione. Questi sistemi vengono chiamati anche *secondo radiologo* e supportano il medico durante la fase diagnostica.

Nel contesto di questo lavoro, nasce quindi l'esigenza di avere a disposizione uno strumento *general-purpose* (*tool* e interfaccia grafica) che consentirà ai medici di effettuare la segmentazione assistita in modo efficiente su immagini che presentano lesioni dei diversi tessuti. Tale strumento dovrà essere in grado di ridurre al minimo l'errore inter-operatore e intra-operatore e di fornire un'accurata predizione sulla tipologia delle lesioni.

L'obiettivo dell'elaborato consiste quindi nel progettare ed implementare l'interfaccia grafica di un software per la segmentazione assistita. In questo modo si permetterà al medico di isolare tali Regioni per consentire al software di eseguire la diagnosi su di esse. La fase di predizione si baserà nel discriminare le lesioni come clinicamente significative o non clinicamente significative. Il caso di studio considerato per questa Tesi, riguarderà le lesioni relative all'organo prostata.

1.2 Organizzazione dei Contenuti

I contenuti del seguente elaborato sono organizzati con l'obiettivo di fornire una descrizione generale e successivamente approfondire il problema trattato.

Il Capitolo 2 fornisce una panoramica generale sulla computer vision in ambito medico: verranno illustrate le principali strumentazioni impiegate, focalizzandosi maggiormente su come questi apparecchi formano l'immagine. Il capitolo prosegue illustrando lo standard Dicom, il problema della segmentazione e classificazione di una lesione e termina mostrando le principali applicazioni presenti.

Il Capitolo 3 descrive la progettazione della soluzione. In particolare l'analisi dei requisiti, la scelta del software da utilizzare come base di partenza per ampliarlo successivamente con le nuove funzionalità, l'architettura del sistema e le principali scelte progettuali effettuate.

Il Capitolo 4 illustra le scelte implementative, evidenziando tecniche e algoritmi utilizzati per sviluppare specifiche funzionalità. Il Capitolo 5 mostra i risultati ottenuti dell'interfaccia grafica sviluppata e del classificatore addestrato. L'elaborato termina, descrivendo i lavori futuri da realizzare per migliorare il progetto ottenuto.

2 Background

La visione artificiale è una branca dell'intelligenza artificiale che permette ad un dispositivo elettronico come ad un computer o ad uno smartphone di analizzare immagini e video. Si tratta di un ambito molto vasto che ricopre molteplici obiettivi tra cui: il riconoscimento e tracciamento di oggetti, la predizione di eventi (si pensi ad esempio alle previsioni meteorologiche), la lettura dei codici a barre, la guida autonoma e la gestione di sistemi interfacciati a robot che compiono movimenti [1]. In particolare gli algoritmi impiegati in questi *task* permettono di:

- riconoscere all'interno di un'immagine o video determinati oggetti (o classi). Per classe si intende un tipo di oggetto, più o meno generico, a seconda del contesto di utilizzo. Ad esempio se è necessario riconoscere il tipo di animale, le classi potrebbero essere cane e gatto. Se invece si intende classificare gli esseri viventi, senza distinguerne il tipo, le classi potrebbero essere solamente uomo, pianta e animale. Si noti come nel primo esempio le classi erano più dettagliate mentre nell'altro erano più generiche. Questo compito è anche detto classificazione.

- localizzare uno o più oggetti. In questo caso, oltre alla classificazione, l'algoritmo deve anche identificare la posizione spaziale all'interno dell'immagine o video. Si tratta di un problema più complesso rispetto al precedente. Per identificare la posizione è necessario determinare un rettangolo (o *bounding box*), in cui l'area interna deve racchiudere pienamente l'oggetto individuato;

- tracciare (o *tracking*) uno o più oggetti. In questo caso oltre alla classificazione e localizzazione, l'algoritmo deve tracciarlo cioè determinare il suo spostamento all'interno di un video (sequenza temporale di immagini);

- segmentare uno o più oggetti, ovvero partizionare una ROI di un'immagine rispetto allo sfondo. Ad esempio se devo segmentare un gatto in un'immagine, in primo luogo dovrò identificare (classificare) l'oggetto, localizzarlo (disegnando la *bounding box*) e infine determinare all'interno del rettangolo i suoi contorni, così da poter estrarre dall'immagine solo l'area che identifica la classe gatto. In un alcuni casi, questo processo coincide con la separazione del *background* con il *foreground* (oggetti in primo piano) di un'immagine.

Un ambito molto promettente è quello medico-sanitario: la figura 2.1 mostra la prospettiva del mercato a livello mondiale dell'utilizzo della *computer vision* in ambito medico, realizzato da Maximize Consulting: azienda di consulenza americana [2]. L'ordinata rappresenta gli investimenti in dollari nel mercato globale (nel 2017 è stato valutato 200 milioni) mentre l'asse delle ascisse rappresenta il progredire degli anni. Si noti come l'area azzurra-celeste (*Medical Imaging and Diagnostics*) aumenti in maniera considerevole (nel 2026 si prevede una stima intorno ai 3450 milioni di dollari).

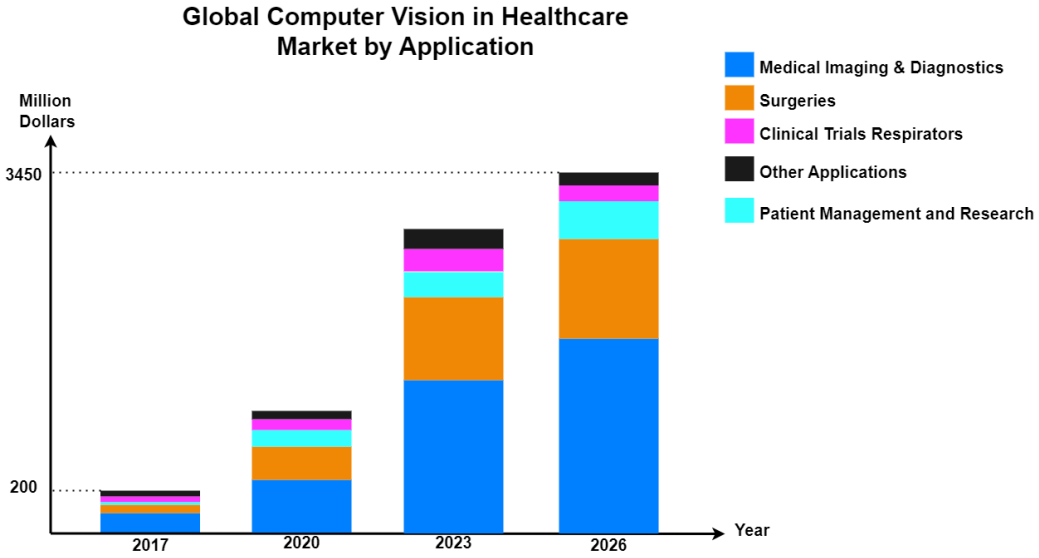


Figura 2.1: Prospettiva del mercato globale della *computer vision* in ambito medico. Studio realizzato da Maximize Consulting, azienda di consulenza americana [2].

Questo successo è attribuibile principalmente all'evoluzione tecnologica che ha permesso una digitalizzazione completa dei dati. Basti pensare che la maggior parte di tutti i dati medici si basano su immagini. Inoltre questi sistemi necessitano di avere alti livelli di accuratezza, superiore alla media rispetto agli altri ambiti, requisito fondamentale se si pensa al contesto in cui vengono applicati. In questo scenario, quello che in passato veniva eseguito dal medico ora può essere svolto, o fornire ausilio in maniera non vincolante, a partire da una macchina. Alcuni obiettivi che questi sistemi possono perseguire si basano sulla diagnosi di malattie. In questi casi non solo *l'occhio digitale* ha una maggiore precisione di visione ma fornisce un risultato in tempi più rapidi rispetto ad un operatore umano. Altri casi riguardano l'utilizzo di tecniche di realtà aumentata per mostrare ologrammi 3D dell'oggetto di studio [3]. Ad esempio nell'ospedale di Singapore, lo staff sanitario utilizza durante gli interventi chirurgici, ologrammi per una maggiore precisione [4].

2.1 Strumentazioni Mediche

Qualsiasi sia l'algoritmo di *computer vision*, per raggiungere il proprio scopo, necessita di ricevere in input dei dati. Quest'ultimi sono rappresentati da immagini e video digitali

prodotti da specifiche strumentazioni che utilizzano un'insieme di tecniche e di processi. In questo paragrafo verrà fornita, in primo luogo, una panoramica che descrive le parti comuni delle apparecchiature presenti nel panorama medico e successivamente ne verranno mostrate le diverse tipologie.

2.1.1 Workstation

A prescindere dalla specifica strumentazione, in una apparecchiatura medica è presente una postazione che permette all'operatore di poter visualizzare e archiviare le immagini. Si tratta di una potente *workstation* diagnostica dotata di schede video dedicate e monitor ad altissima risoluzione (generalmente 4K) in grado di rappresentare le immagini 2D, 3D e 4D in diverse modalità: toni di grigio e a colori. I più moderni sono dotati di tecnologia di retroilluminazione a LED, un sistema di equalizzazione dell'immagine e di sensori integrati che eseguono periodicamente delle verifiche automatiche sulle calibrazioni e performance dei monitor stessi [5–7]. Inoltre sono presenti delle funzionalità che permettono di:

- applicare all'immagine strumenti di base come zoom e rotazione;
- posizionare più immagini su un singolo schermo, riducendo il numero di manipolazioni e la necessità di cambiare *workstation* durante le sessioni di lettura;
- applicare all'immagine le *Look-Up Table* (LUT). Si trattano di informazioni matematiche che vengono utilizzate per modificare tonalità, saturazione e luminosità delle immagini. I singoli valori dei pixel vengono mappati sulla scala desiderata, con l'effetto di enfatizzare un tipo di tessuto piuttosto che un altro;
- applicare all'immagine filtri per migliorare la qualità dell'immagine ed evidenziare una specifica parte dell'organo oggetto di studio. Un esempio può essere l'aumento del rapporto segnale-rumore (*Signal to Noise Ratio* - SNR) di un'immagine;
- creare una fusione multimodale ovvero la generazione di una nuova immagine a partire da due immagini ottenute da metodologie differenti. Ogni tipologia di esame presenta limiti nel descrivere la parte del corpo oggetto di studio e utilizzando un unico tipo di immagine si ha una ristretta capacità di analisi di quest'ultima. Invece grazie alla fusione multimodale, che consiste nell'integrare le informazioni metaboliche e anatomiche in un'unica immagine, a partire da due esami differenti, si può eseguire un'analisi più accurata. Esempi di esami combinati sono la tomografia ad emissioni di positroni (PET) con la tomografia computerizzata (CT) e la PET con la risonanza magnetica (MR);
- esportare le immagini in diversi formati tra cui: jpg, Dicom, Tif e Bmp;

- possibilità da parte dell'operatore di prendere delle annotazioni e di creare i *Key Image Notes* (KIN) ovvero note aggiuntive che migliorano la comunicazione tra le persone coinvolte (equipe medica, paziente). Ad esempio possono essere utilizzati per contrassegnare le immagini come significative per ulteriori esami o interventi futuri.

L'accoppiamento di una *workstation* diagnostica ha permesso di migliorare l'approccio all'indagine da parte dell'operatore rispetto all'utilizzo del diafanoscopio, fornendo notevoli vantaggi per visualizzare, manipolare e archiviare le immagini. L'output di queste strumentazioni è caratterizzato da immagini in toni di grigio avente una *bit depth* a 16 bit. Per fare un confronto, un pixel a 8 bit può rappresentare 256 diverse tonalità di grigio mentre un pixel a 16 bit ne rappresenta 65.536. In quest'ultimo caso avendo più informazioni colore, è possibile visualizzare più variazioni di sfumatura in toni di grigio tra il bianco e il nero. Tutto ciò permette di avere una maggiore sensibilità alla correzione cromatica, requisito fondamentale nel campo dell'*imaging* medico.

2.1.2 Tipologie

In questo paragrafo si entrerà nel dettaglio in merito alle specifiche apparecchiature presenti [8, 9]. Le più comuni sono:

- la radiografia digitale;
- la tomografia computerizzata;
- la risonanza magnetica;
- l'ecografia.

Radiografia Digitale

La radiografia digitale è una tecnica di acquisizione delle immagini che sfrutta i raggi X (onde elettromagnetiche ad alta frequenza). Lo sviluppo tecnologico ha realizzato sensori digitali in grado di sostituire la classica pellicola radiografica in uso fino a qualche anno fa. In questo modo è possibile produrre immagini digitali che possono essere manipolate da strumenti informatici. La radiografia è un esame economico utilizzata per tessuti densi/molli o contenenti aria. L'output prodotto è un'immagine 2D. Esistono due principali tipologie di sistemi per la radiografia digitale: i sistemi a radiologia computerizzata (noti con la sigla CR - *Computed Radiology*) e i sistemi a radiografia digitale (noti con la sigla DR - *Digital Radiography*) [10]. I primi sono costituiti da un pannello che contiene grani di fosforo sensibili alla radiazione. All'emissione di raggi X, l'informazione dell'immagine viene immagazzinata nel pannello e tramite uno scanner laser dedicato, questa viene prima convertita in luce e poi in immagine digitale. Successivamente il laser elimina l'informazione dal pannello, così da renderlo utilizzabile per una successiva radiografia. I sistemi DR invece, sono

dotati di un sensore che acquisiscono i raggi X e li convertono in informazione digitale che forniscono direttamente al computer senza operare nessun tipo di processo intermedio. I sistemi CR sono meno onerosi e richiedono più tempo per l'acquisizione delle immagini, i sistemi DR sono più costosi, più efficienti e in grado di produrre immagini ad una maggiore qualità. Inoltre la tecnica digitale, rispetto a quella su pellicola, è soggetta ad una perdita di informazione di tipo spaziale poichè il punto analogico è molto più piccolo del punto digitale (pixel), sebbene l'occhio umano non percepisca questa perdita di informazione [11].

Mammografia

Un esempio specifico di radiografia è la mammografia (nota anche con la sigla MG - *Mammography*) il cui obiettivo è quello di individuare formazioni potenzialmente tumorali, lesioni e altre anomalie come micro-calcificazioni al seno. La mammella viene posizionata su un sostegno e compressa tra due piastre. La compressione garantisce l'immobilità della mammella permettendo di ottenere immagini di qualità e di ridurre la dose di radiazioni emesse. Un fascio di raggi X, prodotto dal mammografo, attraversano la mammella. L'energia dei raggi viene poi convertita in segnali elettronici in grado di produrre l'immagine. Per ogni mammella vengono generalmente acquisite due proiezioni una dall'alto e l'altra di lato.

Tomografia Computerizzata

La tomografia computerizzata (nota anche con la sigla CT - *Computed Tomography*) è un apparato con il quale si effettuano esami medici molto diffusi per la diagnosi di numerose patologie. Fa uso di fasci a raggi X che applicati nell'area del corpo oggetto di studio, permette di raccogliere un'insieme di dati. Un tubo radiogeno, a forma di ciambella che emette raggi X con diverse angolazioni, ruota attorno al paziente che giace supino su un tavolo motorizzato orizzontale. Un raggio X a seconda dell'oggetto colpito (un organo, un tessuto osseo, un vaso sanguigno, ecc..) subisce un'attenuazione. Un rilevatore, acquisisce il grado di attenuazione delle onde. Questa informazione viene utilizzata da un computer per creare l'immagine di una sezione del paziente, man mano che il lettino si muove. Al termine dell'esame, il risultato è dato dall'insieme delle sezioni della parte del corpo oggetto di studio. Ogni sezione rappresenta un'immagine 2D che può essere elaborata grazie a sofisticati algoritmi. L'insieme delle sezioni permette di ricostruire il volume in modalità 3D. La tomografia è applicabile in qualsiasi parte del corpo poichè permette di ricostruire ogni tipologia di tessuto, ossa e articolazioni. Con l'acronimo TAC (Tomografia Assiale Computerizzata) si fa riferimento alla tomografia eseguita lungo un solo asse mentre ora vengono applicate tecniche multistrato che permettono di ottenere delle ricostruzioni 3D su più assi (frontale, sagittale e assiale). A volte, per ottenere una migliore qualità delle immagini dal punto di vista della vascolarizzazione dei tessuti, può essere applicato un

mezzo di contrasto iniettato per via endovenosa. Poichè vengono utilizzati fasci a raggi X, l'eccessiva esposizione può comportare rischi per la salute umana [12, 13].

In generale, l'output ottenuto da questa strumentazione è rappresentato da un insieme di valori presenti all'interno della scala Hounsfield detta anche numero CT o nota con il simbolo HU - *Hounsfield Unit* (prende il nome dall'ingegnere britannico Godfrey Hounsfield). Quest'ultima viene utilizzata per descrivere quantitativamente la radiodensità cioè l'incapacità delle radiazioni di attraversare un determinato materiale. Si basa sul principio fisico dell'attenuazione di un materiale, per cui ad un certo valore di attenuazione corrisponde un determinato valore HU nella scala. Quest'ultimo viene ricavato dalla seguente formula:

$$HU = 1000 \times \frac{\mu_t - \mu_{acqua}}{\mu_{acqua} - \mu_{aria}} \tag{2.1}$$

in cui μ indica il coefficiente di attenuazione lineare di un materiale ovvero la proprietà che definisce il limite al quale tale materiale assorbe energia (in questo caso la radiazione ionizzante). La formula 2.1, indica il valore HU del materiale t rispetto alla radiodensità dell'acqua distillata a pressione e temperatura in condizioni standard definita come 0 HU e della radiodensità dell'aria in condizioni standard definita come -1000 HU. Poichè la densità dell'acqua pura è 0 HU, la scala di Hounsfield può assumere valori sia negativi che positivi (rispettivamente per materiali meno e più densi dell'acqua). Gli elementi aria e acqua sono stati scelti in quanto sono riferimenti universalmente disponibili. Dalla tabella 2.1 si evince che più alto è il valore, più denso è il materiale.

Materiale	HU
Aria	-1000
Polmone	-500
Tessuto adiposo	da -100 a -50
Rene	30
Sangue	da +30 a +45
Fegato	da +40 a +60
Osso	da +400 a +3000

Tabella 2.1: Valori in HU di alcuni materiali [14].

Considerando un output 2D di un tomografo, il valore di ogni pixel sarà espresso in valori HU (a partire dall'equazione 2.1 rispetto al tipo di materiale), convertibili in toni di grigio proporzionalmente diversificati dal nero al bianco assoluto [14].

Inoltre all'interno di un'immagine, alcuni valori HU possono essere *nascosti* (occlusi) per evidenziare particolari aree all'interno dell'immagine medica. Esistono due parametri che consentono di modificarli:

- *Size window* o ampiezza della finestra: consente di modificare i valori di soglia inferiore e soglia superiore sulla scala di Hounsfield per individuare un range più o meno ampio di valori HU. In questo modo tutti i valori compresi nel range, saranno visualizzati mentre gli altri saranno occlusi. Questa tecnica viene impiegata per rimuovere alcune caratteristiche dall'immagine esaltandone delle altre. Ad esempio, considerando che il potere visivo dell'osservatore è in grado di distinguere solo 40 livelli di grigio, se si sceglie un ampiezza pari a 600 UH, ogni livello di grigio corrisponderà a 15 UH (600/40). Questo significa che nell'immagine si potranno distinguere aree che differiscono per almeno 15 UH;
- Livello: parametro che, definita l'ampiezza della finestra, permette di spostarla avanti o indietro all'interno della scala di Hounsfield per individuare particolari valori HU.

Agendo sui valori della scala Hounsfield, grazie alla regolazione di questi due parametri, è possibile esaltare l'immagine preferendo la visualizzazione di aree più dense rispetto a quelle meno dense o viceversa.

Tomografia a emissione di positroni

La tomografia a emissione di positroni (nota anche con la sigla PET - *Positron Emission Tomography*) è un apparato di medicina nucleare (poichè fa uso di molecole radioattive) che prevede la somministrazione per via endovenosa di una sostanza detta radiofarmaco. In questo modo è possibile mappare il processo patologico poichè il farmaco si accumulerà nelle cellule in grado di captarlo ed emettendo positroni (antielettrone) permette di evidenziare la presenza di alcune patologie. Questo esame viene impiegato maggiormente per confermare una diagnosi di tumore, verificare la presenza di metastasi e per monitorare la variazione delle dimensioni della massa tumorale. La PET è un esame funzionale che non fornisce informazioni anatomiche, per questo motivo deve essere combinata con altri esami, ad esempio una CT. In quest'ultimo caso, dopo aver iniettato il farmaco, viene eseguita prima una CT e poi una PET, sempre con lo stesso macchinario. La CT serve per una corretta ricostruzione delle immagini e per la localizzazione anatomica delle eventuali alterazioni visibili alla PET. Successivamente viene eseguita la PET in cui il tomografo andrà ad individuare le radiazioni emesse dal farmaco, evidenziando specifiche aree dell'oggetto di studio. La PET può essere combinata anche con la risonanza magnetica [15].

Risonanza Magnetica

La risonanza magnetica (nota anche con la sigla MR - *Magnetic Resonance*) è una tecnica diagnostica che utilizza campi magnetici. Viene utilizzata principalmente per tessuti molli ma può essere eseguita su ogni tipologia di tessuto, ossa e articolazioni. Il macchinario, simile a quello di una CT, possiede uno scanner composto da un magnete principale in grado di creare un campo magnetico statico ed omogeneo ad altra frequenza. Delle bobine ausiliari sono in grado di creare ulteriori campi magnetici rotanti e in grado di

variare linearmente nello spazio. Il campo magnetico della risonanza, agisce sugli atomi di idrogeno presente nell'acqua dei tessuti, allineando il loro asse lungo quello del campo creato. Successivamente l'emissione di un impulso a radiofrequenza fa in modo che l'asse di molti atomi si allinei nella direzione opposta rispetto a quello del campo, in una condizione di elevata energia. Una volta cessato l'impulso, gli atomi ritornano allineati al campo magnetico. La velocità di rilascio energetico dato dal riallineamento dell'asse degli atomi nel campo è detta rilassamento T1 mentre l'oscillazione durante la cessione di energia è detta rilassamento T2. Valori alti di T1 e T2 corrispondono ad aree più chiare in un'immagine in toni di grigio, viceversa valori bassi corrispondono ad aree più scure [16]. L'immagine ottenuta a partire dal rilassamento T1 mostra l'anatomia dei tessuti mentre l'immagine ottenuta dal rilassamento T2 mostra i liquidi e le condizioni patologiche. Fornendo informazioni complementari entrambe sono importanti nell'attività di diagnosi. La tabella 2.2 mostra i valori di T1 e T2 relativi a diversi tessuti biologici.

Materiale	T1	T2
Muscolo Scheletrici	860	47
Cuore	860	57
Fegato	520	43
Grasso sottocutaneo	230	85

Tabella 2.2: Valori di T1 e T2 di alcuni materiali [17].

La MR è un esame multi-planare ovvero si possono acquisire le immagini su diversi piani: assiale, frontale e sagittale. Anche in questo caso può essere iniettato per via endovenosa un mezzo di contrasto che permette di aumentare la potenza del segnale di alcuni tessuti. La risonanza magnetica è un esame costoso (per questo più difficile da trovare in tutti gli ospedali), superiore rispetto ad una tomografia caratterizzata da costi più contenuti [17].

A differenza di una radiografia digitale che produce solamente un output 2D, questa strumentazione fornisce anche una ricostruzione 3D, risultando essere più accurata poichè è possibile valutare su più dimensioni la parte del corpo interessata. Confrontando la CT con una MR notiamo che la prima fa uso di raggi X mentre la seconda utilizza campi magnetici, ritenuti sicuri. La MR richiede molto più tempo rispetto a una CT ma ha il vantaggio di evidenziare maggiori dettagli nelle immagini di output.

Ecografia

L'ecografia (nota anche con la sigla US - *Ultrasound*) è una tecnica diagnostica molto diffusa ed economica che a differenza dei precedenti non utilizza né radiazioni ionizzanti, né campi magnetici ma ultrasuoni. Quest'ultimi sono onde meccaniche sonore che per definizione hanno una frequenza di 20 kHz e quelli utilizzati per l'ecografia hanno una frequenza variabile compresa tra i 2 e 15 MHz. Gli ultrasuoni sono capaci di propagarsi in mezzi detti

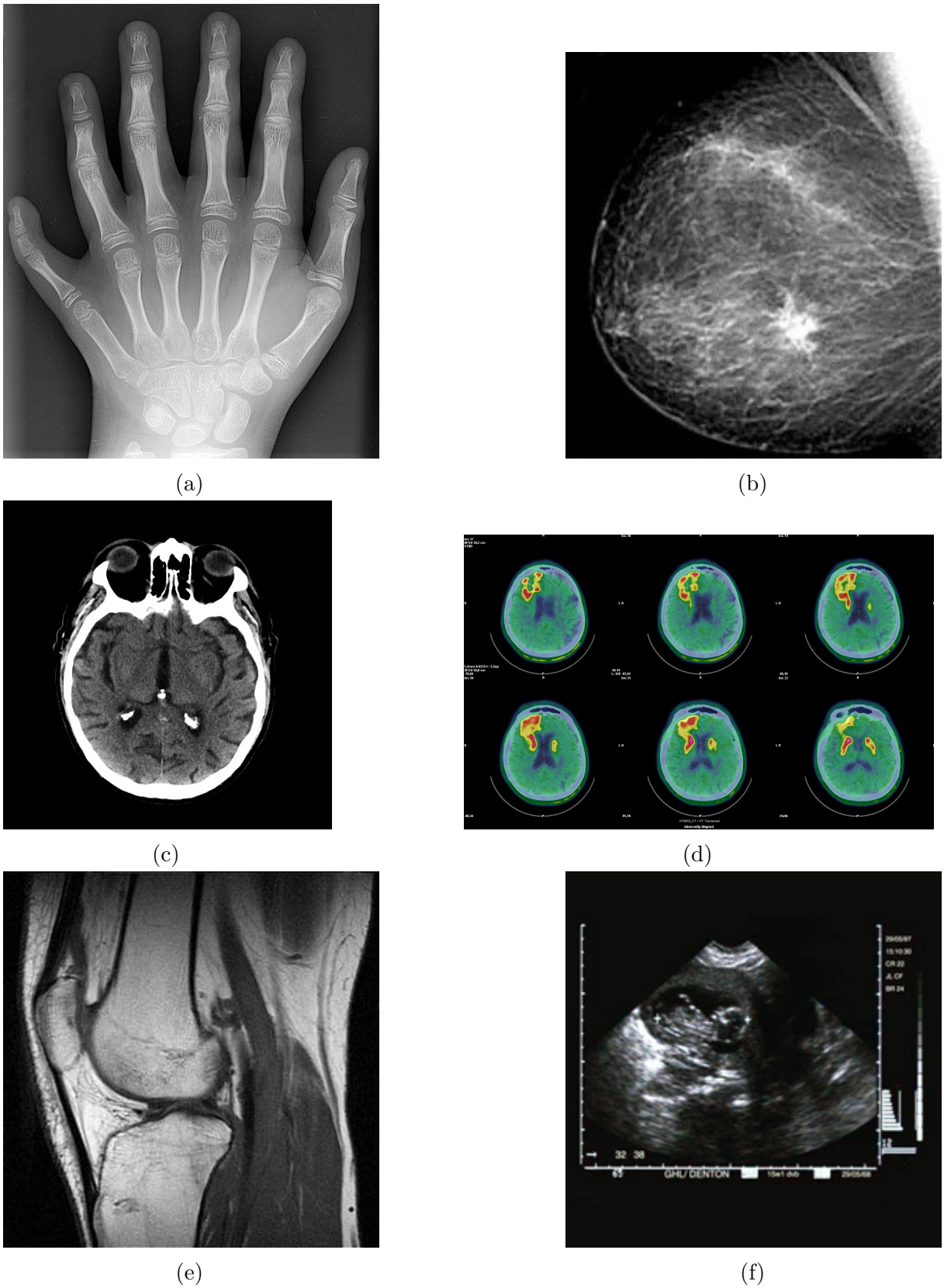
elastici come acqua, tessuti molli, tessuto osseo e flussi ematici a differenti velocità a seconda dell'elasticità e della densità del mezzo considerato. Quando gli ultrasuoni incontrano uno ostacolo cioè una discontinuità del mezzo attraversato, cambiano la loro direzione di propagazione ritornando in parte verso la sorgente. Le onde di ritorno sono caratterizzate da piccole quantità di energia e vengono definite echi. L'immagine viene formata basandosi sul principio impulso-eco: la sorgente emette delle onde sonore sotto forma di impulsi e non in maniera continuativa. Tra l'emissione di un impulso e l'altro, la sorgente riceve gli echi formando così l'immagine. La luminosità (tonalità di grigio) è proporzionale alla quantità di energia presente nell'onda di ritorno. Frequenze maggiori hanno un maggior potere risolutivo ma penetrano meno in profondità. A differenza degli altri esami, questo è operatore-dipendente poichè richiede la presenza e l'esperienza clinica di un medico in grado di utilizzare la giusta frequenza delle onde. É una metodica diagnostica non invasiva, utilizzata principalmente per lo studio dell'addome, vene, arterie e apparato muscolare. Può essere svolto anche con mezzo di contrasto, somministrato per via endovenosa, per aumentare la proprietà del sangue di riflettere segnali eco. Durante l'ecografia, nell'area da esaminare, viene applicato un gel non tossico per consentire una migliore trasmissione degli ultrasuoni [18].

La figura 2.2 mostra l'immagine prodotta da ciascuna apparecchiatura descritta. La 2.2a è una radiografia digitale di una mano sinistra mentre l'immagine 2.2b è una mammografia. La 2.2c mostra un esempio di tomografia computerizzata celebrale mentre la 2.2d mostra una sequenza di immagini durante una PET (a cui è stata applicata una LUT), in particolare è possibile notare l'accumulo del radiofarmaco man mano che questo circola nel corpo. L'immagine 2.2e mostra la risonanza magnetica di un ginocchio posta sul piano sagittale e l'ultima 2.2f è il risultato di un'ecografia.

2.2 Standard Dicom

Lo standard più comune utilizzato in ambito medico è lo standard Dicom [19] - *Digital Imaging and COmmunications in Medicine* cioè immagini digitali e comunicazioni in medicina. Si tratta di files prodotti da studi o esami medici ottenuti da differenti modalità di immagine come descritto nel paragrafo 2.1.2.

Lo standard è stato rilasciato nel 1993 come terza versione di uno già esistente chiamato ACR-NEMA (acronimo delle due associazioni statunitensi: *American College of Radiology* e *National Electrical Manufacturers Association*). L'obbiettivo era quello di definire un modello comune nell'ambito *healthcare* in modo da fornire interoperabilità tra i diversi dispositivi medici dei vari produttori. In particolare è necessario che i sistemi abbiano la capacità di cooperare e scambiare informazioni facilitandone l'interazione tra di essi.



(a) (b)

(c) (d)

(e) (f)

Figura 2.2: Esempi di immagini prodotte dalle apparecchiature descritte.

Senza dubbio quest'ultimo è stato uno dei vantaggi principali che ha portato alla diffusione dello standard. É inoltre importante considerare che la digitalizzazione di questo tipo di

dato ha permesso di semplificarne l'archiviazione: infatti prima della nascita di questo formato, gli esami venivano memorizzati su film cartacei, occupando molto spazio fisico [20]. Altre funzionalità [21] per questo tipo di formato riguardano la possibilità di eseguire *query* e trasmissione di dati su una rete per mezzo di alcuni servizi. Tra i più importanti vi sono:

- *query/retrieve*: questo servizio consente di trovare e recuperare specifici dati da un PACS (*Picture Archiving and Communication System*), tecnologia impiegata nelle organizzazioni sanitarie per archiviare, recuperare, gestire, distribuire e mostrare dati medici;
- *store*: servizio utilizzato per inviare un'insieme di dati ad un PACS o ad una *workstation*.

Un file Dicom, di estensione *.dcm*, contiene al suo interno numerose informazioni relative allo studio medico effettuato: la più importante riguarda la presenza di una o più immagini. Ogni informazione è memorizzata sotto forma di *tag*. Quest'ultimo è composto da tre parti fondamentali:

- l'attributo univoco espresso nel formato (XXXX, XXXX) composto da numeri esadecimali. Ad esempio, attributi validi sono (0010, 0010) e (0010, 0020) che si riferiscono rispettivamente al nome e id del paziente;
- la rappresentazione del valore (VR) che descrive il tipo di dato e il formato del valore dell'attributo. VR validi sono UI (*Unique Identifier*, stringa contenente un id), DT (*Date Time*, valore contente la data e l'ora) e SQ (*Sequence of Items*) una sequenza di elementi o sotto-*tags*.
- il valore del *tag* in accordo con la VR.

In ogni file Dicom è presente un'intestazione detta *header*, che contiene informazioni (metadati) riguardanti il tipo di codifica dei dati, la versione del file e diversi id univoci che caratterizzano il file stesso. Tutti gli altri dati sono organizzati sotto forma di moduli che contengono delle informazioni relative al contesto del modulo. É presente quello riferito al paziente (sesso, nome, data di nascita, ...), allo studio e alle specifiche tecniche del dispositivo medico e infine quello riferito all'immagine (posizionamento e orientazione del paziente). Tutte queste informazioni permettono al medico di avere un'ampia panoramica dell'esame effettuato.

Tra tutte le informazioni, di notevole importanza sono lo UUID (*Universally Unique IDentifier*) e l'istanza di acquisizione. Il primo, è un identificatore universale che caratterizza questi tipi di file: si tratta di una stringa numerica composta da sette parti, separate da punti aventi una specifica sintassi. Grazie a questo identificatore (essendo universale ed univoco) ogni file potrà essere individuato poiché ne esisterà sempre uno diverso, non ripetibile, per ogni file Dicom. L'istanza di acquisizione è un numero che rappresenta la fetta (o *slice*) che

equivale ad un'immagine. Ad esempio, quando si effettua una risonanza magnetica, viene prodotto un volume 3D della parte del corpo oggetto di studio. Tale volume è composto da una serie di *slice* e ogni istanza di acquisizione ne identifica una. Altri *tags* di interesse sono il *Rescale Intercept* (0028, 1052) e il *Rescale Slope* (0028,1053) che permettono di ottenere gli HU a partire da un'immagine in formato Dicom. Tutti i *tags* definiti nello standard possono essere consultati sul *sito* ufficiale dello standard [22].

Nella maggior parte dei casi, l'immagine viene archiviata secondo la codifica con la quale viene prodotta, senza applicare nessun algoritmo di compressione. In altri casi, viene eseguita una compressione detta *lossless* che non porta alla perdita dell'informazione originale. In caso contrario, la perdita di informazione risulterebbe essere un grave problema poichè può distorcere l'immagine (anche per piccolissime differenze non visibili ad occhio nudo) che possono influenzare l'attività di analisi diagnostica.

2.3 Segmentazione di una Regione di Interesse

Come definito nell'introduzione del Capitolo 2, per segmentazione si intende l'individuazione del contorno all'interno di un'immagine, di una o più ROI rispetto a tutta la parte di *background*. In ambito medico, queste Regioni possono riferirsi a lesioni di interesse nell'indagine diagnostica. In particolare, l'attività consiste nell'individuare i pixel (in un'immagine 2D) o voxel (in un volume 3D) appartenenti al contorno della lesione [23]. La figura 2.3 mostra l'output ottenuto dopo una segmentazione in un cervello: si noti il contorno individuato della lesione.

Figura 2.3: Risultato di una segmentazione di una lesione in un cervello.

Tale attività risulta essere molto complessa poichè ci sono tanti fattori che possono pregiudicarne il risultato, alcuni di essi sono: la presenza di rumore nelle immagini, texture difficili da analizzare (superfici non omogenee caratterizzate da trame più o meno regolari),

le lesioni possono avere forme articolate ed irregolari.

La segmentazione risulta essere fondamentale in tutti i casi in cui è necessario classificare l'oggetto. Inoltre permette di tracciarlo, tramite l'estrazione di informazioni volumetriche in modo da poterlo confrontare nel tempo e capire se il suo volume è aumentato o diminuito. Lo standard Dicom definisce il modulo *Surface Segmentation* per descrivere le segmentazioni presenti all'interno dei dati. Il *tag Segment Sequence* (0062, 0002) descrive i segmenti che compongono una ROI e contiene al suo interno ulteriori *tags*, tra cui il *Segment Algorithm Type* (0062, 0008) che descrive il tipo di algoritmo utilizzato (automatico, semi-automatico o manuale) per generare i segmenti. Altre informazioni che possono essere salvate riguardano il numero associato al segmento, una *label* e una descrizione utilizzata dall'operatore per salvare diverse informazioni utili.

La segmentazione può essere eseguita sia in modalità assistita che in modalità automatica. Nel primo caso, il medico provvederà a disegnare la ROI della lesione tramite un software dedicato. Questo è necessario per la raccolta dei dati di addestramento da utilizzare come *ground truth*. Durante la segmentazione assistita vi sono due principali tipologie di errore:

- l'errore intra-operatore: errore commesso dallo stesso operatore con una certa variabilità, su misure ripetute, relative alla stessa lesione o lesioni differenti;
- l'errore inter-operatore: errore commesso da più operatori. Questo tipo di errore può dipendere da diversi fattori, ad esempio l'utilizzo di metodologie differenti che portano a diversi risultati di segmentazione.

Nel caso della segmentazione automatica vengono impiegati software in grado di individuare le lesioni. Tra i sistemi utilizzati vi sono i CAD (*Computer Aided Detection)* come descritto in seguito (paragrafo 2.5). A volte sono di elevata complessità, rendendo difficile l'utilizzo da parte dell'operatore, in quanto non esperto nell'usare i sistemi informatici. É importante sottolineare come questi sistemi non sostituiscono il medico ma forniscono un secondo parere (individuazione e diagnosi assistita) che può mettere in discussione o confermare quello del medico. Alcuni *tools* gratuiti per la segmentazione sono:

- ImageJ [24]: programma di elaborazione digitale delle immagini sviluppato dal *National Institutes of Health* (NIH). Sono presenti numerosi *plugins* che permettono di eseguire funzionalità aggiuntive relative all'acquisizione, l'analisi ed il *processing* delle immagini. In questo caso la segmentazione è solo assistita, eseguita da un operatore;
- MedSeg [25]: *tool* di segmentazione *web-based* che non richiede l'installazione locale sulla macchina. Offre entrambe le modalità: assistita e automatica. É possibile utilizzare anche modelli di intelligenza artificiale pre-addestrati;

- Medviso [26]: sviluppato in collaborazione con l'università di Lund. É un software che tra le tante funzionalità offre anche quelle per la segmentazione e l'analisi della lesione sia in modalità assistita che automatica. Sono presenti anche strumenti per lavorare su output 3D.

Un altro software utilizzato per l'*imaging* medico e per la segmentazione assistita è Aliza Medical Software [27]: in questo caso però si tratta di un programma utilizzabile solo acquistando una licenza valida.

Il metodo più semplice per segmentare un'immagine si basa sulla binarizzazione (*thresholding*) tramite le scelta di una soglia ottimale: se un pixel ha un valore maggiore o uguale alla soglia, questo assume il valore di uno (bianco), altrimenti zero (nero). L'output è ovviamente un'immagine binaria. Nella maggior parte dei casi, tale metodo non è però in grado di gestire sfondi a intensità variabile ed incapace di segmentare oggetti non uniformi ed irregolari. Per questo motivo, il *thresholding* non viene utilizzato in ambito medico a favore di altre metodologie che utilizzano tecniche di *machine learning* (ML) e di *deep learning* (DL). Entrambi i metodi si basano sul calcolo ed estrazione delle *features*: caratteristiche misurabili in grado di discriminare gli oggetti individuati. Questa fase è cruciale poichè saranno proprio le *features* che determineranno la qualità del risultato. Esse possono essere suddivise in tre categorie principali: forma, colore e tessitura [28].

Le tecniche di ML e di DL si differenziano per alcuni aspetti [29]: le prime prevedono l'implementazione di algoritmi in grado di estrarre *features* dai dati di input per addestrare un modello che persegue l'obbiettivo prefissato. Nelle seconde invece, viene addestrata una rete neurale artificiale pensata per simulare i processi biologici del nostro cervello umano. La differenza principale è data da come queste *features* vengono estratte, in cui nel primo caso sono ottenute da algoritmi implementati e modellati da un esperto mentre nel caso del DL le *features* vengono estratte automaticamente dalla rete neurale. Altre differenze riguardano la quantità dei dati di input necessari per addestrare il modello: le tecniche di ML possono lavorare anche con ridotte quantità di dati mentre le seconde necessitano di elevate quantità. Differenti sono anche i tempi di addestramento del modello: contenuti nel caso del ML, molto più lunghi (anche giorni) nel caso del DL.

Una tecnica ampiamente utilizzata per l'identificazione e successiva segmentazione di una lesione è la *pattern recognition* ovvero l'individuazione di *patterns* all'interno delle immagini medicali. Si definisce *pattern*, uno schema comune presente all'interno dei dati diagnostici. La presenza di uno o più *patterns* permette al software CAD, di creare un'insieme di relazioni che rappresentano una situazione di presenza o assenza di una patologia (*detection*) o di specifica caratterizzazione di una lesione (*diagnosis*). Ogni qualvolta che l'algoritmo identifica un *pattern*, questo viene assegnato ad una classe del problema (ad esempio tessuto sa-

no o patologico) o ad una osservazione X (ad esempio un pixel, un organo o una lesione) [29].

Altre tecniche impiegate nel campo del ML fanno uso di algoritmi di *clustering*. Quest'ultimi si basano sulla ricerca di raggruppamenti detti *cluster*, all'interno dei dati di input. Ogni *cluster*, che corrisponde ad una classe del problema, conterrà dati aventi *features* simili tra loro. Alcuni algoritmi utilizzati per questo scopo sono il *Mean Shift* [30] e il *K-Means* [31]. Per quanto riguarda l'ambito del DL, un esempio di rete neurale ampiamente utilizzata in ambito medico è la U-Net [32]. Questa rete è nata proprio per scopi medicali. É detta U-Net per la sua tipica forma ad U che individua le due componenti principali della sua architettura: un encoder e un decoder per codificare e decodificare rispettivamente le informazioni.

Solitamente, le immagini prima di essere utilizzate per la segmentazione, vengono *migliorate* per far emergere le *features* da individuare. Questa fase è detta *pre-processing* dei dati che include ad esempio il filtraggio per la pulizia del rumore e normalizzazione delle immagini [29].

2.4 Classificazione di una Regione di Interesse

L'attività di classificazione si basa nell'identificare due o più classi all'interno dei dati di input (immagini e/o video). Dopo aver segmentato l'immagine e individuato le lesioni, è necessario classificarle per stabilire ad esempio, se queste sono di tipo tumorale o meno e se necessitano di un ulteriore approfondimento da parte del medico. Anche in questo caso, i software utilizzati per compiere questo *task* utilizzano tecniche di intelligenza artificiale di ML e DL.

Nell'ambito del ML, la macchina a vettori di supporto [33] (*Support Vector Machines* - SVM) è un noto algoritmo per svolgere *task* di classificazione per problemi bi-classe, esteso poi anche a problemi multi-classe. Poichè questo lavoro di Tesi riguarderà anche l'addestramento di questo tipo di classificatore, verrà fornita una descrizione più dettagliata.

L'idea alla base, è quello di determinare una superficie (ad esempio una linea o un iperpiano a seconda della dimensione dei dati di input) che separi al meglio le classi del problema. In particolare l'obbiettivo finale non è solamente quello di determinare la superficie ma di far si che questa massimizzi il margine definito come la minima distanza tra i campioni delle differenti classi e la superficie individuata. La massimizzazione permette al classificatore di avere buone capacità di generalizzazione. L'immagine 2.4 mostra un esempio grafico di quanto descritto.

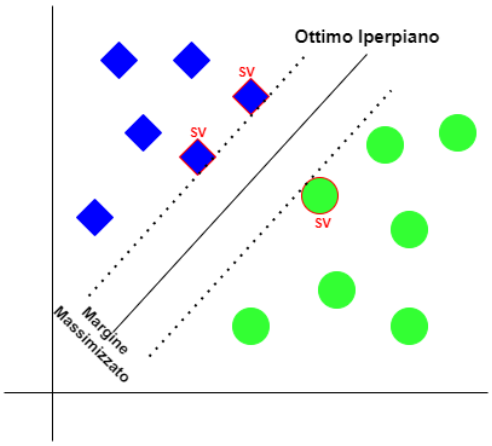


Figura 2.4: Rappresentazione grafica del margine calcolato da un SVM lineare.

In questo caso sono presenti le classi rombo e cerchio e la linea nera al centro identifica la superficie di separazione con distanza minima tra i campioni delle classi. I campioni più vicini alla linea rappresentano i punti più estremi della distribuzione e la loro distanza rappresenta la misura del margine di separazione tra le due categorie, essi sono definiti vettori di supporto (*support vectors* - SV). Sono proprio questi ultimi che definiscono la soluzione del problema e determinano la qualità del risultato.

Considerando come superficie un iperpiano, la sua definizione può essere definita come seguente [34]:

$$g(x) = \vec{w} \cdot \vec{x} + b \tag{2.2}$$

in cui:

- $\vec{w}$ è il vettore normale dell'iperpiano che determina la sua direzione;
- $\vec{x}$ è il vettore di input;
- b è uno scalare che identifica la posizione dell'iperpiano nello spazio.

Tutti i punti per cui $g(x) > 0$ giacciono sopra al piano, tutti i punti in cui $g(x) < 0$ sono sotto al piano.

Esistono due tipologie principali di SVM:

- SVM lineare: in cui i dati utilizzati per l'addestramento del modello sono separabili linearmente;
- SVM non lineare: in cui i dati utilizzati non sono separabili linearmente e non esiste un'ipotesi sulla loro separabilità. In questo caso è necessario individuare una superficie di separazione più complessa rispetto al caso lineare.

Nel caso lineare, dato un insieme di n campioni etichettati $(\overrightarrow{x_1}, y_1), ...(\overrightarrow{x_n}, y_n)$ in cui ogni $\overrightarrow{x_i}$ è un vettore p-dimensionale e y_i rappresenta la classe {+1, -1} a cui appartiene $\overrightarrow{x_i}$, vengono definiti linearmente separabili se esiste un vettore $\overrightarrow{w}$ e uno scalare b che soddisfano le seguenti disuguaglianze:

$$\begin{aligned} \overrightarrow{w} \cdot \overrightarrow{x_i} + b \geq 1, \text{ se } y_i = 1, \\ \overrightarrow{w} \cdot \overrightarrow{x_i} + b \leq -1, \text{ se } y_i = -1 \end{aligned} \tag{2.3}$$

valide per tutti i campioni di input per $i = 1, .., n$.
L'equazione 2.3 può essere riscritta nella forma compatta:

$$y_i(\overrightarrow{w} \cdot \overrightarrow{x_i} + b) \geq 1, \ i = 1, ..., n \tag{2.4}$$

Si definisce iperpiano ottimo:

$$\overrightarrow{w} \cdot \vec{x} + b = 0 \tag{2.5}$$

l'unico iperpiano che separa i campioni con margine massimo: esso determina la direzione $\frac{w}{\|w\|}$ dove la distanza tra le proiezioni dei vettori di input delle due differenti classi è massima. Tutti i vettori $\overrightarrow{x_i}$ per cui $y_i(\overrightarrow{w} \cdot \overrightarrow{x_i} + b) = 1$ sono i vettori di supporto. La distanza ρ (calcolata come la distanza da un punto ad un piano) è calcolata come seguente:

$$\rho = \frac{\|\overrightarrow{w} \cdot \vec{x} + b\|}{\|\overrightarrow{w}\|} \tag{2.6}$$

Considerando due punti di classe opposta che soddisfino l'uguaglianza 2.4, il margine può essere ricavato dall'equazione 2.6 come seguente:

$$\rho = \frac{2}{\|\overrightarrow{w}\|} = \frac{2}{\sqrt{\overrightarrow{w} \cdot \overrightarrow{w}}} \tag{2.7}$$

Questo significa che l'iperpiano ottimo è l'unico che massimizza la distanza ρ sotto i vincoli dell'equazione 2.3 o minimizza la quantità inversa cioè $\frac{||\overrightarrow{w}||^2}{2}$. Pertanto, si tratta di risolvere un problema di programmazione quadratica convessa con vincoli lineari. Tale problema è possibile risolverlo attraverso la formulazione Lagrangiana sotto le condizioni KKT (*Karush-Kuhn-Tucker*) [34].

Quando i campioni non sono tutti linearmente separabili è necessario rilassare i vincoli di separazione per ammettere alcuni errori di classificazione. In questo caso si introducono n variabili positive dette *slack* ξ in cui codificano la deviazione del margine per ogni campione. L'equazione 2.4 viene così modificata:

$$y_i(\overrightarrow{w} \cdot \overrightarrow{x_i} + b) \geq 1 - \xi_i, \ i = 1, ..., n \tag{2.8}$$

in cui la *slack* sarà $\xi = 0$ per campioni correttamente classificati e $\xi > 0$ per campioni erroneamente classificati. In questo caso, l'iperpiano ottimo dovrà ancora massimizzare il margine ma allo stesso tempo, minimizzare il numero di campioni erroneamente classificati. La quantità da minimizzare risulterà essere la seguente, considerando i campioni non correttamente classificati:

$$\frac{||\vec{w}||^2}{2} + C\sum_{i=0}^{n}\xi_i \tag{2.9}$$

Il coefficiente C viene introdotto per indicare il peso da attribuire agli errori commessi dal classificatore rispetto all'ampiezza del margine.

Nel caso di SVM non-lineare, l'idea è quella di eseguire un mapping non-lineare in cui i dati di input appartenenti allo spazio $\mathfrak{R}^d$, vengono mappati in uno spazio $\mathfrak{R}^m$ con una dimensionalità più alta rispetto a quella di partenza ($m > d$). In questo modo si hanno maggiori gradi di libertà. Per eseguire il mapping, vengono utilizzate particolari funzioni dette *kernel function* (K), in cui è possibile ricondurre il prodotto di due campioni, mappati nello spazio $\mathfrak{R}^m$, ad una funzione K dei due campioni originali nello spazio $\mathfrak{R}^d$. Le *kernel function* più conosciute sono: il polinomio di grado q e la *Radial Basis Function* (RBF) [34, 35].

Giocano un ruolo molto importante anche gli iper-parametri ovvero parametri che regolano la fase di addestramento affinché il modello raggiunga ottimi risultati. Poichè non è possibile sapere a priori il valore degli iper-parametri (variano a seconda del contesto e tipo di problema) è necessario eseguire su di essi un'accurata fase di *tuning* (ottimizzazione) per massimizzare i risultati. Tra i metodi di ottimizzazione più comuni vi sono il *Grid Search* e il *Random Search*: entrambi prevedono la necessità di fornire in input, un insieme di valori da valutare, per ogni iper-parametro.

Nel *Grid Search*, il modello migliore sarà ottenuto eseguendo una ricerca esaustiva su tutte le possibili combinazioni dei valori di tutti gli iperparametri. Il principale svantaggio è dato dall'alto costo computazionale richiesto per effettuare tutte le combinazioni. Per questo motivo, tale ottimizzazione è opportuna utilizzarla quando gli iper-parametri sono limitati. Viceversa, eseguendo tutte le combinazioni possibili, troverà nella maggior parte dei casi il modello più performante.

Nel caso del *Random Search*, viene scelto casualmente un sottoinsieme di tutte le possibili combinazioni dei valori forniti in input, su un numero di iterazioni prefissato (fornito anch'esso come input). Il costo computazionale sarà quindi minore rispetto al *Grid Search*, permettendo di ottenere buoni risultati, anche se meno accurato nel ricercare il modello migliore poichè non c'è certezza che, nella scelta randomica, si otterrà il modello più performante. Entrambi gli ottimizzatori si basano sulla ricerca di tipo *uninformed* ovvero non

vi è un apprendimento incrementale a partire dalle iterazioni passate e future, ognuna di esse è indipendente dalle altre. La scelta dell'ottimizzatore adeguato è quindi determinato principalmente dal numero di iper-parametri presenti che è necessario ottimizzare.

L'addestramento di un modello SVM prevede la definizione di un *dataset* contenente i dati da utilizzare per la fase di addestramento e test. Tale *dataset* viene suddiviso in *Training Set*, *Validation Set* e *Test Set*. Il primo, contiene l'insieme dei dati da utilizzare esclusivamente per l'addestramento del modello, il secondo contiene l'insieme dei dati su cui tarare gli iper-parametri, mentre l'ultimo contiene i dati per testare le performance sul modello addestrato. La tecnica di apprendimento utilizzata è detta addestramento supervisionato in cui vengono assegnate delle etichette (classi) ai dati, che rappresentano le verità. Tali etichette vengono stabilite da un professionista in base al contesto del problema (ad esempio in ambito clinico può essere un medico). L'obiettivo principale nell'addestrare un modello di intelligenza artificiale risiede nella convergenza dell'algoritmo ovvero fare in modo che il modello sia in grado di generalizzare con elevata precisione anche su nuovi dati, mai visti. Il problema principale in cui si può incorrere durante l'addestramento è l'*overfitting* ovvero quando il modello non riesce a generalizzare: apprende correttamente i dati di *training* ma non riesce ad eseguire corrette predizioni sui dati di *test*. Un tipico esempio è quando si ottiene un'elevata accuratezza sul *Training Set* e bassa accuratezza sul *Test Set*. Alcuni fattori che possono causare l'overfitting sono: *dataset* di piccole dimensioni o sbilanciati e eccessivi gradi di libertà sul modello [36].

Una tecnica che può essere impiegata per ridurre o prevenire l'*overfitting* è la *K-Fold Cross Validation* che prevede la suddivisione di tutti i dati di addestramento in K partizioni (*fold*) di uguali dimensioni. Vengono eseguite tante iterazioni quante sono le partizioni e ad ogni iterazione viene scelta una *fold* come *Validation* e le rimanenti come *Training*.

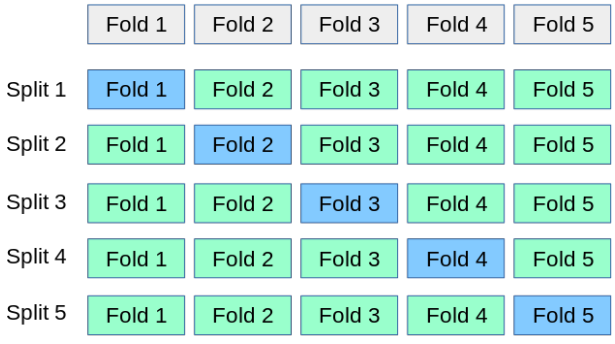


Figura 2.5: Esempio di *K-Fold Cross Validation* con 5 partizioni.

La figura 2.5 mostra un esempio di *5-Fold Cross Validation*: i dati di *training* sono suddivisi in 5 partizioni. Ad ogni *split* (iterazione), la *fold* celeste verrà considerata come *Validation* e le altre (*folds* verdi) come *Training*. Al termine dell'addestramento verrà scelto il modello con

le prestazioni migliori.

Per valutare un modello di intelligenza artificiale è necessario calcolare le metriche ovvero formule matematiche che permettono di rappresentare (e fornire un'idea) dell'efficacia del modello. Esistono diverse metriche utilizzabili a seconda del contesto e della tipologia del problema che si sta affrontando. Quella più utilizzata è sicuramente l'accuratezza. Dato un problema a due classi (*Positive* e *Negative*) ed N l'insieme dei risultati predetti, l'accuratezza (A) è definita come il rapporto tra le classificazioni corrette e tutti i risultati effettuati:

$$A = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.10}$$

in cui, quando un modello classifica:

- correttamente la classe positiva, il risultato è definito TP (*True Positive*).
- correttamente la classe negativa, il risultato è definito TN (*True Negative*);
- erroneamente la classe negativa come classe positiva, il risultato è definito FP (*False Positive*);
- erroneamente la classe positiva come classe negativa, il risultato è definito FN (*False Negative*).

L'accuratezza del modello non è sufficiente come metrica di valutazione, per questo motivo vengono utilizzate ulteriori metriche. Le più usate in ambito medico sono:

- la sensibilità definita come $\frac{TP}{TP+FN}$. In letteratura questa metrica si può trovare anche con il nome di TPR (*True Positive Rate*) o *recall* ed indica la selettività cioè se il modello, in termini quantitativi, è in grado di fare previsioni corrette sui dati di classe positiva;
- la specificità o TNR (*True Negative Rate*) definita come $\frac{TN}{TN+FP}$ indica in termini quantitativi se il modello è in grado di fare previsioni corrette sui dati di classe negativa;
- la PPV (*Positive Predictive Value*) o *precision* definita come $\frac{TP}{TP+FP}$ indica quanto il modello è preciso, cioè la probabilità che una previsione sia effettivamente di classe positiva se il modello ha predetto questa classe per quel dato di input;
- la NPV (*Negative Predictive Value*) definita come $\frac{TN}{TN+FN}$ indica la stessa probabilità della PPV considerando però il caso di classe negativa.

Quando si addestrano modelli di questo tipo è necessario scegliere una soglia cioè un valore che permette di stabilire se il risultato predetto dal modello, può essere considerato

accettabile o meno. Quando il modello predice un risultato fornisce una probabilità, se tale valore è maggiore rispetto alla soglia viene considerato, altrimenti viene scartato. La scelta della soglia permette di bilanciare classificazioni corrette da quelle errate. Soglie elevate (restrittive) riducono i FP a discapito dei FN, viceversa avviene per soglie basse (tolleranti). Se consideriamo i FPR e i FNR (la *False Positive Rate* calcolato come $\frac{FP}{\#\text{ tutti i casi positivi}}$ e la *False Negative Rate* calcolato come $\frac{FN}{\#\text{ tutti i casi negativi}}$) come funzioni rispetto ad una ipotetica soglia, queste possono essere condensate in alcuni grafici. Piuttosto utilizzata è la ROC (*Receiver Operating Characteristic*) [37] che in ordinata riporta il TPR $(1 - FNR)$ mentre in ascissa riporta il FPR come illustrato nell'immagine 2.6.

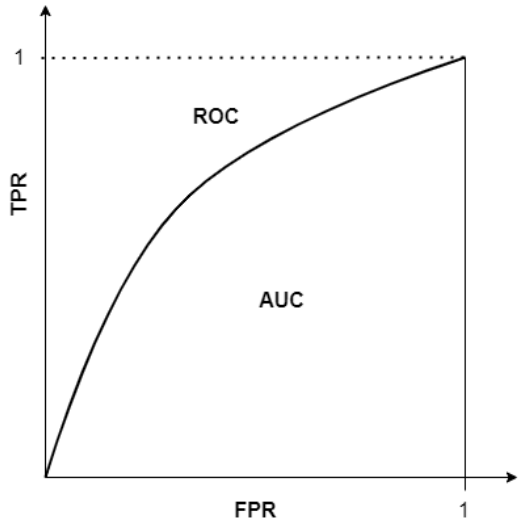


Figura 2.6: Esempio di ROC e AUC.

La ROC è la curva mostrata, mentre l'area sotto la curva è chiamata *Area Under Curve* (AUC). Quest'ultima indica la prestazione media di un modello (è possibile riassumere le prestazioni in una singola misura) e permette appunto di poter fare un confronto tra più classificatori per selezionare quello migliore.

Nel caso del DL, molte sono le reti neurali presenti in letteratura per svolgere compiti di classificazione. Restano validi i cenni fatti sugli iper-parametri e metriche da utilizzare. Tra le reti più note vengono indicate: la VGGNet, la ResNet e la GoogLeNet.

2.5 Principali Applicazioni

In questo paragrafo verranno illustrate alcune applicazioni in ambito medico, applicate a contesti reali. Queste ultime forniscono supporto ai medici nella cura del paziente.

CyberKnife

Si tratta di un dispositivo elettromedicale radioterapico utilizzato per debellare le lesioni degli organi (metastasi, meningiomi) [38]. É composto da diverse parti meccaniche tra cui un braccio robotico e due sistemi di *imaging*. Il primo sistema permette di individuare la posizione del paziente per correggere la direzione del braccio. Il secondo detto Synchrony® permette di effettuare il *tracking real-time* della posizione della lesione per sincronizzare il movimento del fascio di trattamento a raggi X. Si tratta di una tecnica non invasiva che permette di ridurre gli interventi chirurgici. É possibile utilizzare questo dispositivo solo per lesioni di modeste dimensioni.

Figura 2.7: Dispositivo CyberKnife.

Il paziente prima del trattamento, è sottoposto ad un esame medico per la localizzazione della lesione (tomografia, risonanza magnetica) che servirà a *calibrare* il dispositivo CyberKnife. Il trattamento è composto da due fasi: in primo luogo, il paziente viene posizionato sul lettino per permettere al sistema di individuare la lesione. Successivamente il dispositivo emette un fascio di radiazioni ionizzanti, grazie ad un sistema incrociato di raggi X stereoscopici, mirato sulla lesione oggetto di studio per evitare di colpire tessuti sani. Per tutta questa fase, il dispositivo effettua un monitoraggio continuo per correggere la direzione del braccio rispetto a piccoli movimenti del paziente (ad esempio respirazione) o movimenti bruschi.

Laparascopia con Realtà Aumentata

La laparascopia è un intervento chirurgico mininvasivo che prevede l'introduzione di un tubo sottile (laparoscopio) attraverso un piccolo taglio nell'addome. L'utensile è dotato di una fonte luminosa e di una videocamera in grado di acquisire le immagini e proiettarle in un monitor. In questo modo l'operatore è in grado di visualizzare i dettagli dei tessuti. La tecnica è utilizzata principalmente per indagini diagnostiche e asportazioni di campioni di tessuto da analizzare. Il progresso tecnologico ha permesso l'impiego della realtà aumentata in questo tipo di tecnica, attraverso la ricostruzione tridimensionale dell'oggetto di studio [3, 39, 40]. L'equipe medica, dopo aver indossato speciali occhiali 3D [41] (visori che permettono di vedere la realtà associata ad ologrammi virtuali), avrà la percezione di immergersi all'interno del corpo umano. L'ologramma di un organo riproduce fedelmente la realtà su tre dimensioni permettendo di visualizzare e manipolare, proprio con le mani,

il modello 3D in modo da poter interagire con esso: spostarlo, ingrandirlo ed applicarlo al campo operatorio sull'organo reale. In questo modo, il medico chirurgo riuscirà ad orientarsi con i punti di riferimento dell'organo oggetto di analisi. Tutto ciò porta ad un aumento della percentuale di pazienti operabili ed una riduzione dei costi poichè grazie all'ologramma non sarà più necessario realizzare modelli 3D fisici ma solo virtuali a fronte di una spesa iniziale per l'acquisto dei visori e del software. Per fare un confronto, il costo degli occhiali 3D e del software (utilizzabili per tutti gli interventi) è di circa 30.000€ mentre ogni modellino fisico per ogni intervento può costare dai 1.500€ ai 3.000€.

Sistemi CAD

I sistemi CAD sono piattaforme informatiche che forniscono ausilio al medico radiologo in fase di indagine diagnostica [42]. Anche in questo caso lo sviluppo tecnologico ed in particolare l'intelligenza artificiale, ha fornito un notevole contribuito poichè ha permesso di rendere questi sistemi ancora più accurati. Il loro obiettivo è quello di individuare a partire da uno studio, un insieme di *features* predittive. Esse rappresentano delle caratteristiche numeriche misurabili. Grazie alle *features* individuate, questi sistemi sono in grado di determinare all'interno di immagini biomedicali, se ad esempio sono presenti delle lesioni tumorali o aree sospette. Oltre alla fase di individuazione sono in grado di fare diagnosi, ovvero fornire una probabilità che indica se è presente o meno un tumore e sulla sua natura: benigna o maligna. Come già accennato, è importante far notare come questi sistemi non sostituiscono il medico ma forniscono un secondo parere (individuazione e diagnosi assistita) che può mettere in discussione o confermare quello del medico. L'impiego dei sistemi CAD ha dimostrato come la sensibilità di diagnosi sia aumentata, riducendo il tasso di errore. Viceversa, l'aumento di sensibilità tende a sovrastimare la presenza di lesioni, soprattutto di falsi positivi, portando ad un maggiore approfondimento diagnostico da parte del medico radiologo [43].

Nel panorama medico esistono due tipologie di sistemi CAD: i CADe (*Computer Aided Detection*) e i CADx (*Computer Aided Diagnosis*). I primi si focalizzano sulla individuazione di lesioni nelle immagini, i secondi invece si focalizzano sulla diagnosi e classificazione delle lesioni precedentemente individuate. La differenza è data quindi dall'obbiettivo finale da perseguire. L'output dei sistemi CADe è dato dal posizionamento delle lesioni a cui viene applicato un contornamento automatico dei margini mentre l'output dei sistemi CADx è uno *score* o valore di probabilità che indica se una specifica lesione appartenga ad una determinata classe rispetto ad un'altra. Poichè le tecniche impiegate da questi sistemi sono simili, quando si parla di un sistema CAD ci si riferisce sia alla fase di *detection* che a quella di *diagnosis*. Ad esempio, il funzionamento di un sistema CAD di individuazione e classificazione delle lesioni, a prescindere dallo specifico caso d'uso, può essere generalizzato nelle seguenti fasi [44]:

1. Ricostruzione tridimensionale dell'immagine del tessuto a partire dall'esame effettuato (CT, MR , ...);

2. Fase di *detection*: ricerca all'interno del tessuto le lesioni (ROI) potenzialmente dannose;

3. Fase di *diagnosis*: classificazione di ogni lesione individuata restituendo come risultato uno *score*. Se lo *score* supera una certa soglia allora il sistema attribuisce la lesione come patologica.

Altre Applicazioni

Esistono inoltre ulteriori applicazioni della *computer vision* in ambito medico che non forniscono un supporto diretto nella cura del paziente ma indiretto, utili al personale medico. Un esempio è il monitoraggio di un paziente a domicilio grazie alla quale i medici possono controllare lo stato di salute in remoto. Altri esempi riguardano l'automatizzazione nella gestione di farmaci e prodotti sanitari in ospedali e strutture sanitarie.

2.6 Caso di Studio

Il caso di studio di questo elaborato, relativo alla tipologia di lesione analizzata, ha come focus l'organo della prostata. L'AIRC (Associazione Italiana Ricerca Cancro), indica come questo tumore risulta essere quello più frequente nel sesso maschile (circa il 18,5% di tutti i tumori diagnosticati nell'uomo). É stato stimato che ogni anno provoca circa 40 mila casi e 7 mila decessi ma nonostante l'alta incidenza, il rischio che la malattia abbia un esito fatale risulta essere basso [45].

La prostata è una ghiandola presente solo nel sesso maschile, la cui principale funzione è quella di produrre liquido seminale. Dopo la soglia dei 50 anni, questo organo tende ad andare incontro a diverse patologie tra cui il tumore alla prostata. In questa situazione, il volume che in condizioni normali è simile ad una noce, tende ad aumentare. La causa principale di questo tumore è dato dall'accrescimento incontrollato delle cellule presenti all'interno della ghiandola [45]. Alcuni fattori di rischio che possono aumentare la probabilità sono l'ereditarietà, il fumo, l'alimentazione e la vita sedentaria.

Il tumore alla prostata è generalmente asintomatico, sopratutto nella fasi iniziali. Quando la massa tumorale aumenta possono essere avvertiti alcuni sintomi urinari tra cui: la difficoltà a urinare o bisogno di urinare spesso, dolore quando si urina e la perdita di sangue. Nei casi più gravi, in presenta di metastasi, gli altri organi interessati sono: il polmone, il tessuto osseo e le stazioni linfatiche (linfonodi) medie e profonde.

Tutti i soggetti che presentano una sintomatologia che può essere associata ad una patologia prostatica, vengono valutati da personale medico specialistico mediante visita urologica, nella quale viene effettuata una Esplorazione Rettale digitale (ER). Questo esame permette di valutare la forma, la consistenza e la regolarità della ghiandola. Successivamente, in caso di sospetto clinico si procede con una biopsia e una MR. Dal risultato della MR, è possibile individuare ed analizzare particolari aree sospette.

Il tumore alla prostata risulta essere una malattia eterogenea dal punto di vista biologico: sono presenti delle forme molto aggressive in grado di espandersi rapidamente, portando dopo poco tempo alla morte del paziente. Esistono altri casi in cui il tumore è indolente cioè a lenta espansione ed incapace di causare gravi danni al paziente, anche con il passare di molti anni.

Il punteggio di *Gleason* (o *Gleason Score* dal nome del medico patologo americano Donald Floyd Gleason) è uno dei principali indicatori che permette di descrivere la tipologia di tumore appena descritto. Il punteggio viene calcolato assegnando un valore compreso tra 1 e 5 in base alla malignità, alle due aree tumorali maggiormente presenti dei quali, il primo è il *pattern* più rappresentato (a cui viene attribuita maggiore importanza) e il secondo è quello meno rappresentato [46]. Infine vengono sommati i valori. L'analisi del tessuto tumorale può essere effettuata al microscopio individuando le caratteristiche che permettono di discriminare l'osservazione. Un valore pari ad 1 indica una forma tumorale con aspetti tissutali simili a quelli della ghiandola in condizioni normali mentre un valore pari a 5, indica una forma tumorale indifferenziata ovvero avente delle caratteristiche morfologiche molto diverse rispetto ad una prostata non malata. Sommando i valori delle due aree tumorali si ottiene un punteggio compreso tra 2 e 10 (valori inferiori a 5 sono molto rari) che indica una correlazione con diversi fattori, tra cui l'ereditarietà [45, 46]. Lo *score* finale viene così descritto:

- un punteggio pari o inferiore a 6 indica un tumore a basso rischio;
- un punteggio pari a 7 indica un tumore a rischio intermedio;
- un punteggio pari a 8 e superiore indica un tumore ad alto rischio.

Nella pratica clinica, i valori di *Gleason* sono classificati, in ordine crescente di malignità, nel seguente modo:

- *Score* 6: 3+3;
- *Score* 7A: 3+4;
- *Score* 7B: 4+3;

- *Score* 8: 4+4;
- *Score* 9: 4+5 e 5+4;
- *Score* 10: 5+5.

in cui considerando lo *score* della forma $N + M$, con $(N > M)$ tale punteggio indica che all'interno del campione il *pattern* N risulta essere maggiormente rappresentato rispetto al *pattern* M.

3 Progetto della soluzione

Questo capitolo illustra la fase progettuale della soluzione: essa include l'analisi dei requisiti che descrivono le funzionalità nella loro interezza, e la scelta del software da utilizzare come base di partenza. Verrà poi presentata l'architettura del sistema, mostrando ad alto livello la struttura complessiva in termini di moduli e come questi interagiscono tra loro.

3.1 Analisi dei Requisiti

L'analisi dei requisiti è un'attività fondamentale che permette di individuare i requisiti (*software requirements*) che descrivono i vincoli che il software dovrà rispettare e le funzionalità che dovrà fornire. In questo modo tutte le parti coinvolte hanno una chiara visione di quello che si vuole ottenere.

I requisiti si suddividono in quattro principali categorie: utente, di sistema, funzionali e non funzionali, descritti nei paragrafi successivi.

3.1.1 Utente

Si definiscono requisiti utente (*user requirements*), tutti i servizi che il software dovrà soddisfare per l'utente finale, così definiti:

- Il software dovrà permettere agli utenti di segmentare le immagini mediche attraverso specifiche funzionalità;
- Il software dovrà permettere agli utenti di eseguire l'analisi delle segmentazioni per identificare se una specifica lesione è clinicamente significativa dal punto di vista medico.

3.1.2 Di Sistema

I requisiti di sistema (*system requirements*) permettono di descrivere agli sviluppatori e agli utenti cosa sarà necessario implementare, focalizzandosi su aspetti funzionali e vincoli operazionali. I requisiti individuati sono:

- la ROI dovrà essere formata da un insieme di vertici e di archi che costituiranno gli elementi principali;
- il sistema dovrà fornire le funzionalità per creare, modificare, interagire, copiare, eliminare, esportare, importare e analizzare le ROI. Tutte queste funzionalità dovranno essere organizzate in specifici menù;
- il sistema dovrà mostrare tutte le ROI create in un'apposita tabella, una per ogni riga;
- la creazione di una ROI dovrà avvenire attraverso la visualizzazione di una nuova finestra. Non sarà possibile creare più di una ROI avente le stesse informazioni già presenti nella tabella da un'altra ROI;
- il sistema dovrà distinguere due modalità di interazione per l'utente: quella di *paint mode* in cui sarà possibile disegnare le ROI e quella di *edit mode* in cui sarà possibile interagire con le ROI;
- quando la modalità è impostata su *edit mode*, gli elementi di una ROI dovranno essere selezionabili dall'utente quando questo si posiziona sopra con il cursore;
- le informazioni relative alle segmentazioni presenti in una o più *slices* dovranno essere memorizzate su un singolo file esterno. Il file dovrà essere aggiornato automaticamente ad ogni nuova modifica da parte dell'utente, senza attendere la chiusura del software;
- il sistema non salverà ROI aperte ad ogni nuovo salvataggio nel file;
- ad ogni apertura di uno o più files Dicom, il sistema dovrà caricare i dati relativi alle ROI contenuti nel file esterno, se presente;
- il sistema fornirà una seconda finestra di visualizzazione che mostrerà lo stesso contenuto di quella principale, e permetterà all'utente di orientarsi durante il disegno delle ROI nella finestra principale;
- la visualizzazione dell'output ottenuto dal processo di analisi delle ROI dovrà essere mostrato in una nuova finestra di visualizzazione;
- il processo di analisi delle ROI potrà essere eseguito solo se esiste almeno una segmentazione associata ad una specifica lesione;
- il sistema dovrà permettere all'utente di poter interagire con l'interfaccia grafica (*Graphical User Interface* - GUI) anche quando il processo di analisi delle ROI è in esecuzione.

3.1.3 Funzionali

I requisiti funzionali (*functional requirements*), indicano le funzionalità e i servizi forniti dal software ed indicano come questo deve comportarsi:

- i requisiti funzionali legati alla parte di segmentazione permetteranno di:
  - creare una ROI. Le informazioni che possono essere inserite sono: nome, colore e se tale ROI rappresenta la segmentazione dell'intero organo oggetto di studio;
  - modificare tutte le informazioni inserire durante la creazione di una ROI;
  - disegnare una ROI. L'utente tramite il mouse potrà disegnare la ROI all'interno dell'immagine;
  - interagire con una ROI: l'utente può selezionare, deselezionare, creare e spostare i singoli vertici o l'intera ROI;
  - eliminare un singolo vertice (e di conseguenza i due archi ad esso associati), eliminare una specifica ROI o eliminare le ROI-multiple (più ROI che condividono le stesse informazioni su una specifica *slice* all'interno di uno studio);
  - copiare le ROI dalla *slice* precedente o successiva rispetto a quella considerata di riferimento cioè quella che si sta visualizzando;
  - esportare le ROI in ulteriori formati utili in ambito medico: `roi`, `txt`, `rt-struct` e `xml`;
  - importare le ROI nel software da files in formato `xml` e `rt-struct`, appartenente allo stesso studio medico;
  - gestire le impostazioni. Sarà possibile modificare i valori relativi alla dimensione e saturazione del colore degli elementi che costituiscono una ROI.
- i requisiti funzionali legati alla parte di classificazione, dovranno prevedere un processo di analisi delle ROI, il quale fornirà una previsione che indicherà se la lesione presente è clinicamente significativa, dal punto di vista clinico per il quale è necessaria un'accurata supervisione da parte dei medici. L'output sarà composto da immagini colorimetriche, che evidenzieranno l'area interessata, e dagli *score* di probabilità.

L'immagine 3.1 mostra i casi d'uso che l'utente può compiere all'interno del sistema. In tutti i quei casi in cui è necessaria la presenza di una segmentazione di una ROI è stato definito un *extension point*.

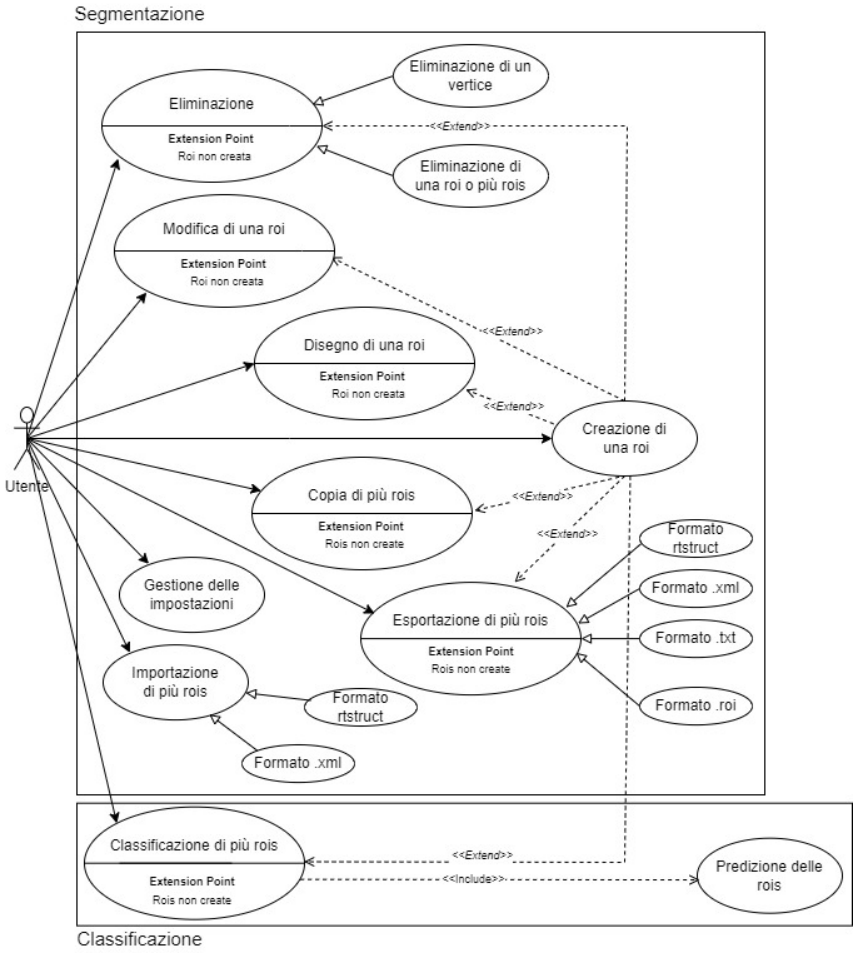


Figura 3.1: Casi d'uso del sistema.

3.1.4 Non Funzionali

I requisiti non funzionali (*non-functional requirements*) sono caratteristiche non richieste dall'utente finale ma proprietà che devono essere soddisfatte per la riuscita dello sviluppo software. In questo caso quelli individuati sono:

- usabilità: il sistema dovrà fornire agli utenti un'interfaccia chiara, semplice, ben organizzata e *responsiveness* in modo da poter utilizzare al meglio tutte le sue funzionalità messe a disposizione e visualizzate;
- scalabilità: il sistema dovrà essere in grado di supportare un elevato numero di studi medici aperti, senza subire una degradazione delle performance.

3.2 Scelta del Software: Aliza Medical Software

La fase successiva ha permesso di individuare un software che potesse essere utilizzato come base di partenza per lo sviluppo delle nuove funzionalità. Quello che si richiedeva era un sistema in grado di poter caricare e visualizzare i files Dicom di cui si disponeva

anche del codice sorgente. La scelta è ricaduta sull'applicativo *open-source* Aliza Medical Software [27] poichè soddisfava entrambi i requisiti.

Questa sezione ha come obbiettivo quello di descrivere il software: in un primo momento, verrà fornita una panoramica introduttiva e successivamente si entrerà nello specifico delle principali funzionalità presenti.

3.2.1 Panoramica

Aliza Medical Software (AlizaMS) si rivolge ad utenti esperti o che lavorano nel campo medico e bio-informatico ed è destinato a scopi di ricerca e sviluppo, didattici e per l'imaging medico. In particolare permette di visualizzare e manipolare immagini 2D, 3D e 4D in diversi formati tra cui: Dicom, MetaIO [47], Nifti [48] e Nrrd [49]. Grazie alle numerose funzionalità offerte, il medico può avere un'ampia panoramica del referto che sta visualizzando, permettendogli di operare e manipolare su di esso al fine di effettuare un'accurata diagnosi del paziente.

AlizaMS offre due versioni: la prima chiamata Aliza Medical Imaging & DICOM Viewer utilizzabile solo previo l'acquisto di una licenza, la seconda chiamata solamente Aliza MS DICOM Viewer utilizzabile gratuitamente. Queste due versioni differiscono principalmente per l'uso di funzionalità che riguardano la manipolazione e l'esportazione/importazione dei dati, utilizzabili solamente con licenza. Nel nostro caso si è scelta la versione gratuita poichè era l'unica di cui si disponeva del codice sorgente su cui poter lavorare, disponibile su *Github* [50].

Il software è disponibile sia per Windows, Linux che per macOS. É ottimizzato per processori multi-core e hardware grafico OpenGL 3. É scritto in linguaggio `C++` e `C` ed è rilasciato sotto la licenza GPL-3.0 [51].

Sulla base della scelta effettuata, vengono descritti i requisiti implementativi che saranno necessari rispettare. Verrà utilizzato `C++` come linguaggio di programmazione mentre le librerie da cui dipenderà il sistema sono: le QT [52] per la gestione dell'interfaccia grafica, le OpenGL [53] utilizzate per la computer grafica 3D e le ITK [54] per la manipolazione delle immagini.

3.2.2 Principali Funzionalità

In questa sezione verranno descritte le principali funzionalità che AlizaMS mette a disposizione nella sua versione base. La figura 3.2 mostra la schermata principale, dopo aver aperto uno studio.

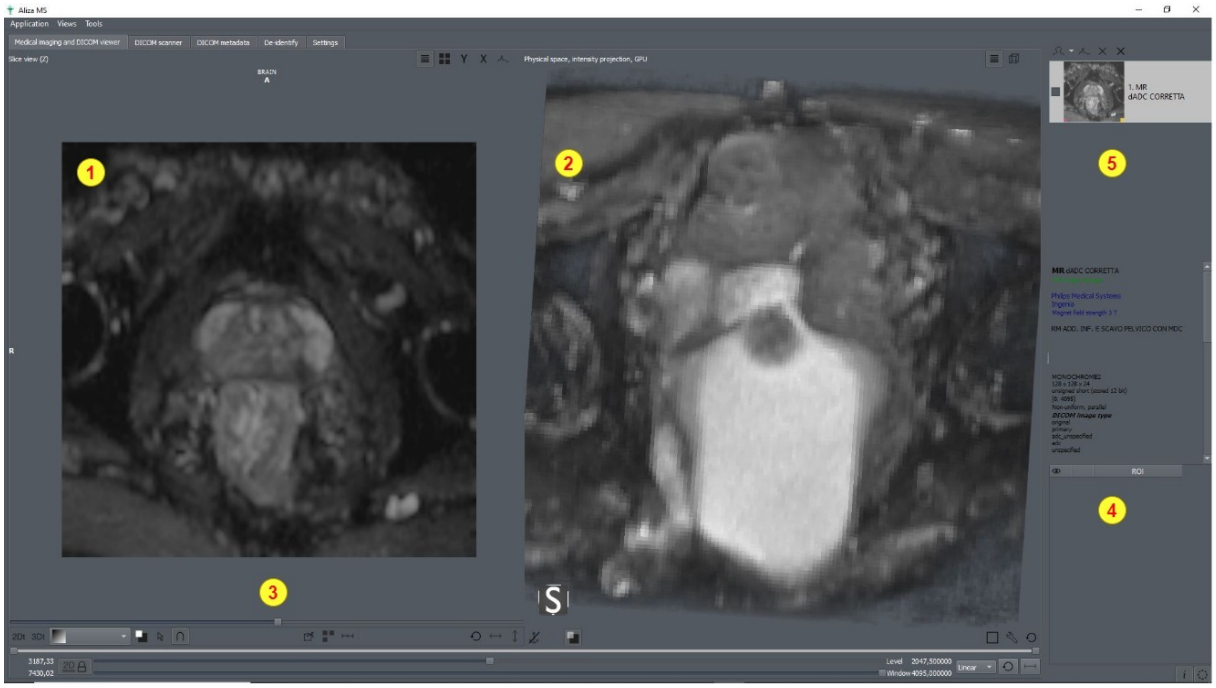


Figura 3.2: Schermata principale di Aliza Medical Software.

In particolare possiamo notare:

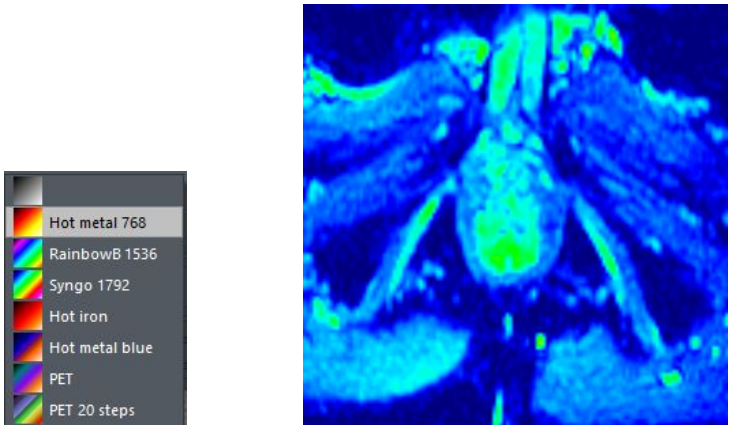
1. l'immagine in modalità vista 2D;
2. il volume renderizzato in modalità vista 3D;
3. la barra di scorrimento orizzontale con cui spostarsi avanti o indietro nella visualizzazione delle immagini 2D e quindi delle *slices*;
4. la tabella delle ROI create;
5. la lista dei files mantenuti aperti. Tramite questa lista è possibile passare da un file all'altro cliccando sull'elemento corrispondente, senza doverlo riaprire nuovamente.

Di seguito vengono descritte le principali funzionalità:

- **Visualizzazione dei files Dicom**
  In AlizaMS è possibile aprire e visualizzare un singolo file Dicom o uno studio completo (composto da più *slices*) attraverso la funzione Dicom-scanner tramite il menù *Application*;
- **Modalità di vista 2D e 3D**
  Ogni studio può essere visualizzato sia in modalità 2D (punto 1 della figura 3.2) che in modalità 3D (punto 2 della figura 3.2). Nel primo caso si può visualizzare un'immagine per volta, avendo però la possibilità di fare un focus su uno specifico asse (`x,y`). Nella vista 3D invece verrà effettuato il rendering del volume 3D dello

studio per visualizzare tutte le *slices*. In quest'ultimo caso l'utente può spostare il volume rispetto ai tre assi. Tramite il menù *Views*, l'utente può scegliere se visualizzare entrambe le modalità (2D e 3D una affiancata all'altra) o visualizzarne una sola.

- **Applicazione delle LUT**
  In entrambe le viste è possibile applicare le LUT (*Look-up Table*). La figura 3.3 mostra a sinistra, le LUT disponibili che possono essere utilizzate nelle immagini di uno studio mentre a destra un esempio dell'applicazione della LUT Syngo 1792 ad un'immagine. In questo modo è possibile esaltare alcune caratteristiche dell'immagine utili ai fini diagnostici, non visibili senza l'applicazione della LUT;



(a) (b)

Figura 3.3: A sinistra la lista delle LUT disponibili. A destra un esempio dell'applicazione della LUT Syngo 1792 ad un'immagine.

- **Visualizzazione delle animazioni**
  Nella modalità 2D è possibile visualizzare le varie *slices* dello studio come se fosse un video;

- **Multi view per immagini 2D**
  In AlizaMS è possibile visualizzare solo una *slice* alla volta per la modalità di vista 2D. La modalità *multi view* consente all'utente di creare una tabella con un numero a piacere di righe e colonne per visualizzare contemporaneamente più *slices* appartenenti allo stesso studio;

- **Calcolo della distanza**
  Una funzionalità molto utile è la possibilità di poter calcolare la distanza (in millimetri) di due punti definiti dall'utente all'interno dell'immagine. In questo modo si può avere un'idea delle dimensioni reali della ROI evidenziata;

- **Calcolo dell'istogramma**
  Dato un singolo file Dicom o uno studio completo è possibile calcolare anche il suo

istogramma. Esso rappresenta la distribuzione dei valori in toni di grigio di tutte le immagini presenti. Tramite i valori di *Size Window* e *Level* (come descritto in 2.1.2) ci si può spostare sull'istogramma per visualizzare specifici livelli di grigio. In questo modo si può individuare la soglia con cui segmentare l'immagine;

- **Selezione di una regione**
  Dato uno studio, è possibile selezionare nella modalità di vista 2D una determinata regione rettangolare (*bounding-box*) così da potersi concentrare maggiormente su quell'area. Tutta la parte dell'immagine nella modalità di vista 3D, eccedente alla *bounding-box* verrà resa invisibile all'utente;

- **Visualizzazione dei metadati**
  É possibile visualizzare i metadati (o *tags*) di ogni specifico studio aperto.

Menù principali

Come mostrato nella figura 3.4, in alto a sinistra sono presenti i menù di AlizaMS. Partendo dal primo, la voce *Application* permette di aprire i singoli file e cartelle che contengono i files Dicom, la voce *Views* permette di gestire le diverse modalità di vista 2D e 3D mentre la voce *Tools* include le utilità per supportare le diverse funzionalità.

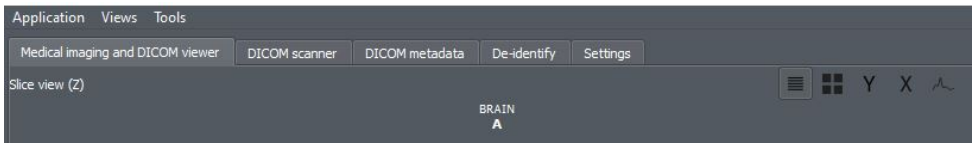
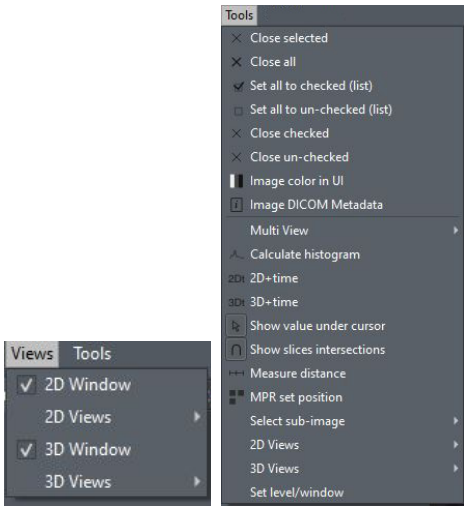


Figura 3.4: Menù principale di Aliza Medical Software.

Sotto al menù *Application*, troviamo cinque sezioni: la prima (*Medical Imaging and DICOM viewer*) consente di visualizzare lo studio aperto, la sezione DICOM scanner permette di gestire l'apertura di uno studio composto da più *slices*. La terza sezione permette di visualizzare i metadati, la sezione *De-identify* permette di gestire le informazioni dei files Dicom mentre l'ultima, *Settings* permette di impostare le preferenze utente. L'immagine 3.5 mostra invece i rispettivi menù per la voce *View* e la voce *Tools*.

Nel primo menù sono presenti due checkbox per abilitare una o entrambe le modalità 2D e 3D mentre nel secondo menù sono presenti le voci di utilità che supportano le principali funzionalità di AlizaMS. Queste ultime possono essere raggruppate in due categorie: nella prima sono presenti le funzionalità per aprire e chiudere i files, settarli su *checked* o *unchecked* e consultare i metadati. Nel secondo gruppo sono presenti le funzionalità per il calcolo dell'istogramma, la visualizzazione delle animazioni, il calcolo della distanza, la selezione di una porzione dell'immagine (*sub-image*) e così via.



(a) (b)

Figura 3.5: Voci del menù *View* e del menù *Tools*.

3.3 Architettura del Sistema

Questa sezione ha come scopo quello di illustrare l'architettura del sistema. L'immagine 3.6 riassume l'architettura ad alto livello, di come i moduli interagiscono tra loro: ogni *box* è un modulo software significativo o dipendente rispetto alle nuove funzionalità da realizzare. Le frecce indicano la direzione di interazione tra i moduli.

A livello puramente funzionale, il modulo:

- *Main Window*, gestisce l'interfaccia grafica relativa alla finestra principale del software e dei diversi elementi che la compongono;
- *Secondary Windows*, gestisce l'interfaccia grafica di tutte le finestre secondarie;
- *Core*, rappresenta la *businnes logic* dell'applicazione. In essa è contenuta tutta la logica relativa alle funzionalità di base e quelle legate alle ROI.
- *2D Scene Management*, aggiorna la scena nella modalità di vista 2D relativa all'immagine e alle ROI;
- *ROI Manager*, si occupa della gestione delle ROI.

L'architettura mostrata nella figura 3.6 può essere estesa attraverso il diagramma delle classi, il quale rappresenta la vista di un sistema software che pone l'accento sulla struttura degli oggetti. In questo caso, è stato necessario selezionare una parte dell'intero diagramma, estraendo la struttura di interesse per le funzionalità da realizzare. La figura 3.7 mostra il diagramma, in cui le frecce indicano un'associazione tra le classi ovvero quando un attributo è un oggetto di un'altra classe. La molteplicità è specificata ai capi delle frecce.

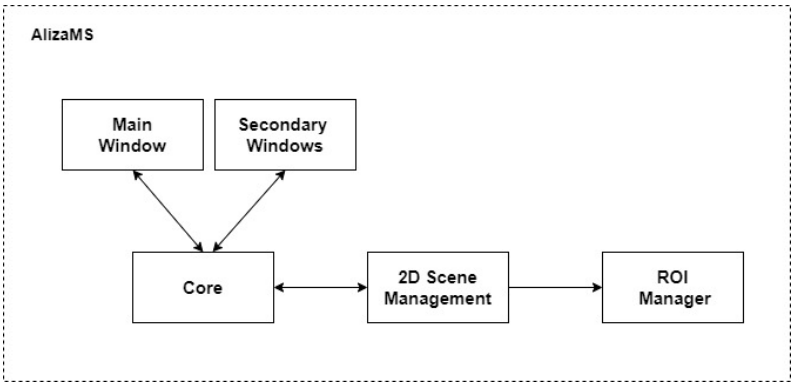


Figura 3.6: Interazione dei moduli all'interno del software AlizaMS.

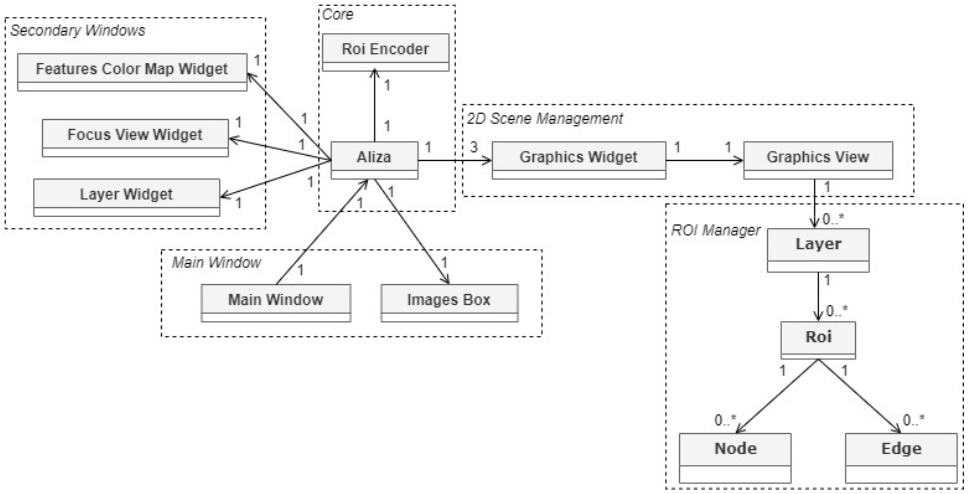


Figura 3.7: Diagramma delle classi.

I *box* tratteggiati indicano i moduli precedentemente descritti.
Le classi presenti, sono così descritte:

- *Main Window*: implementa il layout principale dell'interfaccia grafica del software e dei diversi elementi che la compongono (ad esempio i menù e i pulsanti), intercettando gli input dell'utente;
- *Images Box*: implementa la tabella delle ROI (inserimento ed eliminazione di una riga, modifica delle informazioni presenti di una specifica ROI, ...) e lista dei files aperti;
- *Aliza*: implementa le principali funzionalità di base e quelle relative alle ROI. Implementa il salvataggio sul file esterno ed invia le richieste utente (ricevute tramite la *Main Window*) alla classe di riferimento della specifica funzionalità, comunicando gli aggiornamenti alla GUI;
- *Roi Encoder*: implementa l'esportazione delle ROI nei formati `roi` e `txt`;

- *Graphics Widget*: è un *wrapper* della classe *Graphics View* e gestisce tutti gli aggiornamenti che avvengono nella modalità vista 2D (ad esempio relativo all'immagine e al disegno di una ROI) gestendo anche le animazioni;
- *Graphics View*: permette di aggiornare la scena nella modalità vista 2D relativa a tutte le funzionalità associate alle ROI (ad esempio disegno, selezione, eliminazione e spostamento di un vertice o di un arco);
- le classi *Layer*, *Roi*, *Node* e *Edge* implementato rispettivamente l'oggetto Layer, ROI, vertice e arco. Come detto, ogni ROI è composta da un insieme di vertici e archi ed appartiene ad uno specifico Layer avente caratteristiche comuni (questo aspetto sarà approfondito in seguito nel paragrafo 3.4.1);
- le classi *Focus View Widget*, *Features Color Map Widget* e *Layer Widget* implementano le finestre di visualizzazione secondarie e vengono utilizzate rispettivamente per permettere all'utente di orientarsi durante il disegno di una ROI, visualizzare l'output del processo di analisi e creare un nuovo Layer.

Dopo aver illustrato l'architettura interna di AlizaMS, viene mostrata come è stata progettata l'interazione verso l'esterno e quali dipendenze il sistema presenta. La figura 3.8 mostra appunto quest'ultimo aspetto.

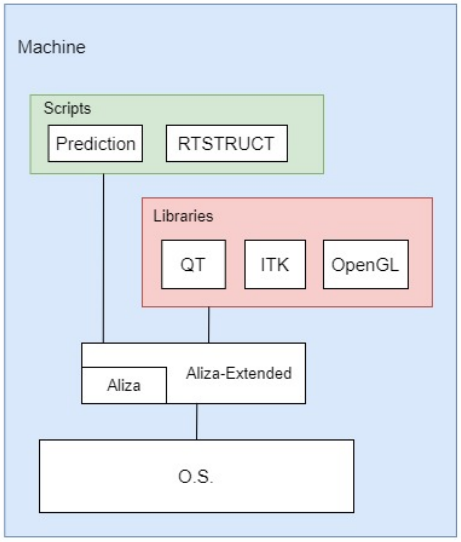


Figura 3.8: Dipendenza e interazione del software AlizaMS con l'esterno.

Sono stati definiti tre principali *box*, ognuno identificato da un colore differente:

- il *box* blu identifica la macchina, intesa come quel dispositivo hardware in grado di poter eseguire il software AlizaMS. All'interno sono presenti due elementi principali:

il primo si riferisce al sistema operativo (*Operating System* - OS) e il secondo al software AlizaMS. Il termine *Aliza* indica il software scelto come base di partenza a cui sono state realizzate le nuove funzionalità (*Aliza-Extended*);

- il *box* rosso mostra le librerie da cui dipende il software;
- il *box* verde mostra gli *scripts* o moduli esterni che sono stati realizzati per poter eseguire particolari funzionalità: il processo di analisi delle ROI (*prediction*) e l'esportazione/importazione delle ROI in formato `rt-struct`.

3.4 Principali Scelte Progettuali

Dopo aver definito i requisiti che il sistema deve avere, identificato il software di partenza e illustrato l'architettura, questa sezione affronta le principali scelte progettuali effettuate, specificandone le motivazioni, pensate sopratutto per la successiva fase di sviluppo e implementazione.

3.4.1 Layer e Roi

Durante la fase di progettazione, è emersa l'esigenza di definire i concetti di Layer e ROI. L'idea è di far si che una o più ROI, aventi caratteristiche in comune (nome, colore e se la ROI rappresenta l'organo di riferimento dell'oggetto di studio) possano appartenere ad un unico livello (o Layer). Supponiamo l'esempio in cui il medico visualizzi un'immagine che presenta diverse aree, distinte e non contigue, che si riferiscono alla stessa lesione. Il corretto approccio è quello di disegnare tante ROI quante sono le aree presenti nell'immagine, tutte dello stesso tipo (o aventi tutte le stesse informazioni). In questo caso è utile creare un Layer e disegnare nell'immagine tante ROI tutte appartenenti sempre allo stesso Layer. L'esempio descritto, ha permesso di illustrare cosa si intende per ROI-multiple: più ROI distinte, disegnate in un unica *slice* e appartenenti ad un Layer comune.

É importante sottolineare che la differenziazione dei due concetti (Layer e ROI) è nata da un'esigenza di sviluppo e di implementazione; per questo motivo l'utente rimarrà estraneo alla distinzione poichè considererà tutto ciò implicito nel sistema.

3.4.2 Persistenza dei Dati

Un altro aspetto che sarà considerato riguarda la persistenza dei dati, legata al salvataggio delle informazioni delle ROI. In questo caso due sono le scelte che possono essere intraprese: salvataggio delle informazioni su un file esterno o tramite la creazione di un database. Gli aspetti emersi, che hanno poi determinato la scelta, sono i seguenti:

- la quantità di informazioni da salvare sono limitate a pochi dati;

- non si necessita di interrogare i dati salvati;
- la complessità e il tempo richiesto per lo sviluppo di un database è maggiore rispetto a quello di un file esterno.

Sulla base di queste considerazioni, si è deciso di realizzare la persistenza tramite la creazione di un file esterno. Viceversa, una problematica emersa scegliendo questa opzione, è data dalla completa perdita dei dati qualora l'utente, accidentalmente cancelli il file di persistenza. Si presume che l'utente utilizzi il software e i files Dicom in maniera responsabile, tale da evitare questa casistica.

Il formato del file scelto è `xml` poichè richiede di definire una struttura interna che permette di organizzare i dati, agevolando la fase di analisi ed estrazione delle informazioni. Al contrario, un file di testo generico (ad esempio `txt`), non richiede necessariamente una struttura ben definita e per questo motivo le informazioni risultano essere più difficili da analizzare.

3.4.3 Interazione Utente

Molta attenzione sarà dedicata all'interfaccia grafica, giustificata dalla tipologia di utente che utilizzerà il software. Questi utenti, non essendo per la maggior parte esperti nel usare un sistema informatico, necessiteranno di una GUI chiara, semplice e facile da usare. Ad esempio nel modellare uno specifico layout, saranno evitati elementi di design secondari che possono confondere l'utente, scegliendo inoltre nomi semplici e di facile comprensione. Durante le progettazione, ad ogni elemento grafico (ad esempio voci che compongono un menù e singoli pulsanti) sarà associata una funzionalità chiara, ben precisa. Un altro elemento rilevante che caratterizzerà la GUI è la notifica all'utente di messaggi di errore, allerta e successo ad ogni cambiamento di stato del software. Dove necessario si cercherà di mantenere la GUI *responsiveness* per fare in modo che l'utente possa continuare ad utilizzare AlizaMS. Un esempio riguarderà l'analisi delle ROI in cui all'avvio del processo (in *background*), la GUI continuerà a rispondere agli input dell'utente.

L'interazione utente con un software non avviene solamente tramite l'uso dell'interfaccia grafica ma anche grazie alla tastiera e al mouse; anche in questo caso, si cercherà di facilitarne l'uso. Nel caso della tastiera, saranno definiti degli *shorcut* da associare a tutte le funzionalità da realizzare, resi ben visibili accanto alle relative voci del menù. In questo caso, l'utente potrà accedere ad una specifica funzionalità non solo cliccando sull'elemento grafico ad esso associato ma anche grazie alla combinazione dei tasti. Un altro esempio che migliorerà l'esperienza utente, sarà dato dall'uso dei tasti freccia (invece dell'utilizzo della barra orizzontale) per spostarsi da una *slice* all'altra all'interno di uno studio. Anche nel caso del mouse, l'obiettivo è di agevolare l'utente con l'utilizzo del software. Ad esempio,

si cercherà di agevolare lo *switch* da una modalità all'altra (*edit mode* ↔ *paint mode*) tramite il click di un particolare tasto (o combinazione di tasti), senza dover ricorrere direttamente all'elemento grafico specifico della GUI.

Per agevolare l'utilizzo di tutte le funzionalità legate alle ROI saranno progettati due menù ad hoc, *Roi Manager* e *Analyze* (un esempio è riportato in figura 3.9), ognuno riguardante un macro aspetto. Il primo, disporrà delle voci per creare, selezionare, importare, esportare ed eliminare una ROI. Inoltre sarà presente anche la possibilità di poter copiare più ROI tra slice consecutive. Il menù *Analyze* sarà dedicato al processo di analisi delle segmentazioni.

Figura 3.9: Menù *Roi Manager* e *Analyze*.

A livello grafico, la progettazione del layout della finestra principale può essere schematizzato come nell'immagine 3.10: al centro sono presenti i *box* per la visualizzazione dell'immagine 2D e del volume 3D, a destra vi è la lista dei files mantenuti aperti e in basso la tabella contenente i Layers. Il *box* tratteggiato in rosso indica la parte del layout in cui saranno inseriti i menù legati alle funzionalità delle ROI mentre il *box* tratteggiato in blu indica la zona del layout in cui saranno aggiunti specifici pulsanti.

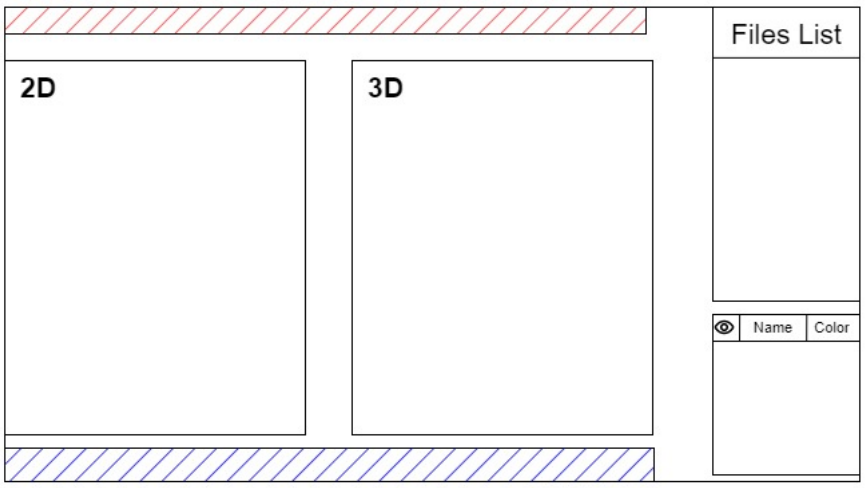


Figura 3.10: Layout dell'interfaccia grafica della finestra principale.

La ROI è stata definita come un poligono che evidenzia un'area all'interno dell'immagine nella modalità vista 2D di AlizaMS. Ogni ROI è stata pensata per essere composta da due elementi principali: un vertice e un arco. Ogni vertice collega sempre e solamente due archi e si definisce ROI quando è composta da un numero minimo di tre vertici ed evidenzia un'area chiusa. L'immagine 3.11 mostra un esempio: gli archi sono rappresentati da segmenti che collegano i vertici della ROI.

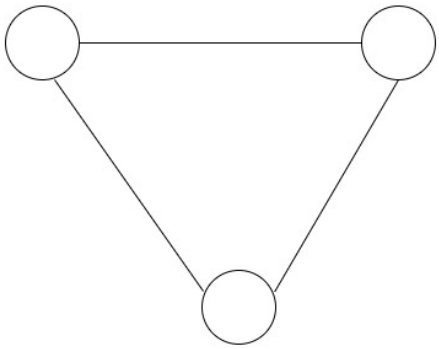


Figura 3.11: Esempio grafico di ROI.

La possibilità di poter selezionare una o più aree, permette al medico di focalizzarsi su specifiche porzioni dell'immagine al fine di identificare, in un secondo momento grazie all'utilizzo di tecniche di ML, eventuali mutazioni e anomalie. In questo modo il medico avrà un secondo parere ovvero quello del software, che fornirà aiuto e ausilio per la diagnosi del paziente. I paragrafi seguenti entreranno nel dettaglio della progettazione dell'interfaccia grafica per le singole funzionalità legate alle ROI.

Creazione e Modifica

Come descritto, le ROI apparterranno sempre ad un Layer, per questo motivo quando si parla di creazione, si farà riferimento a quella di un Layer mentre quando si parla di disegno si farà riferimento alla creazione di una ROI.

Due sono le modalità che sono state progettate per la creazione di un nuovo Layer che potrà avvenire solo dopo aver aperto un file Dicom o uno studio completo. La prima modalità, prevederà il click su una specifica voce presente nel menù *Roi Manager* e la seconda il click nel pulsante caratterizzato dall'icona +, aggiunto nel menù in basso (area blu nella figura 3.10). Indipendentemente dalla scelta effettuata, verrà mostrata una nuova finestra (*Layer Widget*) in cui sarà necessario attribuire un nome, un colore e stabilire se il Layer rappresenterà l'organo oggetto di studio. Il nome predefinito assegnato ad ogni Layer sarà *Lesion_N* con N il numero progressivo di Layers creati fino a quel momento per lo studio corrente. Per la gestione del colore è stata definita una palette di colori predefiniti. Questo

significa che alla creazione del primo Layer di qualsiasi studio aperto, verrà attribuito sempre lo stesso colore di default e così via per quelle successive. Ovviamente all'atto della creazione si potrà scegliere qualsiasi nome e colore ad eccezione che non esisterà già un Layer con le stesse caratteristiche, altrimenti verrà mostrato un messaggio di *warning*. Se il Layer si riferirà all'organo oggetto di studio, in fase di analisi tutte le segmentazioni ad esso associate, verranno considerate come *ground truth*. Una volta completato il widget, sarà possibile creare il Layer e questo verrà aggiunto alla tabella. La riga associata al Layer appena creato verrà selezionata.

Lo scenario appena descritto è stato modellato attraverso un diagramma sequenziale nell'immagine 3.12.

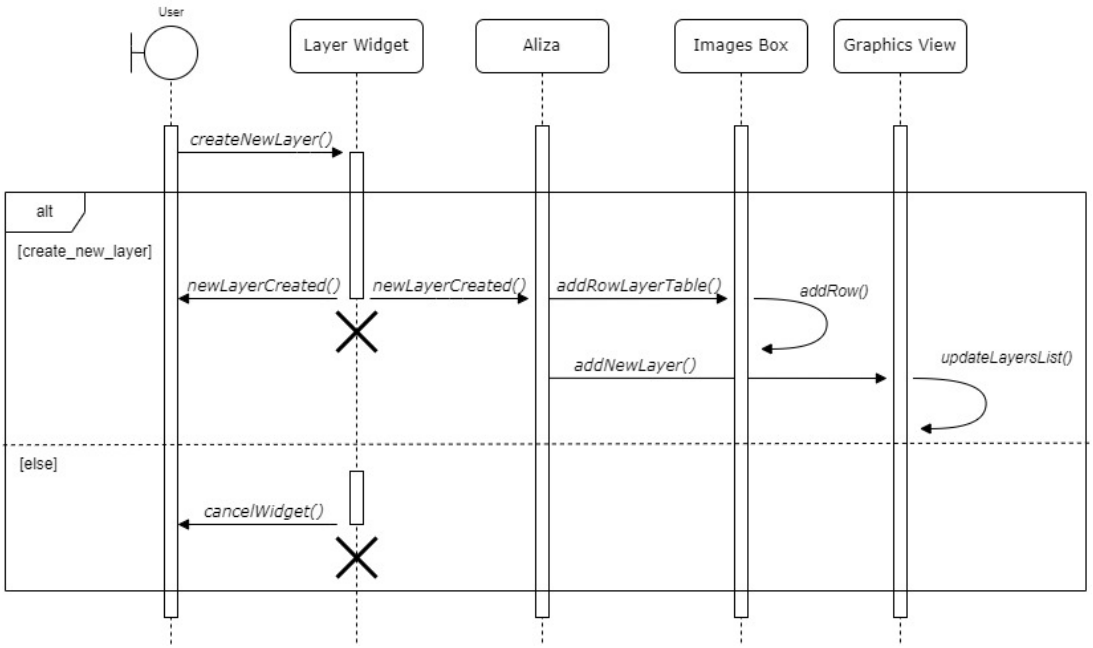


Figura 3.12: Diagramma sequenziale per la creazione di un nuovo Layer.

L'utente si interfaccerà con il *Layer Widget*, pensato appositamente per questo scopo. In caso di creazione del Layer, la classe *Aliza* comunicherà alla classe *Images Box* di aggiungere una nuova riga nella tabella e alla classe *Graphics View* l'avvenuta creazione di un nuovo Layer. Se l'utente annullerà la creazione, il widget sarà chiuso e nessun Layer verrà creato.

La tabella contenente i Layers è stata pensata per contenere la quantità minima di informazione e per consentire all'utente di identificare ogni Layer creato. Tre sono le colonne definite: un *checkbox* per rendere visibile o invisibile tutte le ROI associate ad uno specifico Layer presenti nell'immagine, la colonna contenente il nome e quella relativa alla visualizzazione del colore. La figura 3.13 mostra la modellazione appena descritta.

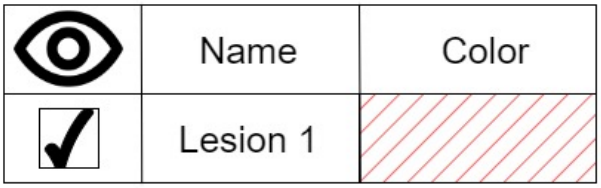


Figura 3.13: Modellazione delle colonne della tabella delle ROI.

Per gestire tutte le funzionalità associate ad un Layer creato, è stato pensato di aggiungere un menù che sarà possibile visualizzare cliccando con il tasto destro del mouse, nella riga della tabella corrispondente. In questo modo, qualsiasi sia la funzionalità, questa avrà effetto solo sul Layer selezionato e su tutte le ROI ad esso associate. Nel menù saranno presenti le voci per modificare ed eliminare un Layer. In caso di modifica, verrà visualizzato lo stesso widget mostrato in fase di creazione, in cui sarà possibile modificare tutte le informazioni. L'eliminazione verrà trattata nei paragrafi successivi.

Disegno e Interazione

Il disegno di una ROI è stato progettato nella seguente modalità: l'utente seleziona la riga nella tabella, associata al Layer che vuole disegnare e procede disegnando la ROI nella finestra di visualizzazione 2D. Selezionando il Layer nella tabella, verrà abilitata la cosiddetta *paint mode*. Per far emergere questa modalità a livello visivo, si è pensato di cambiare icona del cursore in e di mostrare un quadrato, di colore verde brillante, che caratterizza la finestra di visualizzazione 2D. Tale quadrato non verrà invece mostrato quando la modalità sarà su *edit mode* in cui, in questo caso, sarà possibile sono l'interazione con le ROI. Il disegno di uno o più vertici sarà possibile cliccando solamente all'interno della finestra con il tasto sinistro del mouse. Man mano che i vertici verranno creati, verranno disegnati anche gli archi corrispondenti. Per agevolare l'utente nella segmentazione della ROI e determinare il prossimo vertice da creare, verrà disegnato un arco che seguirà il cursore man mano che si procederà con la costruzione della ROI. In questo modo si faciliterà maggiormente la fase di segmentazione senza che l'utente vedrà l'arco disegnato solo dopo aver creato un ulteriore vertice che lo collegherà. La figura 3.14 schematizza quanto descritto.

In caso di un eventuale errore, è stata progettata la possibilità di annullare solo l'ultimo vertice disegnato e non in maniera sequenziale e ricorsiva tutti i vertici delle ROI presenti. Per chiudere una ROI ed evidenziare un'area chiusa, sarà necessario cliccare con il tasto sinistro del mouse sul primo vertice creato, evidenziato per identificarlo tra tutti i vertici presenti, se passato sopra con il cursore. La chiusura di una ROI con un numero di vertici inferiore a tre, eliminerà automaticamente la ROI.

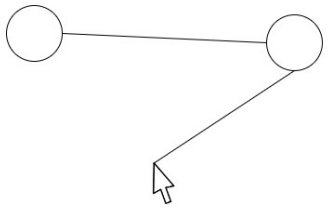


Figura 3.14: Esempio grafico nel disegnare una ROI.

L'immagine 3.15 mostra lo scenario di creazione di una ROI avente tre vertici e tre archi.

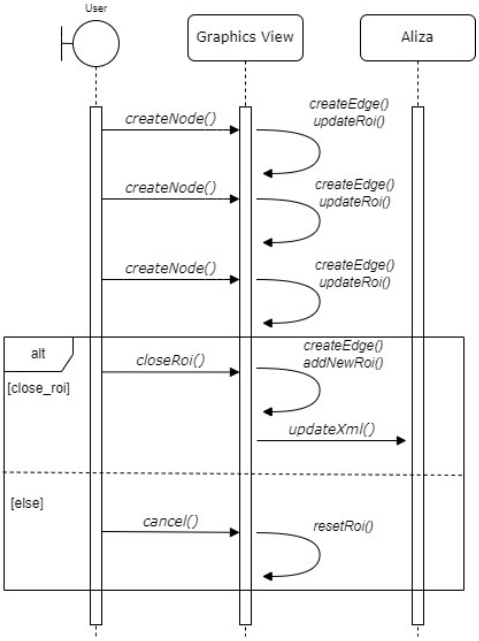


Figura 3.15: Diagramma sequenziale per la creazione di una nuova ROI.

La classe *Graphics View* intercetterà tutti gli eventi associati al mouse. Ad ogni nuovo click del mouse verrà creato il vertice nella posizione corrente del cursore e l'arco ad esso associato. La chiusura della ROI avverrà creando una nuova istanza della classe *Roi*, collegando l'ultimo e il primo vertice tramite un arco. La funzione *updateXml* aggiornerà il *flag* relativo al salvataggio delle nuove informazioni sul file esterno. Se il *flag* sarà a *True* allora saranno presenti nuove informazioni da salvare. In caso di annullamento, tutti i vertici e gli archi creati verranno cancellati.

Per facilitare la segmentazione, e quindi il disegno della ROI, si è pensato utile realizzare una seconda finestra chiamata *Focus View*. Il vantaggio di utilizzare questa finestra sarà il seguente: quando si segmenteranno piccole ROI ad elevato zoom, l'utente perderà

la cognizione spaziale, non riuscirà a capire in quale specifica area dell'immagine si sta trovando. Per ovviare a questo problema, senza dover aumentare e diminuire continuamente lo zoom, l'utente potrà visualizzare la propria posizione tramite una seconda finestra: la *Focus View*. Quest'ultima mostrerà la stessa immagine di quella principale, incluse le ROI disegnate, ma ad uno zoom ridotto. Inoltre, mentre la *paint mode* sarà abilitata verrà mostrato un cerchio che indica la posizione corrente del cursore con lo stesso colore della ROI selezionata. In sintesi, la *Focus View* è stata pensata per rappresentare una vista dall'alto dell'immagine mentre quella principale una vista del particolare e dettaglio. Da precisare che gli zoom delle due viste saranno indipendenti e quindi facilmente personalizzabili a seconda delle esigenze. Sarà invece sincronizzato lo spostamento dell'immagine: ad esempio se in quella principale si effettuerà uno spostamento di una certa quantità verso destra, lo stesso avverrà nella *Focus View* tenendo conto della differenza di zoom presente, non sarà possibile invece il viceversa. Per aprire la *Focus View* sarà necessario selezionare l'apposita voce dal menù *Roi Manager* o cliccare il relativo pulsante, aggiunto a fianco del pulsante di creazione di una ROI, nel menù in basso.

Dopo aver disegnato una o più ROI nella finestra 2D, sarà possibile interagire con esse solo se la modalità sarà impostata in *edit mode*. Le interazioni che sono state progettate sono molteplici e descritte come seguenti:

1. muovendo il mouse sopra alla ROI, i singoli vertici o archi si selezioneranno/deselezioneranno, a seconda della posizione del cursore. Per una migliore esperienza utente si è deciso di cambiare la forma dei vertici selezionati della ROI (da cerchio a quadrato) e il colore con quello complementare a quello utilizzato per disegnarla (sia per i vertici che per gli archi). Si definisce colore complementare, il colore che si trova al lato opposto del cerchio cromatico (utilizzato per classificare le sfumature del colore) rispetto a quello preso come riferimento. Nella *edit mode*, l'icona del mouse assumerà questa figura in cui sarà possibile spostarsi nell'immagine con il classico *Drag and Drop*. Quando invece si selezionerà un vertice o un arco allora l'icona del mouse assumerà questo simbolo . Dopo che un vertice o un arco sarà selezionato, è possibile interagire con esso.

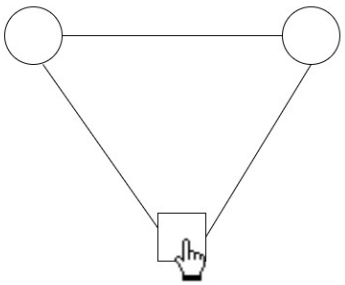


Figura 3.16: Esempio grafico di selezione di un vertice.

2. con un vertice selezionato, sarà possibile spostarlo tenendolo premuto, modificando così la ROI. Questo implica che anche gli archi ad esso collegati si muoveranno in accordo con la nuova posizione. Inoltre, si è pensato di fissare le selezione solamente di un vertice alla volta, in modo tale che questo rimanga selezionato anche quando il cursore continuerà a muoversi, bloccando di fatto la selezione degli altri elementi della ROI. In questo modo sarà possibile eliminare quel vertice (si rimanda al paragrafo 3.4.3). Per sbloccare la selezione sarà necessario cliccare su un qualsiasi altro punto all'interno dell'immagine.

3. con un arco selezionato, sarà possibile creare un nuovo vertice. Cliccando con il tasto sinistro su un qualsiasi punto dell'arco selezionato, verrà creato il nuovo vertice e i due nuovi archi. Quest'ultimi si collegheranno al vertice precedente e successivo rispetto a quello creato. Il vecchio arco invece verrà eliminato.

4. oltre a poter fissare la selezione di un vertice, si è pensato utile fissare anche la selezione dell'intera ROI (ad esempio per eliminarla o spostarla interamente in un secondo momento). In questo caso, la modalità che si è ritenuta, a livello utente, più intuitiva è stata quello di selezionare la ROI quando il cursore si trova all'interno di essa. La selezione si bloccherà cliccando con il tasto sinistro all'interno della ROI, oppure fissando la selezione di un vertice e utilizzando la specifica voce *Select All Roi* dal menù *Roi Manager*. Inoltre, per facilitare la selezione di un'altra ROI quando una di queste è già selezionata, senza dover deselezionare quella corrente e selezionare quella nuova, quest'ultima sarà possibile selezionarla cliccandoci al suo interno. In questo modo la prima verrà automaticamente deselezionata.

5. un'altra interessante funzionalità che è stata progetta è quella di poter spostare spazialmente nell'immagine l'intera ROI, utile quando è necessario muoverla di qualche pixel senza intervenire nello spostamento di ogni singolo vertice. Fissata la selezione, spostando il cursore all'interno della ROI e tenendo premuto il tasto sinistro, questa inizierà a muoversi. Anche in questo caso, si è deciso di cambiare icona del cursore in ✣ per enfatizzare l'azione dell'utente. Deselezionando la ROI, la modalità impostata sarà di nuovo quella di *edit mode*. Come per il disegno, si è progettata la possibilità di annullare lo spostamento tramite la combinazione *CTRL + Z* quando la ROI è ancora selezionata, permettendo di ritornare fino alla posizione iniziale.

Per facilitare il passaggio dalla *edit mode* alla *paint mode* e viceversa, oltre al click del pulsante con il simbolo (già presente nella versione base) nel menù in basso alla finestra di visualizzazione 2D, è stato pensato di aggiungere un'altra modalità ancora più rapida: cliccare direttamente il tasto destro del mouse.

Eliminazione

Per questa funzionalità si è pensato utile non solo permettere l'eliminazione di una singola ROI ma anche delle sue componenti, in modo tale da poterla modificare anche dopo la creazione. Le tre modalità che sono state progettate sono: la cancellazione di un vertice, di una singola ROI o di ROI multiple. Ogni qualvolta si effettuerà un'operazione di cancellazione non sarà più possibile annullare l'azione e quindi ripristinare i dati persi. La tre modalità sono state modellate nel seguente modo:

- **Eliminazione di un vertice**
  Dopo aver fissato la selezione di un vertice (si rimanda al punto 2 del paragrafo 3.4.3) sarà possibile eliminarlo tramite la voce dal menù *Roi Manager* o *shorcut* da tastiera. Un messaggio di conferma sarà previsto per effettuare la cancellazione. A livello grafico, l'immagine 3.17 mostra quello che accadrà: verrà eliminato il vertice e gli archi ad esso associati (quelli contrassegnati con la x). Un nuovo arco verrà creato (linea tratteggiata) che collegherà il vertice precedente e successivo a quello eliminato. Se la ROI è composta da tre vertici e uno di questi verrà eliminato, automaticamente l'intera ROI verrà cancellata.

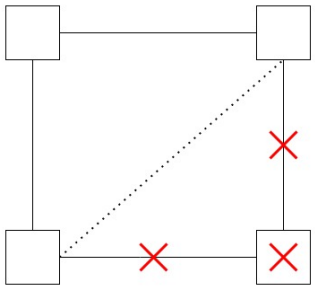


Figura 3.17: Esempio grafico di eliminazione di un vertice.

- **Eliminazione di una singola ROI**
  In questo caso la modalità di eliminazione sarà analoga a quella del vertice ma riguarderà l'intera ROI, in cui sarà necessario fissare la selezione (si rimanda al punto 4 del paragrafo 3.4.3).

- **Eliminazione di ROI Multiple**
  Inoltre sarà anche possibile cancellare tutte le ROI appartenenti ad un unico Layer, direttamente dalla tabella. In questo caso la funzionalità riguarderà tutte le ROI presenti nello studio aperto per tutte le *slices*. Essendo un'azione importante, dal punto di vista della perdita dei dati, si è pensato di mostrare un doppio messaggio di conferma prima di completare l'operazione, al fine di evitare cancellazioni accidentali. L'eliminazione avrà successo selezionando la specifica voce dal menù, visualizzabile tramite il tasto destro del mouse, sul Layer da eliminare.

Copia delle ROI

Segmentando più *slices* appartenenti ad uno studio completo, molte di esse si differenziano per piccoli particolari: l'area da segmentare potrebbe variare di pochi pixels tra una *slice* e l'altra rendendo monotona la segmentazione dall'inizio della stessa lesione. Per questo motivo è nata l'esigenza di poter permettere all'utente di poter copiare le ROI create di una specifica *slice* in quella precedente e/o successiva rispetto a quella considerata come rifermento. Questa funzionalità potrà essere molto utile se combinata con quella descritta al punto 5 del paragrafo 3.4.3, in cui dopo aver copiato le ROI, sarà possibile spostarle spazialmente nell'immagine. In questo modo non sarà più necessario ricreare tutte le ROI ma copiarle e successivamente modificarle a seconda della regione da segmentare. Tale funzionalità sarà accessibile dal menù *Roi Manager* in cui, per poter copiare le ROI, sarà sufficiente spostarsi nella slice di destinazione.

Durante la progettazione è fondamentale individuare tutti gli scenari possibili che possono avvenire durante l'utilizzo di AlizaMS da parte dell'utente. In questa fase, è emerso il caso in cui nella *slice* di destinazione potrebbero essere già presenti delle ROI. In questo caso le ROI presenti verranno sovrascritte da quelle nuove. Per evitare di perdere i dati, verrà mostrato un messaggio di *warning* che comunicherà all'utente che le ROI correnti verranno sovrascritte con quelle della *slice* di origine scelta.

Esportazione e Importazione dei Dati

Un aspetto in cui si è attribuito molta attenzione riguarda l'esportazione ed importazione delle ROI, dato sopratutto dal numero di formati richiesti, per favorire l'interoperabilità dei dati con l'utilizzo di altri programmi.

Per l'esportazione, quattro saranno i formati che potranno essere utilizzati: `roi`, `txt`, `rt-struct` e `xml`. Nel primo caso, si tratta di un formato proprietario del software ImageJ [24]: in questo modo è possibile lavorare sugli stessi files, importando le ROI nel software poichè offre diverse funzionalità non presenti in AlizaMS. Il formato `txt` invece è il comune formato per i file di testo mentre `rt-struct` è il formato che permette di salvare le ROI all'interno dei files Dicom, aggiungendo e modificando specifici *tags*.

Per poter effettuare l'esportazione, in primo luogo dovrà essere presente almeno una segmentazione, dopodiché sarà necessario selezionare la voce *Export Roi* dal menù *Roi Manager* e scegliere il tipo di formato.

Per l'esportazione nel formato `roi` è stata prevista la creazione di una cartella `zip` contenente tante sotto-cartelle quanti sono i diversi Layers presenti nella tabella che hanno delle segmentazioni da esportare. Ognuna di queste sotto-cartelle conterrà tutti i files

`roi`, ognuno associato ad una differente *slice*. La sintassi scelta per questo tipo di file è `LayerName_slice_filename.roi`, contenente: il nome assegnato al Layer seguito dal numero della *slice* e dal nome del file Dicom a cui fa riferimento. Nel caso in cui sono presenti ROI multiple, sarà eseguita una combinazione per poter memorizzare più ROI sul file corrispondente e visualizzarle successivamente in ImageJ.

Nel secondo caso ovvero per il formato `txt`, verrà creata una cartella contenente al suo interno tanti files di testo quanti sono i Layers. In ognuno di questi file saranno contenute le informazioni riguardanti le coordinate delle ROI per l'intero studio o per il singolo file.

Il terzo caso è il formato `rt-struct` (*Radiotherapy Structure Set*) che permette di salvare le ROI all'interno di nuovi files Dicom, modificando specifici *tags* definiti dallo standard. I *tags* relativi al `rt-struct` sono tutti quelli che iniziano per (0036, XXXX). Dopo aver effettuato l'esportazione, ogni volta che si aprirà un file Dicom si potranno visualizzare le ROI create con qualsiasi software in grado di visualizzare le segmentazioni in `rt-struct`. In AlizaMS le ROI in formato `rt-struct` sono visibili sia in modalità di vista 2D che in modalità di vista 3D. Molto utile è quest'ultima funzione poichè permette di avere una panoramica di tutte le ROI create sull'intero volume. Le informazioni relative alle ROI che verranno salvate in `rt-struct` saranno: il nome, il colore e le coordinate di ogni nodo.

L'importazione delle ROI invece sarà possibile eseguirla solamente con files in formato `xml` e `rt-struct`, poichè si è voluto valorizzare maggiormente la parte di esportazione. L'importazione avrà successo solo se, dopo aver selezionato la relativa voce dal menù *Roi Manager*, i files importati hanno lo stesso UUID e lo stesso numero di *slices* rispetto allo studio aperto.

Persistenza dei dati

La persistenza dei dati relative alle ROI, è stata progettata attraverso la creazione di un file `xml`. Il salvataggio dei dati sarà automatico e completamente trasparente all'utente. Se si aprirà un singolo file Dicom, verrà creato il suo corrispondente file `xml` avente lo stesso nome, salvato nello stesso percorso del *file system* in cui sarà presente il file Dicom. Se si apre uno studio completo, verrà creato un file `xml` che si riferirà a tutte le *slices* che compongono quello studio. In questo caso, il nome attribuito al file `xml` sarà quello della directory che conterrà i files e verrà salvato nello stesso percorso del *file system* in cui sarà presente tale directory.

Gestione delle Roi aperte

Come detto, una ROI si definisce tale quando questa evidenzia un'area chiusa ed è

composta da un minimo di tre vertici. É stato previsto che durante il disegno di una ROI è necessario muoversi nell'immagine o modificare altre ROI, prima di chiudere quella che si stava disegnando. Tutto ciò, prevede la definizione di un insieme di casistiche in cui le ROI non chiuse verranno eliminate. Alcuni di questi scenari sono: chiusura del file, passaggio da un file all'altro tra quelli mantenuti aperti nella lista (punto 5 della figura 3.2), spostamento in avanti o indietro di una *slice* tramite la barra di scorrimento o tasti freccia, click nella tabella su un Layer diverso da quello selezionato e creazione di un nuovo Layer. Se invece l'utente volontariamente ha intenzione di eliminare una ROI aperta, uno specifico tasto della tastiera verrà definito per eseguire tale azione.

Processo di Diagnosi

Il menù *Analyze* offrirà invece le funzionalità per analizzare le segmentazioni effettuate. Due sono le voci che lo comporranno: *Analyze Roi* e *Features Color Map View*. La prima permetterà di eseguire il processo di diagnosi. All'avvio, verrà mostrata una nuova finestra che indicherà all'utente che il processo è stato avviato, con la possibilità di terminarlo anticipatamente. Al termine del processo, verrà notificato il completamento o fallimento dell'analisi. La seconda voce permetterà invece di visualizzare i risultati. Quest'ultimi verranno salvati all'interno di una nuova directory avente come suffisso `_roianalyze`, nella stessa posizione del *file system* in cui si troveranno i files Dicom. Cliccando sulla voce *Features Color Map View* verrà mostrata una nuova finestra chiamata *Color Map View* in cui sarà possibile consultare le mappe colorimetriche e gli *score* delle classi. Cambiando slice, se saranno presenti delle segmentazioni, la *Color Map View* mostrerà la relativa mappa colorimetrica altrimenti l'immagine originale. L'immagine 3.18 mostra il diagramma sequenziale del processo di diagnosi.

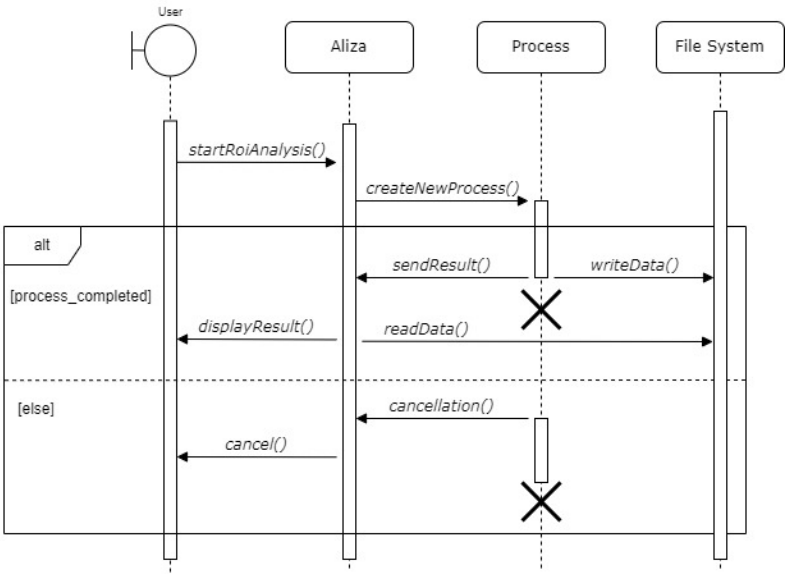


Figura 3.18: Diagramma sequenziale per la fase di analisi delle ROI.

All'avvio del processo da parte dell'utente, *Aliza* creerà un processo di sistema che chiamerà

il modulo esterno dedicato all'analisi delle ROI. La scelta di eseguire un processo di sistema permetterà di avere più controllo su di esso, monitorando anche il suo stato; in caso di *crash* è possibile intercettare l'errore. In caso di successo, il *task* salverà l'output nel *file system* e comunicherà i risultati ad *Aliza*. In caso di errore verrà mostrato un messaggio di notifica all'utente.

Gestione delle Impostazioni

Il software AlizaMS nella sua versione di base, mette a disposizione una sezione *Settings* per settare le impostazioni e permettere di scegliere le preferenze utente. Tra queste si è progettato di aggiungere quelle riguardanti la dimensione e il colore delle ROI. In particolare l'utente avrà la possibilità di personalizzare, per i vertici e per gli archi:

- la dimensione, scegliendo un numero compreso tra 1 (piccolo) e 5 (grande). In questo modo si ha la possibilità di segmentare anche porzioni di immagini molto piccole (soprattutto quando si lavora con zoom elevati) in modo tale che la dimensione del vertice o dell'arco non sovrasti quella della porzione dell'immagine;
- la saturazione del colore, scegliendo un numero compreso tra 30 (trasparenza minima) e 255 (senza trasparenza). Questa opzione è molto utile poiché si fornirà la possibilità di poter vedere il contenuto dell'immagine sotto la ROI che si sta disegnando, per effettuare una segmentazione ottimale.

A fianco di ogni parametro che è possibile modificare, verrà inserito un pulsante che se premuto imposterà il parametro al valore predefinito. Ogni volta che l'utente modificherà le impostazioni delle ROI, tali valori verranno memorizzati internamente al software. Ad ogni nuova riapertura di AlizaMS, i valori dei parametri saranno aggiornati con l'ultima modifica effettuata.

3.4.4 Moduli Esterni

Come descritto nel paragrafo 3.3, per modulo esterno si intende una componente software (o *script*) che non è parte integrante del software AlizaMS, ma è richiamata da quest'ultimo per compiere specifiche funzionalità: la predizione diagnostica della ROI e l'esportazione/importazione dei dati in formato `rt-struct`. Nel primo caso, la scelta è determinata dall'esigenza di mantenere separata la parte grafica di interazione con l'utente con la parte di predizione, tramite modelli di intelligenza artificiale. Nel caso del formato `rt-struct`, lo sviluppo di un modulo esterno è data dall'assenza di una libreria in linguaggio `C++`, *cross platform*, pronta all'uso in grado di leggere e manipolare i valori dei *tags* Dicom.

3.4.5 Classificazione

L'obiettivo di questa funzionalità è la possibilità di determinare, a partire dall'utilizzo di un software, se una lesione (in questo caso il tumore prostatico) è clinicamente significativa (*Clinically Significant PCa* ovvero un *Gleason Score* pari o superiore a 7B) o non clinicamente significativa (*Not Clinically Significant PCa*, *Gleason Score* uguale o inferiore a 7A) dal punto di vista medico. In particolare, il primo caso indica la presenza di un tumore ad alto rischio e particolarmente aggressivo mentre il secondo indica un tumore indolente, a lenta espansione. Nell'ambito della visione artificiale questo *task* prende il nome di classificazione binaria in cui, per perseguire l'obiettivo, è necessario addestrare un modello di ML o di DL.

Il *task* di classificazione è già stato realizzato e proviene da una piccola parte di un lavoro pregresso realizzato dal CVG [55] (*Computer Vision Group*). Questo lavoro [56] ha permesso di generare, estrarre e selezionare *features* di interesse altamente significative che hanno consentito, tramite l'addestramento di un modello di intelligenza artificiale, di classificare le lesioni prostatiche come descritto in precedenza.

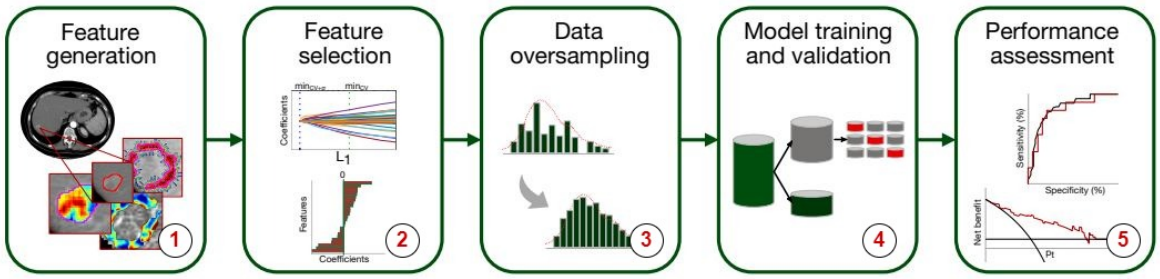


Figura 3.19: Pipeline dello sviluppo del modello di classificazione del CVG.

La figura 3.19 mostra la *pipeline* del lavoro del CVG, composta da 5 principali fasi:

- *Feature Generation*: un insieme di *features* sono state generate a partire dall'analisi delle immagini biomedicali. Per ogni pixel appartenente ad una ROI di una specifica *slice*, è stata utilizzata una finestra di dimensione 9 x 9 per calcolare un'insieme di 12 *features* locali (tra cui la media, la mediana, la *skewness*, la *kurtosis*, l'*interquartile range*, il coefficiente di variazione e l'entropia) che riassumono le caratteristiche salienti di un intorno del pixel considerato. Sulla base del calcolo di ogni *feature* locale, considerando tutti i pixel appartenenti alla ROI, è possibile ottenere una mappa colorimetrica che rappresenta la distribuzione di ogni *feature* locale associata ad una specifica *slice*. Ogni mappa colorimetrica è caratterizzata da un'area colorata in cui il colore rosso (blu) indica il valore massimo (minimo), reale o normalizzato, che una specifica *feature* può assumere. Una barra colorimetrica è associata ad ogni mappa, per individuare tutte le sfumature di colore presenti. L'interpretazione che può essere definita sulla

base del colore nella mappa è la seguente: il rosso indica un'alta presenza di una specifica *feature* mentre il colore blu indica una bassa presenza, in accordo con la barra colorimetrica per le relative sfumature di colore.

A partire dalla distribuzione di ogni *feature* rappresentativa dell'intero volume dello studio, sono stati calcolati 11 descrittori globali considerando altrettante *features*, ottenendo 132 combinazioni;

- *Feature Selection*: a partire dai descrittori globali ottenuti nella fase precedente, sono state individuate le quattro migliori combinazioni dal punto di vista clinico. Esse sono quelle più significative, in grado di separare e predire al meglio le classi del problema.

  Le quattro migliori *features* sono state selezionate, dopo una normalizzazione e standardizzazione dei dati. La normalizzazione è stata effettuata mediante l'aumento del contrasto utilizzando la scala lineare di ogni singolo istogramma della *feature*. Le quattro *features* selezionate sono: la mediana del coefficiente di variazione, l'uniformità della media, la *skewness* della *skewness* e l'*interquartile range* della *standard deviation*. Le *features* ottenute saranno successivamente utilizzate per la fase di addestramento e predizione del classificatore;

- *Dataset Oversampling*: per aumentare la significatività statistica dei sottoinsiemi coinvolti in tutte le fasi di sviluppo del classificatore è stato eseguito un *oversampling*;

- *Model Training and Validation*: addestramento di un classificatore binario (SVM) sulla base delle *features* selezionate;

- *Performance Assessment*: calcolo delle metriche e valutazione delle performance sul modello addestrato.

Nel contesto di questo elaborato, l'obiettivo è quello di effettuare il *porting* a partire dall'addestramento del modello (fase 4 - *Model Training and Validation*) ovvero far si che tale implementazione, realizzata in `Matlab`, non sia più dipendente da questo ambiente ma che venga realizzata con un altro linguaggio di programmazione facilmente integrabile in AlizaMS.

Nel *dataset* messo a disposizione dal CVG [55], sono presenti 89 pazienti con cancro alla prostata. Viene considerata come *True Positive* (TP) la classe di lesioni *Clinically Significant PCa* e come *True Negative* (TN) la classe *Not Clinically Significant PCa*. Il dataset contiene in totale 117 lesioni, di cui 87 (42 TP e 45 TN) utilizzate per l'addestramento e 30 (14 TP e 16 TN) per il test. Ogni lesione è rappresentata da un vettore a 4 dimensioni, in cui ogni dimensione corrisponde ad una specifica *features* selezionata. Il kernel utilizzato

nell'addestramento sarà di tipo lineare, data anche la ridotta dimensione del *dataset*, e per controllare l'*overfitting* verrà impiegata la *K-Fold Cross Validation*.

La fase di addestramento è stata progettata come seguente: verranno effettuate N iterazioni di *K-Fold Cross Validation*. Ad ogni iterazione, verrà utilizzato un numero casuale per suddividere il *Training Set* in K partizioni. Per ogni modello addestrato verrà calcolata la ROC e la corrispondente AUC. Pertanto i modelli affetti da *overfitting* produrranno una AUC sul *Validation Set* maggiore rispetto a quella relativa al *Training Set*. In quest'ultimo caso tali modelli verranno scartati. Inoltre, poichè per ogni iterazione si addestrano K modelli, solo uno di questi modelli SVM verrà mantenuto: quello con l'AUC più alta sul *Validation Set*. Successivamente i modelli selezionati, al massimo N, verranno ri-addestrati sull'intero *Training Set*. Il modello migliore selezionato sarà quello con il maggior valore di *Informedness* (BM) ottenuta come $TPR + TNR - 1$ e AUC sul *Training Set*, in cui verrà attribuita maggiore priorità alla BM. Infine verranno calcolate le performance utilizzando il modello migliore sul *Test Set* (fase 5 dell'immagine 3.19).

4 Implementazione

In questo capitolo verranno illustrate come le scelte progettuali, descritte nel Capitolo 3, sono state implementate. In particolare verrà descritta come si è condotta la fase di sviluppo per l'intero lavoro di tesi, nonché la modellazione delle classi e le scelte implementative più rilevanti.

4.1 Organizzazione del Processo di Sviluppo

Trattandosi di un progetto multidisciplinare (informatica e medicina), la fase di progettazione è iniziata con la partecipazione di tutte le figure coinvolte delle varie discipline. L'esperto del dominio in ambito medico è Margherita Mottola, bioingegnere presso il DIMES [57] (Dipartimento di Medicina Specialistica, Diagnostica e Sperimentale) e membro del CVG [55] (*Computer Vision Group*). La presenza di questa figura ha permesso di comprendere i concetti fondamentali, di definire le funzionalità richieste e come queste dovranno essere utilizzate in futuro dai medici, evitando così il rischio di produrre un software di difficile utilizzo. Infatti, gli informatici avendo una propria *forma mentis*, tendono a realizzare programmi software che si avvicinano più al loro modo di pensare ma che potrebbero risultare di difficile comprensione e usabilità per coloro che non fanno parte di questo ambito. Le altre figure coinvolte, in ambito informatico, sono il professore Alessandro Bevilacqua e l'ingegnere Alessandro Gherardi. Durante la progettazione sono state definite le funzionalità principali che il software AlizaMS doveva implementare e che rappresentano l'oggetto dell'attività di Tesi.

Durante tutta la fase di implementazione delle funzionalità richieste, con cadenza settimanale venivano effettuate delle *calls* per fare il punto sulla situazione tramite *Microsoft Teams*. Questi incontri permettevano di discutere di eventuali dubbi emersi in fase di implementazione, mostrare lo stato attuale del software e programmare le prossime attività da svolgere. É stato utilizzato *Trello* [58] come strumento per la gestione del lavoro svolto e per tenere traccia di quello ancora da realizzare tramite un'apposita bacheca [59]. Dopo aver ultimato gran parte delle funzionalità e ottenuto una versione stabile del software, quest'ultimo veniva testato da tutte le figure coinvolte per verificarne l'usabilità, sopratutto da parte dei medici. Successivamente a partire dai loro feedback, se necessario le funzionalità realizzate venivano 'raffinate': ovvero si apportavano piccole modifiche soprattutto legate alla GUI

per rendere l'utilizzo di AlizaMS ancora più agevole. In questo modo, si è realizzato un software in ambito medico utilizzato, testato e collaudato da parte delle figure professionali di riferimento, pronto all'uso.

4.2 Dettagli Implementativi

Il software è scritto in `C++` e utilizza le seguenti librerie esterne:

- ITK [54] valida libreria poichè è ampiamente utilizzata per lo sviluppo di programmi di segmentazione e registrazione delle immagini;
- QT [52] utilizzata per lo sviluppo di programmi con interfaccia grafica;
- OpenGL [53] utilizzata per la computer grafica 3D.

Il codice è ben documentato ed è stato utilizzato il *tool* Doxygen [60] come strumento per la generazione automatica della documentazione a partire dal codice sorgente. Tale documentazione è contenuta nella cartella `doc`.

4.2.1 Struttura del codice

Il codice presenta numerose directories. In questa sezione non verranno descritte tutte le directories presenti ma solo quelle oggetto di questo elaborato, che si differenziano rispetto alla versione base. La cartella `GUI` è quella relativa alla gestione della *User Interface*. Da questo momento in poi, tutti i riferimenti fatti ai singoli file si trovano all'interno di questa directory. Al suo interno ne troviamo un'altra con il nome di `LayerManager` in cui sono contenuti tutti i files creati per la realizzazione delle funzionalità. Questa cartella è organizzata nel seguente modo e presenta a sua volta le seguenti sotto-cartelle:

- `Graph` utilizzata per la gestione degli elementi grafici della ROI: vertici e archi;
- `Layer` utilizzata per l'implementazione delle classi Layer e ROI descritte in seguito;
- `RoiFormat` utilizzata per l'implementazione della logica per l'esportazione delle ROI nei formati `roi` e `txt`;
- `UITable` utilizzata per la gestione dei singoli elementi grafici della tabella delle ROI;
- `Utils` contiene i files di utilità;
- `Widgets` utilizzata per l'implementazione di nuovi widgets;
- `Xml` utilizzata per creare, leggere e modificare i files `xml` per la persistenza dei dati.

Allo stesso livello di `GUI` sono presenti i due moduli esterni: la cartella `rt-struct` contenente il codice per l'esportazione e importazione in questo formato e la cartella `prediction` contenente il codice per l'addestramento del SVM e per l'analisi delle ROI.

AlizaMS utilizza il modulo interno `MDCM` (*DICOM media storage*) per la lettura dei *tags* dei files Dicom. Tale modulo è basato su `GDCM` (*Grassroots DICOM*) una libreria multi-piattaforma scritta in `C++`.

4.2.2 Principali Classi

In questa sezione verranno descritte nel dettaglio le principali classi realizzate o ampliate per implementare tutte le funzionalità.

Regione di Interesse

L'implementazione del modulo *Roi Manager* descritto nel paragrafo 3.3 prevede la definizione di quattro classi, quella di *Layer* e *Roi* e i due elementi grafici che la compongono *Node* e *Edge*.

Come descritto nelle scelte progettuali, dal punto di vista implementativo sono stati introdotti i concetti di Layer e ROI. Ogni volta che l'utente crea un nuovo Layer verrà creata un nuova istanza della classe *Layer*. Ogni volta che l'utente disegnerà una nuova ROI nell'immagine della finestra di visualizzazione 2D, verrà creata una nuova istanza della classe *Roi*. É importante sottolineare come l'utente conosca solamente il concetto di ROI mentre è estraneo a quello di Layer. La modellazione delle due classi è mostrata nella figura 4.1.

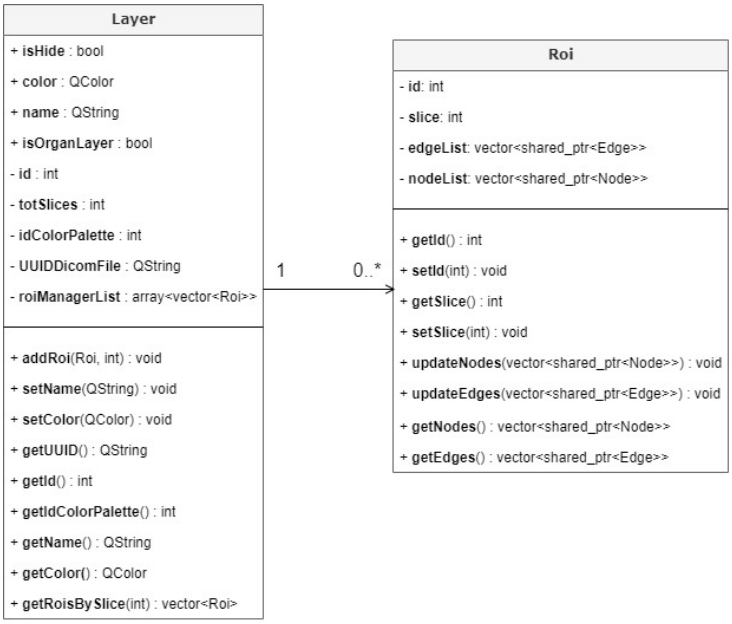


Figura 4.1: Modellazione delle classi *Layer* e *Roi*.

In questo caso la molteplicità indica che ad ogni *Layer* può corrispondere zero o più *Roi* e viceversa, che ogni *Roi* può essere associata ad un unico *Layer*. La classe *Layer* presenta come attributi le informazioni che l'utente ha fornito all'atto della creazione, il numero totale di *slices* dello studio aperto, il suo UUID e un *array* contenente tutte le ROI associate a quel Layer. Ogni cella dell'array contiene un lista (vettore) di ROI che rappresenta una specifica *slice*. Ad esempio, se si considera uno studio composto da 20 *slices* allora le prime 20 celle dell'*array* rappresentano le 20 *slices*. Ad ogni *slice* è possibile disegnare più ROI, per questo motivo ogni cella è una lista di ROI.

La classe *Roi* è caratterizzata dagli attributi *id* e *slice* che permettono di identificarla univocamente all'interno di uno studio. Inoltre ha due liste che si riferiscono rispettivamente all'insieme degli archi (*edges*) e all'insieme dei vertici (*nodes*) che la compongono. Si noti che tali liste contengono degli *shared pointer*, particolari *smart-pointer* [61] in grado di condividere informazioni di un oggetto o di una classe. Quando il *reference counter* ovvero il contatore dei riferimenti a cui il puntatore è associato è zero, il puntatore si dealloca liberando memoria, riducendo così la presenza di *memory leak*. Per questo motivo vengono definiti puntatori intelligenti.

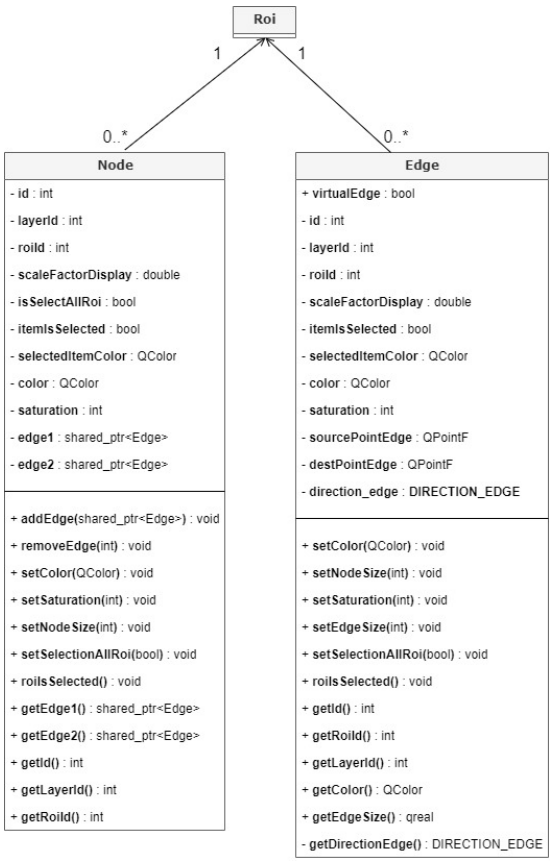


Figura 4.2: Modellazione delle classi *Node* e *Edge*

La ROI è composta da due elementi grafici principali: un vertice e un arco. La figura 4.2 mostra come la classe *Node* e la classe *Edge* sono state modellate e come queste sono associate

alla classe *Roi*. Ogni *Roi* può avere zero o più elementi grafici, viceversa ogni istanza di *Node* e *Edge* è associata esclusivamente ad una *Roi*. Entrambi gli elementi grafici estendono la classe *QGraphicsItem* [62] di QT, che fornisce tutti gli strumenti necessari per manipolare gli oggetti grafici. I vertici e gli archi sono individuati univocamente dalla tripla: *id*, *roiId* e *layerdId*. Ogni vertice ha il riferimento ai due archi a cui è collegato mentre ogni arco memorizza le coordinate del segmento dell'arco (*sourcePointEdge* e *destPointEdge*). Seguono ulteriori attributi e funzioni per la visualizzazione e la gestione della selezione/deselezione di ogni elemento grafico.

Dal diagramma delle classi (Capitolo 3, figura 3.7) sono presenti ulteriori classi di interesse: già presenti nella versione base di AlizaMS ma ulteriormente ampliate. Queste ultime sono:

- *Main Window*;
- *Aliza*;
- *Graphics Widget* e *Graphics View*.

La classe *Main Window* implementa il layout della finestra principale di AlizaMS e gestisce tutta l'interfaccia utente. Implementa tutti i comandi sotto forma di azioni (istanze di *QAction*) relative alle funzionalità di base e alle funzionalità di gestione delle ROI. Ogni comando può essere richiamato dall'utente in diverse modalità: tramite una voce del menù, un pulsante presente nella barra degli strumenti e uno *shorcut* da tastiera. Poiché l'utente si aspetta che ogni comando verrà eseguito sempre allo stesso modo, indipendentemente dall'interfaccia utente e modalità utilizzata, è utile rappresentare ogni comando come un'azione. In questo modo, tutte le azioni che fanno riferimento sempre allo stesso comando, saranno mantenuti automaticamente sincronizzati grazie a QT.

Figura 4.3: Modellazione della classe *Main Window*.

Nel diagramma UML (figura 4.3), è presente una piccola parte della classe, data dall'elevato numero di attributi e funzioni presenti. La modellazione è comunque esemplificativa: come attributi sono presenti tutte le istanze relative ai comandi di tutti gli elementi grafici che compongono l'interfaccia utente. Ad esempio è possibile notare *selectAllRoiAct* e *deleteAct* utilizzati rispettivamente per la selezione ed eliminazione di una ROI. Ad ognuno di questi comandi è associata una funzione, tramite il meccanismo *signal* e *slot* fornito da QT, che esegue la logica del comando quando utilizzato dall'utente. Le funzioni della classe rappresentano appunto la logica dei comandi, eseguite quando *triggerate* dall'utente o quando aggiornano lo stato di un particolare elemento grafico.

La classe *Aliza* è una della classi che rappresenta il *core* del sistema. L'immagine 4.4 mostra gli attributi e le funzioni presenti in questa classe relativo al contesto dell'elaborato.

Figura 4.4: Modellazione della classe *Aliza*.

Essendo una classe *Core* al suo interno sono presenti le principali funzionalità di base come ad esempio la lettura dei singoli file Dicom o degli interi studi e il calcolo degli istogrammi delle immagini. Invia le richieste utente, ricevute tramite la *Main Window*, alla classe di riferimento della specifica funzionalità, comunicando gli aggiornamenti alla GUI. Relativo alle ROI, gestisce la creazione del Layer come descritto anche nel paragrafo della progettazione e il processo di analisi delle segmentazioni. Implementa le funzioni *serializeData* e *deserializeData* per permettere rispettivamente di scrivere e leggere le informazioni dal file di persistenza. Ricevendo gli input dell'utente, *triggera* i diversi comandi (ad esempio la selezione o l'esportazione di una ROI). Per ognuno di questi comandi, richiama la

classe relativa alla funzionalità da eseguire. Gli attributi presenti, permettono di tenere traccia della *slice* e UUID correnti, rispetto al file Dicom aperto, e della palette dei colori personalizzata dall'utente.

La classe *Graphics Widget* e la classe *Graphics View* sono utilizzate per la gestione della scena della finestra di visualizzazione in modalità vista 2D. Permettono quindi di aggiungere e manipolare gli elementi grafici. La *Graphics Widget* funge da *wrapper* rispetto alla *Graphics View*.

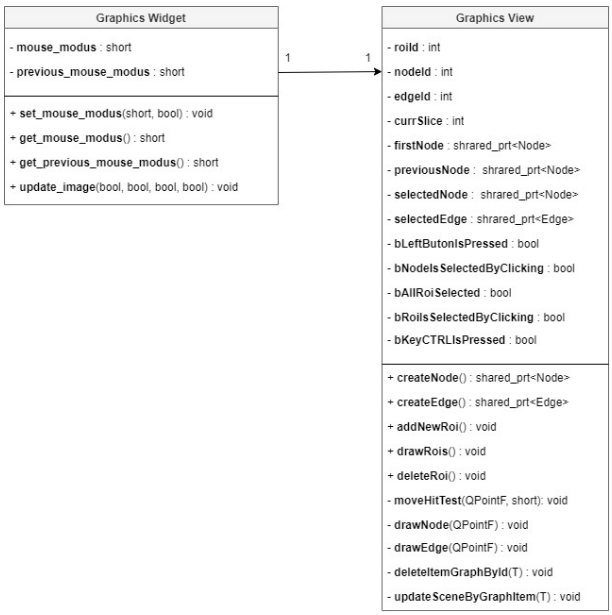


Figura 4.5: Modellazione della classe *Graphics Widget* e della classe *Graphics View*.

Partendo dalla *Graphics Widget*, essa gestisce le modalità del mouse. I programmatori di AlizaMS hanno definito delle modalità, tramite l'utilizzo degli interi, con cui gestire le funzionalità del mouse, relative a tutto ciò quello che può influenzare la scena. Per la gestione delle ROI due sono le modalità che sono state utilizzate: la *edit* e la *paint mode*. La funzione che gestisce la funzionalità del mouse in base alla modalità è la *moveHitTest* in *Graphics View*. Quest'ultima è la funzione principale dedicata al *check* degli *hit testing* in base alla specifica modalità impostata, in grado di intercettare tutti gli eventi associati al mouse. Ad esempio, quando il mouse si trova in *edit mode* gli elementi grafici sono selezionabili, questo significa che ad ogni spostamento del cursore è necessario verificare se questo, si trovi in prossimità di un elemento grafico per selezionarlo o deselezionarlo. Se invece la modalità è impostata su *paint*, non deve essere eseguito nessun controllo poichè questa modalità è utilizzata esclusivamente al disegno di una ROI. In questo caso ad ogni click del mouse corrisponderà la creazione di un nuovo vertice. Il diagramma illustra come la *Graphics View* è responsabile della gestione delle ROI. Gli attributi definiti sono utilizzati principalmente per la manipolazione di tutti gli elementi grafici associati alle ROI. Ad esempio, per tenere traccia del primo e ultimo vertice creato (quando è necessario chiudere

la ROI) e per memorizzare l'elemento grafico selezionato (quando è necessario eliminarlo). Inoltre sono presenti numerosi *flags* per indicare se un vertice o una ROI è selezionata, se il tasto sinistro del mouse è premuto e così via. Le funzioni presenti gestiscono la manipolazione delle ROI in base alle diverse funzionalità.

Esportazione e Importazione

Dal punto di vista implementativo, per eseguire l'esportazione nei formati `roi` e `txt`, è stata realizzata la classe *Roi Encoder*.

Nel caso del formato `roi` (formato proprietario di ImageJ [24]) è stato necessario re-implementarsi la classe relativa a questa funzionalità dal loro progetto: ovvero considerare la classe `RoiEncoder` [63] e realizzarla in `C++`.

Per quanto riguarda l'esportazione dei files in formato `txt`, è stata prevista la creazione di semplici files di testo che contengono:

- lo UUID legato allo studio Dicom;
- il numero di ROI segmentate (conteggiando le ROI multiple come un'unica ROI);
- il nome e il colore (in formato esadecimale) di tutti i Layers presenti;
- per ogni *slice*, il nome del file Dicom e tutte le ROI presenti. Per ognuna di esse, vengono salvate le coordinate `x` e `y` (in formato decimale) di ogni vertice.

Infine per l'esportazione ed importazione nel formato `rt-struct`, non essendo disponibile una libreria in `C++`, *cross-platform* in grado di poter leggere e modificare i *tags*, è stato necessario creare un modulo esterno. Dopo un'accurata ricerca, sono state trovate due librerie adatte al nostro contesto: *PyDicom* [64] e *rt-utils* [65]. La prima permette di leggere e modificare i *tags* dei files Dicom, la seconda permette l'esportazione/importazione del formato `rt-struct`. Sulla base di queste due librerie, è stato necessario realizzare un piccolo progetto in `Python`.

L'esportazione delle ROI in formato `rt-struct` prevede la creazione di specifici *tags*, secondo una sintassi ben precisa. Alcuni dei *tags* da salvare sono:

- *Structure Set ROI Sequence*: in cui viene salvata una lista contenente tutti i Layers presenti (id, nome, UUID) per tutte le *slices*;
- *ROI Contour Sequence*: in cui viene salvata una lista di tutte le ROI presenti per tutte le *slices*. Per ognuna di esse viene memorizzato il colore, la *slice* di riferimento, il numero di vertici e le loro coordinate. Prima del salvataggio, le coordinate devono essere convertite da sistema di riferimento immagine a sistema di riferimento paziente. L'equazione è mostrata nella figura 4.6 in cui:

1. P_{xyz} sono le coordinate (`x`,`y`, `z`) appartenenti al sistema di riferimento paziente;
2. S_{xyz} sono le tre coordinate del tag *ImagePositionPatient* (0020, 0032) che indicano la posizione del paziente;
3. X_{xyz} e Y_{xyz} sono le tre coordinate del tag *ImageOrientationPatient* (0020, 0037) che indicano l'orientamento dell'immagine;
4. Δ_i e Δ_j indicano la risoluzione dei pixels colonna e riga del tag *PixelSpacing* (0028, 0030);
5. i e j indicano rispettivamente l'indice colonna e riga del piano immagine ovvero le coordinate (`x`, `y`) da convertire.

Dopo aver ottenuto i valori, viene eseguito il prodotto matriciale per ogni vertice.

$$\begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} = \begin{bmatrix} X_x \Delta_i & Y_x \Delta_j & 0 & S_x \\ X_y \Delta_i & Y_y \Delta_j & 0 & S_y \\ X_z \Delta_i & Y_z \Delta_j & 0 & S_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i \\ j \\ 0 \\ 1 \end{bmatrix} = M \begin{bmatrix} i \\ j \\ 0 \\ 1 \end{bmatrix}$$

Figura 4.6: Equazione per convertire le coordinate da sistema di riferimento immagine a sistema di riferimento paziente.

L'importazione dei dati nel formato `xml` prevede il *parsing* del file, come avviene per la gestione della persistenza (descritto in seguito nel paragrafo 4.2.6). Per il formato `rt-struct` è stato necessario effettuare il *parsing* del file Dicom in modalità `rt-struct` e convertire successivamente le coordinate da sistema di riferimento paziente a sistema di riferimento immagine, applicando la matrice inversa rispetto a quella mostrata nella figura 4.6.

4.2.3 Classificazione

Per la parte di classificazione delle lesioni, è stato utilizzato il linguaggio di programmazione `Python` tramite il servizio *Google Colab*, poichè fornisce numerose librerie messe a disposizione per lo sviluppo di algoritmi di ML e DL. In particolare l'addestramento del modello SVM è stato condotto utilizzando la libreria *Scikit-learn* [66].

Come descritto nel paragrafo 3.4.3, l'avvio del processo di analisi delle ROI da parte dell'utente, coincide con la creazione di un processo di sistema che ri-chiamerà il modulo esterno. L'Api utilizzata per l'avvio del processo di sistema è la *CreateProcess* [67] di Windows che permetterà appunto di chiamare il modulo esterno. In caso di successo, il *task* salverà l'output nel *file system* altrimenti verrà mostrato un messaggio di errore.

4.2.4 Gestione del Fattore di Scala

Il disegno degli elementi grafici di una ROI (vertici e archi) all'interno della finestra di visualizzazione 2D, deve tener conto della risoluzione del display del dispositivo che stiamo utilizzando. La risoluzione è definita come il numero di pixels orizzontali e verticali presenti in uno schermo, spesso confusa come parametro per valutare la qualità dell'immagine visualizzata, che invece è misurata come densità di punti (per unità di misura lineare). Quando si considera un'immagine di dimensione 512 x 512 significa che questa è formata da 512 righe e 512 colonne, i quali formano una matrice. La risoluzione è molto importante poiché ci permette di scalare gli elementi grafici nelle immagini, a differenti dimensioni, che stiamo visualizzando. Se non scalassimo gli elementi grafici, cioè disegnati tutti alla stessa dimensione a prescindere dalla dimensione delle immagini, noteremo che in quelle più grandi (ad esempio 1024 x 1024) gli elementi grafici sono molto piccoli mentre più grandi in immagini di piccole dimensioni (ad esempio 128 x 128). L'obiettivo è quello di visualizzare gli elementi grafici tutti alla stessa dimensione a prescindere dalla dimensione dell'immagine, tenendo conto della risoluzione. Per fare ciò, è necessario calcolare il fattore di scala per ogni immagine come il rapporto tra la larghezza del display del dispositivo che stiamo utilizzando e la larghezza dell'immagine (lo stesso concetto può essere applicato con l'altezza al posto della larghezza). Quello che è stato descritto è il rapporto di similitudine lineare ovvero il rapporto tra le lunghezze dei lati omologhi di figure simili: in questo caso le figure simili sono il rettangolo dello schermo e dell'immagine. Dopo aver calcolato il fattore di scala, ogni qualvolta che è necessario disegnare un elemento grafico, la sua dimensione (fissata a priori) sarà moltiplicata con il relativo fattore per ovviare al problema descritto. La figura 4.7 mostra un esempio di due ROI scalate per il relativo fattore di scala in immagini a differenti dimensioni. Entrambi mostrano l'organo prostata: nell'immagine di sinistra è individuata dalla ROI di colore viola mentre nell'immagine di destra dalla ROI di colore gialla. La ROI rossa identifica una lesione.

4.2.5 Gestione degli Hit-Testing

Tutte le funzionalità che richiedono un'interazione utente sono state implementate attraverso il calcolo della distanza tra la posizione dell'elemento grafico e quella del cursore. Ad esempio, per la selezione di un vertice o di un arco, è necessario verificare che il cursore sia esattamente nella stessa posizione dell'elemento grafico o nei dintorni di quest'ultimo. In informatica, questi processi prendono il nome di *hit-testing*, tipici nelle applicazioni desktop che richiedono una GUI. Essi permettono di rispondere alle azioni utente che possono avvenire tramite il movimento e click del mouse. Nel file `graphicsview.cpp` è presente la funzione `moveHitTest` che si occupa di tutta la gestione degli *hit-testing*, in particolare permette di verificare se il cursore si trova:

1. all'interno della finestra di visualizzazione 2D;

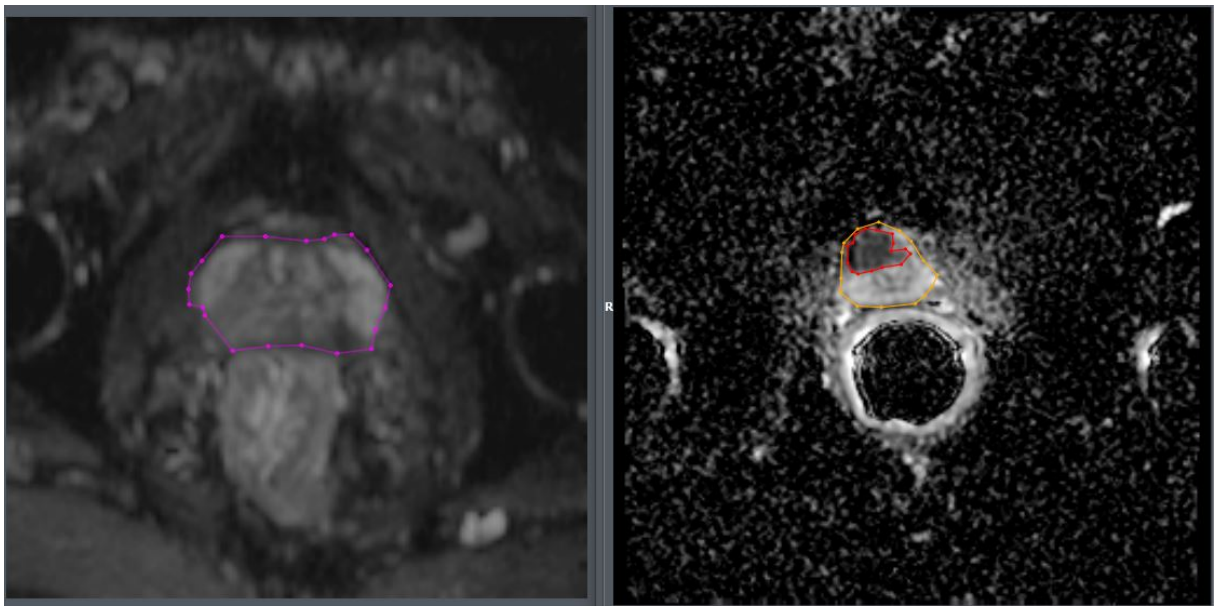


Figura 4.7: Esempio di gestione del fattore di scala: a sinistra un'immagine di dimensione 128 x 128 mentre a destra un'immagine di dimensione 512 x 512.

2. in prossimità o all'interno di un vertice;

3. in prossimità o all'interno di un arco;

4. all'interno di una ROI.

Nel primo caso basta semplicemente verificare che le coordinate (`x`, `y`) del cursore si trovino all'interno dell'immagine del file Dicom che si sta visualizzando.

Nel secondo e terzo caso è necessario calcolare la distanza tra la posizione del cursore e quella dell'elemento grafico. Per il punto 2 è stata utilizzata la distanza euclidea per calcolare la distanza tra le coordinate del cursore (posizione A) e quelle del centro del vertice (posizione B), come definita dalla formula:

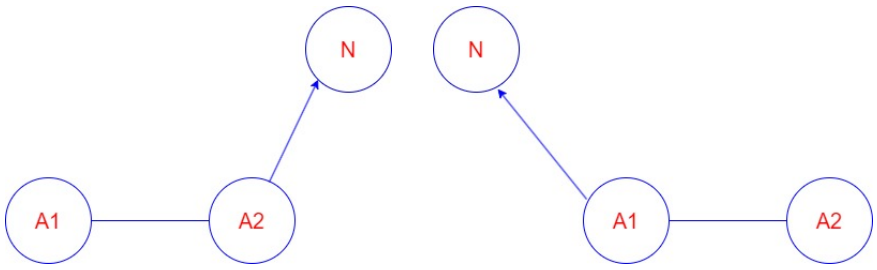
$$dist = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2}$$

Dopo aver calcolato la distanza, se è minore di una certa soglia (definita a priori e normalizzata per il fattore di scala) il vertice verrà selezionato altrimenti deselezionato, se era precedentemente selezionato, oppure nessuna azione verrà intrapresa. Lo scenario 3 è più complesso: in questo caso dobbiamo calcolare la distanza minima tra la posizione del cursore che chiameremo N, e quella dell'arco che possiamo considerare come un segmento di estremi A_1 e A_2. In primo luogo dobbiamo individuare la posizione più vicina sul segmento di coordinate (`x`, `y`) rispetto ad N e poi calcolare la loro distanza. L'idea è quella di utilizzare il concetto dei vettori. Suddividiamo il problema in 3 casistiche:

1. se il prodotto scalare tra il vettore $\vec{A_1A_2}$ e il vettore $\vec{A_2N}$ è positivo, maggiore di zero (ovvero se l'angolo compreso è acuto), allora la posizione più vicina a N è la posizione A_2, come mostra la figura 4.8. Dopo aver individuato la posizione, viene calcolata la distanza euclidea;

2. se il prodotto scalare tra il vettore $\vec{A_1A_2}$ e il vettore $\vec{A_1N}$ è negativo, minore di 0 (ovvero se l'angolo compreso è ottuso), allora la posizione più vicina a N è la posizione A_1, come mostra la figura 4.8. Dopo aver individuato la posizione, viene calcolata la distanza euclidea;

3. se il prodotto scalare è nullo (cioè uguale a 0), come mostra la figura 4.9, allora la posizione N è perpendicolare al vettore $\vec{A_1A_2}$. In questo caso la distanza perpendicolare può essere calcolare come seguente: $|dist| = |(\vec{A_1A_2} \text{ x } \vec{A_1N}) \ / \ |\vec{A_1A_2}||$.



(a) Caso 1: la posizione più vicina ad N è la posizione A_2 poiché il prodotto scalare è positivo.

(b) Caso 2: la posizione più vicina ad N è la posizione A_1 poiché il prodotto scalare è negativo.

Figura 4.8: Caso 1 e 2.

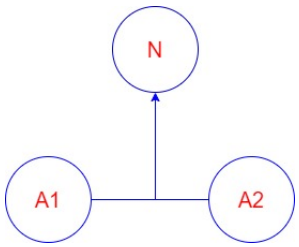


Figura 4.9: Caso 3: il prodotto scalare è 0 e la posizione più vicina ad N è quello individuato dalla distanza perpendicolare tra N e il vettore $\vec{A_1A_2}$.

Dopo aver ottenuto la distanza, secondo le 3 casistiche tra N e la posizione individuata sull'arco della ROI, se è minore di una certa soglia (definita a priori e normalizzata per il fattore di scala) l'arco verrà selezionato altrimenti deselezionato, se era precedentemente selezionato, oppure nessuna azione verrà intrapresa.

L'ultimo caso da analizzare è il punto 4 ovvero verificare se ad ogni spostamento del

cursore, questo si trova all'interno di una ROI qualsiasi. La funzione `pointIsIntoTheRoi`, all'interno del file `threshold.cpp` contenuto nella cartella `Graph`, mostra l'implementazione di questo *hit-testing* (frammento 4.1).

Descrizione dell'Algoritmo

L'idea è quella di tracciare un raggio semi-infinito orizzontale (fissando la coordinata `y` e incrementando la coordinata `x`) che parte da $x = 0$ e termina nel punto di test (posizione del cursore) e contare quanti archi della ROI attraversa. Se vengono attraversati un numero pari di archi allora il cursore è fuori dalla ROI altrimenti se il numero di archi è dispari, il cursore si trova al suo interno. L'algoritmo è chiamato il teorema di Jordan curve [68] è efficiente e funziona sia per poligoni convessi che concavi. Lo stesso algoritmo si può applicarlo tracciando un raggio semi-infinito verticale (al posto che orizzontale) fissando la coordinata `x` e incrementando la `y`. Facendo riferimento al frammento 4.1 [69], gli argomenti passati alla funzione sono:

- `point`: coordinate del cursore (`x` , `y`);
- `nodes`: lista (vettore) ordinata dei vertici di una ROI.

Vengono definite le variabili locali (`nNodes` indica il numero dei vertici della ROI, `i` e `j` sono i contatori del ciclo e `isIntoRoi` è il risultato da restituire). Alla riga `12` viene inizializzato il ciclo `for` in cui i vertici della ROI vengono scorsi da $[0, N-1]$. Ad ogni ciclo vengono memorizzate nelle variabili `INode` e `JNode` rispettivamente i vertici relativi alle variabili contatori `i` e `j` del vettore `nodes`. La condizione nell'`if` rappresenta il cuore dell'algoritmo: invertendo ripetutamente il valore di `isIntoRoi`, l'algoritmo conta quante volte la linea attraversa la ROI. La variabile `isIntoRoi` passa da 0 a 1 e da 1 a 0 ogni volta che il raggio orizzontale attraversa un arco. Ogni volta che il cursore si sposta, AlizaMS richiamerà questa funzione, tante volte quante solo le ROI presenti.

4.2.6 Persistenza dei Dati

Come descritto nella sezione 3.4.2 la persistenza dei dati avviene per mezzo di un file `xml` che viene mantenuto aggiornato ad ogni modifica eseguita dall'utente. L'`xml` è un metalinguaggio [70] per la definizione di linguaggi di *markup* utilizzati per salvare informazioni strutturate attraverso delle etichette o *tags* create appositamente dal programmatore. All'interno di ogni etichetta è possibile salvare le informazioni che la riguardano attraverso dei campi. I files caratterizzati da questa estensione, si basano su regole sintattiche che permettono di controllare il significato delle etichette contenuti in un documento di testo. Tutt'oggi questi files vengono comunemente utilizzati per l'esportazione di dati e per salvare specifiche configurazioni di applicazioni e sistemi operativi. Per gestire un documento `xml` in `C++` si è utilizzata la libreria *tinyxml2* [71].

Frammento 4.1: Implementazione della funzione `pointIsIntoTheRoi` per verificare se dato un punto questo si trova all'interno di una ROI.

```
1  bool pointIsIntoTheRoi(QPointF point,
2                         std::vector<std::shared_ptr<Node>>& nodes)
3  {
4      int nNodes = nodes.size();
5
6      qreal x = point.x();
7      qreal y = point.y();
8
9      int i = 0;
10     int j = 0;
11     int isIntoRoi = 0;
12     for (int i = 0, j = nNodes - 1; i < nNodes; j = i++)
13     {
14         QPointF INode = nodes.at(i)->pos();
15         QPointF JNode = nodes.at(j)->pos();
16
17         if (((INode.y() > y) != (JNode.y() > y)) &&
18         (x < (JNode.x() - INode.x()) * (y - INode.y()) /
19         (JNode.y() - INode.y()) + INode.x()))
20             isIntoRoi = !isIntoRoi;
21     }
22
23     return isIntoRoi;
24 }
```

Vengono descritte le etichette presenti:

- nell'etichetta principale *DicomFile* vengono memorizzate le informazioni riguardanti il file Dicom quindi il suo UUID, le *slices* totali (se si fa riferimento ad uno studio completo, altrimenti tale valore è settato a 1), il valore della *slice* corrente (l'istanza numerica). L'ultimo campo dell'etichetta *DicomFile* varia a seconda del tipo di file: se si apre un singolo file sarà presente l'etichetta *FileName* che indica il nome del file. Se si apre uno studio completo sarà presente l'etichetta *DirectoryName* che indica il nome della directory che contiene l'insieme dei files.

- nell'etichetta *ColorPalette* vengono salvati i 10 colori che compongono la palette utilizzata per mostrare uno specifico colore alla creazione delle ROI. Ogni colore è identificato dal *tag Color* e viene memorizzato attraverso i suoi valori RGB. All'atto della creazione del file `xml`, all'interno dell'etichetta *ColorPalette*, vengono salvati i colori di default. Man mano che l'utente modifica tali colori questi vengono aggiornati nel file.

- l'etichetta *Layer* identifica un Layer. Ciascuno di esso mantiene salvato un identificativo, il suo nome, il suo colore (in formato esadecimale), l'id del colore associato alla palette, l'informazione relativa al fatto se le ROI associate al Layer devono essere mostrate o rimanere non visibili. Quest'ultima informazione è riferita al campo *Hide* che assume il valore di 0 (non visibile) o 1 (visibile). L'ultima etichetta è *isOrganLayer* che esprime in forma booleana se tale Layer rappresenta l'organo di riferimento associato allo studio corrente. Nell'esempio 4.2, il file Dicom associato a questo file `xml` presenta un solo Layer, nel caso in cui sono presenti più Layers allora vedremo l'etichetta *layer* presentarsi tante volte quanti sono i Layers creati.

- all'interno di ogni Layer vengono salvate le ROI con la relativa etichetta *Roi*. Quest'ultima tiene traccia solamente del suo id. Anche in questo caso se il file Dicom presenta più ROI appartenenti a questo Layer o a Layers differenti, troveremo nel relativo file `xml` più volte l'etichetta *Roi* all'interno della specifica etichetta *layer*.

- all'interno del *tag Roi* vengono memorizzati i vertici. Ogni vertice è individuato dall'etichetta *Node* e tiene traccia del suo id e della sua posizione nelle coordinate immagine `x` e `y` in formato decimale. Per evitare informazioni ridondanti relative agli archi, i vertici vengono memorizzati in senso orario (o antiorario) in modo tale da poter ricostruire come questi sono collegati tra loro. Ogni qualvolta che l'utente interagisce con la ROI (spostamento, eliminazione di un vertice, ...) il file `xml` verrà aggiornato.

Frammento 4.2: Esempio di un file `xml` per la persistenza dei dati.

```
<DicomFile UUID="X.X.X.X.X.X.XXXXX.X.X.X.XXXXXXXXXXX" TotSlice="10" Slice="2"
    FileName="Example.dcm">
    <ColorPalette>
        <Color R="255" G="0" B="0"/>
        <Color R="0" G="255" B="0"/>
        <Color R="0" G="0" B="255"/>
        <Color R="255" G="255" B="0"/>
        <Color R="85" G="0" B="255"/>
        <Color R="0" G="255" B="255"/>
        <Color R="255" G="170" B="0"/>
        <Color R="255" G="0" B="255"/>
        <Color R="170" G="170" B="100"/>
        <Color R="130" G="60" B="5"/>
    </ColorPalette>
    <Layer Id="1" Name="Lesion 1" Color="#ff0000" IdColorPalette="0" Hide="0"
        IsOrganLayer="0">
        <Roi Id="0">
            <Nodes>
                <Node Id="0" X="399.18644067796612" Y="276.82711864406781"/>
                <Node Id="1" X="434.05272509794929" Y="278.73056154186361"/>
                <Node Id="2" X="431.29491525423725" Y="309.80338983050848"/>
                <Node Id="3" X="420.01355932203387" Y="344.5152542372881"/>
                <Node Id="4" X="392.24406779661018" Y="343.64745762711863"/>
                <Node Id="5" X="374.02033898305086" Y="319.34915254237285"/>
            </Nodes>
        </Roi>
    </Layer>
</DicomFile>
```

Il frammento 4.2 mostra un esempio di file `xml` per il salvataggio delle ROI.

5 Risultati

In questo capitolo verranno mostrati i test effettuati per il corretto salvataggio delle ROI sul file esterno e i risultati ottenuti a partire dal software sviluppato.

5.1 Validazione delle Regioni di Interesse

Una fase importante durante il salvataggio delle ROI è stata quella della loro validazione, ovvero verificare che le coordinate salvate nel file `xml` corrispondono effettivamente alle coordinate immagine in AlizaMS, senza la presenza di *offset* (differenza rispetto ad un valore di riferimento che in questo caso coincide con le coordinate di ogni vertice). La validazione risulta essere una fase fondamentale poichè se le coordinate sono affette da un *offset*, queste andrebbero a sfalsare l'esportazione della regione selezionata, da utilizzare per la successiva fase di classificazione. La validazione è stata condotta in primo momento, su immagini sintetiche e successivamente su sequenze Dicom fornite dal CVG [55] contenenti lesioni della prostata.

L'immagine 5.1 mostra un'immagine sintetica utilizzata per i test preliminari. In questo caso la ROI è rappresentata dal rettangolo grigio.

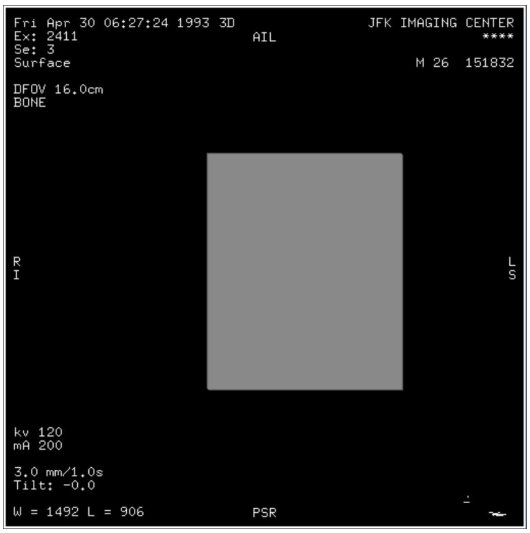


Figura 5.1: Immagine sintetica utilizzata per i test preliminari.

Per rendere più efficace questa fase è stato deciso di effettuare la validazione sia su un'immagine di dimensione 512 x 512 sia su un'immagine di dimensione 128 x 128. Questo per

verificare che al variare delle dimensioni, gli elementi grafici vengono visualizzati correttamente in funzione del rispettivo fattore di scala. Inoltre, per ogni diversa dimensione dell'immagine sono state disegnate tre ROI separate (ognuna su una nuova immagine della stessa dimensione) in cui per ognuna di esse, il centro dei loro vertici ricade:

- all'interno del bordo del rettangolo della figura 5.1;
- sul bordo del rettangolo della figura 5.1;
- esterno al bordo del rettangolo della figura 5.1.

Per ogni dimensione sono stati eseguiti 3 test per un totale di 6 test di validazione. Per ogni validazione, a prescindere dalla dimensione del file Dicom scelto e da quale ROI veniva disegnata rispetto al bordo del rettangolo, le fasi eseguite sono:

1. disegno della ROI e acquisizione di uno screenshot;
2. esecuzione di un *cropping* della ROI disegnata al punto 1 e acquisizione di uno screenshot dell'area ritagliata. Il *cropping* è stato eseguito a partire dalle coordinate del centro di ogni vertice;
3. confronto visivo dei due screenshot acquisiti con verifica che la regione ritagliata coincida effettivamente con la ROI disegnata. Avendo un rettangolo di colore grigio su uno sfondo nero, il controllo visivo è stato sicuramente più facile.

Per testare la validazione, nel menù *Roi Manager* è stata aggiunta la voce *Crop Roi* che esegue il *cropping* di una ROI disegnata. La figura 5.2 mostra un esempio dei due screenshot acquisiti per la validazione di una ROI in cui i centri dei vertici ricadono sul bordo del rettangolo.

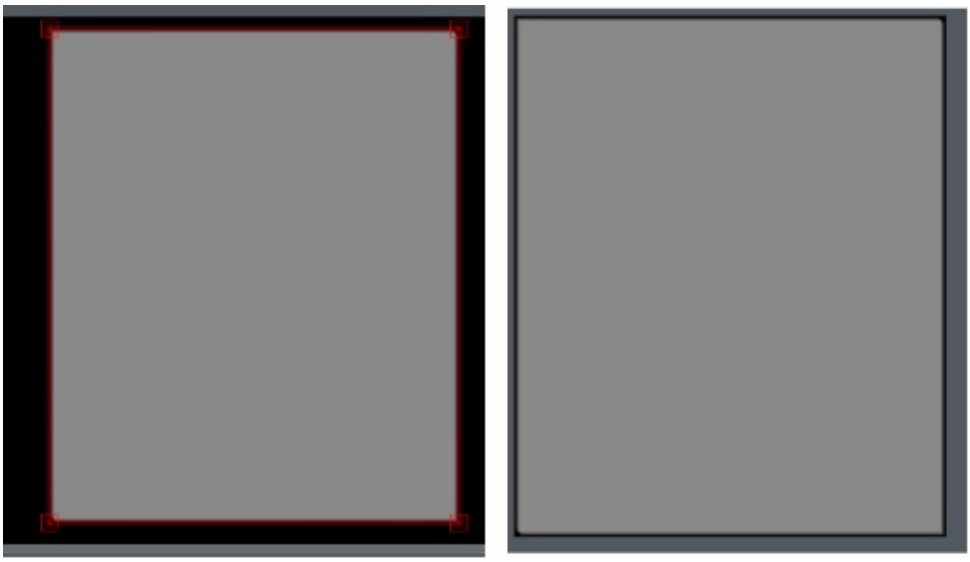


Figura 5.2: Confronto delle due ROI per effettuare la validazione.

A sinistra è presente lo screenshot della ROI disegnata e a destra lo screenshot del *cropping* dell'area. Concentrandosi sopratutto sui vertici del rettangolo è possibile notare che le due aree coincidono perfettamente, senza la presenza di *offset*.

Dopo aver effettuato i test preliminari, sono state considerate sequenze Dicom fornite dal CVG [55] in cui poter verificare nuovamente se le coordinate delle ROI sono affette da un *offset*. In questo caso sono state fornite delle ROI *sorgenti*, tramite files Dicom in formato `rt-struct`, segmentate dai medici con software di terze parti con cui testare le ROI di AlizaMS. L'idea è quella di importare le stesse segmentazioni *sorgenti* su una nuova sequenza Dicom dello stesso studio tramite AlizaMS, e verificare se le coordinate di entrambe le ROI (cioè quelle appartenenti a quelle *sorgenti* e a quelle importate), coincidono a livello visivo e da un punto di vista qualitativo.

Dopo aver importato i files `rt-struct` in una nuova sequenza Dicom dello stesso studio in AlizaMS, il software visualizzerà tutte le segmentazioni presenti a seconda della *slice* di riferimento che si sta visualizzando. Per verificare la presenza di un *offset* da un punto di vista qualitativo, è stata effettuata una sottrazione dei valori pixel a pixel tra due segmentazioni appartenenti alla stessa *slice*. Per quest'ultimo motivo, sono stati acquisiti due screenshot rettangolari che rappresentano delle *bounding box*, tali da contenere tutta la ROI in esame.

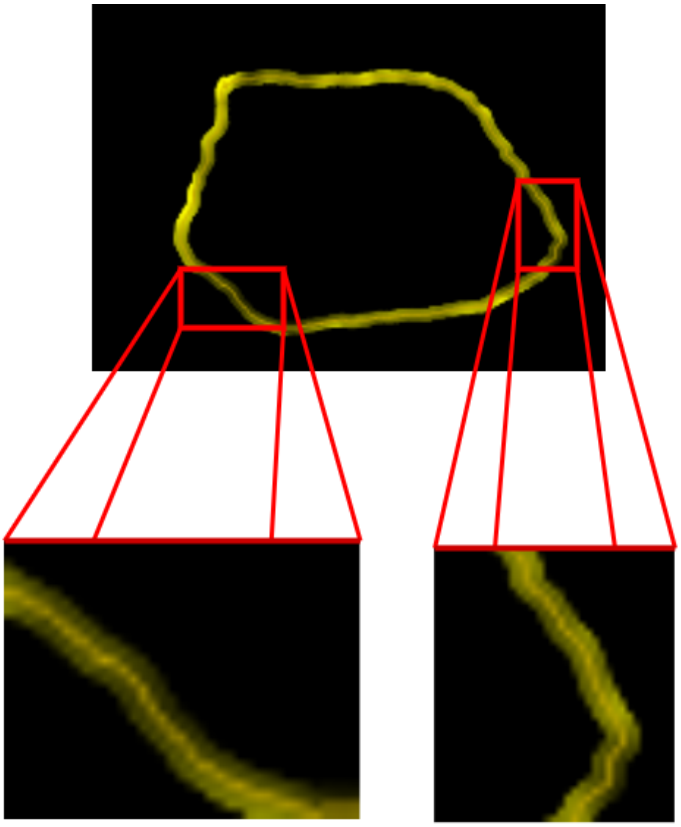


Figura 5.3: Validazione delle ROI tramite sottrazione delle immagini.

La sottrazione è stata eseguita come differenza assoluta dei valori pixel a pixel tra le due immagini. Poichè la ROI disegnata in AlizaMS risulta avere una dimensione degli archi e dei nodi, in termini di pixel, più grande rispetto alla ROI in formato `rt-struct` (la ROI è individuata con un margine di un pixel), l'immagine relativa alla ROI di AlizaMS è stata sottratta alla ROI *sorgente* così da poter apprezzare il risultato della sottrazione. L'immagine 5.3 mostra il risultato ottenuto dal test descritto. É presente l'immagine contenente tutta la ROI di colore giallo ottenuta per differenza dei valori dei pixel. L'area nera individua pixel con valori pari a 0 pertanto, gli stessi pixel nelle due immagini acquisite hanno lo stesso valore. A partire dall'immagine della ROI ottenuta per differenza, sono state selezionate alcune parti di interesse in modo tale da apprezzare maggiormente il risultato mostrato. Nelle porzioni selezionate, è possibile notare il contorno appartenente alla ROI disegnata in AlizaMS e al centro di esso, una linea più piccola. Quest'ultima linea rappresenta la ROI *sorgente* nel formato `rt-struct`. Se ci si concentra sul contorno della ROI *sorgente*, è possibile evidenziare come questo resti sempre centrato (o contenuto nel mediano) rispetto a quello più esterno, indicando l'assenza di un *offset*. Viceversa in caso di *offset*, il risultato ottenuto a partire dalla differenza dei valori dei pixel delle due immagini, dovrebbe contenere una variazione di intensità (che equivale a valori dei pixel diversi da zero) nelle aree nere, oltre a quella della ROI di colore giallo come mostrato in figura 5.3. Sulla base di quest'ultimo aspetto, la parti della ROI selezionate mostrano appunto una netta distinzione tra il contorno e l'area di colore nero.

Grazie alla validazione delle ROI, è stato possibile determinare sia a livello visivo che qualitativo la corretta implementazione e visualizzazione delle ROI.

5.2 Disegno di una Regione di Interesse

L'immagine 5.4 mostra un esempio di segmentazione assistita. Sono stati creati due Layers (vedere tabella delle ROI), in cui quello di colore arancione rappresenta l'organo di riferimento. L'utente sta disegnando la ROI e può tenere traccia della posizione del cursore tramite la *Focus View*. Il quadrato di colore verde brillante indica di essere in *paint mode*. Come descritto, la *Focus View* mostra una vista dall'alto dell'immagine e la segmentazione non ancora completata, mentre la finestra in modalità vista 2D mostra una vista del particolare. Inoltre nella *Focus View* è presente un cerchio che indica la posizione corrente del mouse.

Dopo aver completato la ROI, quando si è in *edit mode*, è possibile interagire con essa. La figura 5.5 mostra appunto quest'ultimo aspetto. Se l'utente muove il mouse sopra agli elementi grafici di una ROI, si noterà che il cursore cambierà icona e che gli elementi grafici verranno selezionati con il colore complementare rispetto al colore del Layer di

appartenenza. In questo modo è possibile interagire con gli archi e vertici di una ROI.

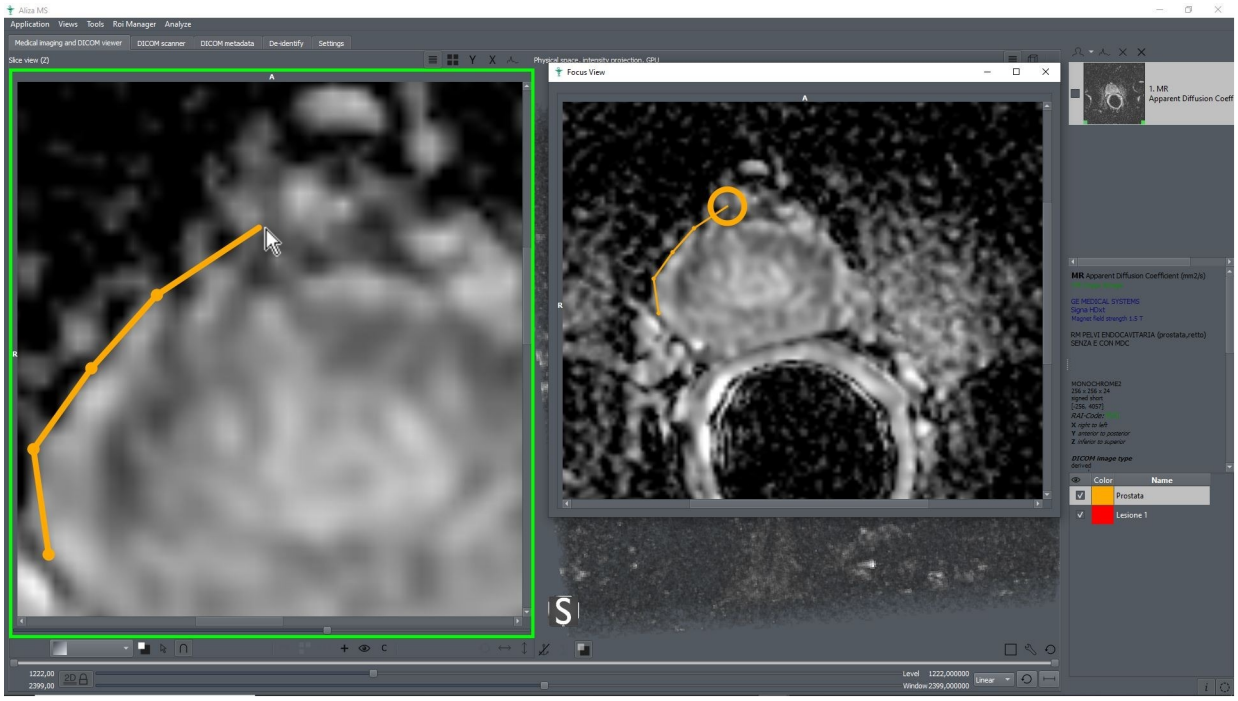


Figura 5.4: Disegno di una ROI con visualizzazione della *Focus View*.

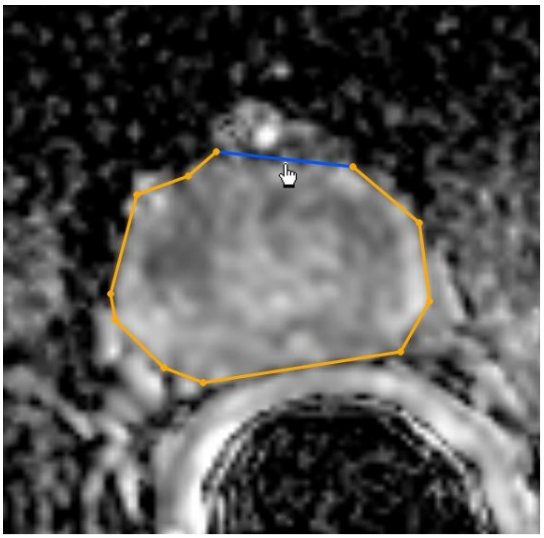


Figura 5.5: Interazione con una ROI.

5.3 Esportazione in Formato Rt-struct

Prima di poter esportare le segmentazioni in formato `rt-struct` è necessario crearle in una o più *slices*. Successivamente, dopo aver selezionato la voce *Export Roi* in *rt-struct*, verrà prodotto un nuovo file Dicom che se aperto in AlizaMS, visualizzerà un esempio dello stesso tipo mostrato in figura 5.6.

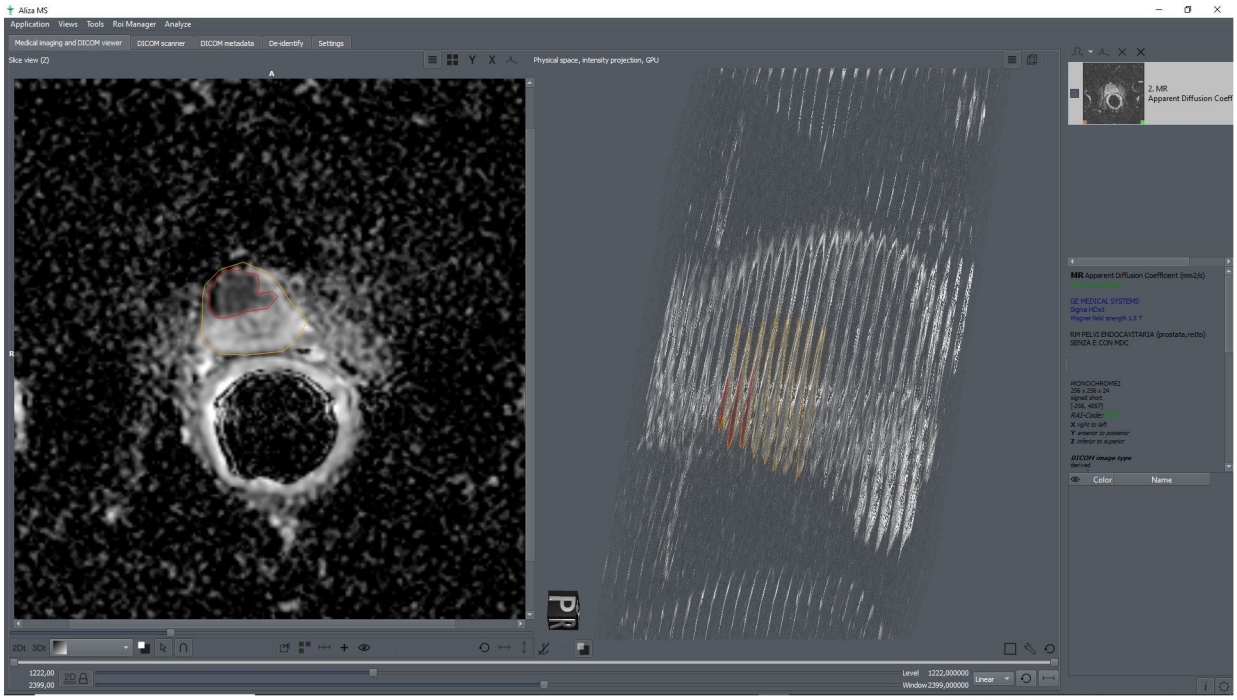


Figura 5.6: Visualizzazione di un file Dicom con ROI in formato `rt-struct`.

A sinistra, è possibile vedere le ROI in `rt-struct` contenute in ogni singola *slice* mentre a destra tutte le ROI esportate presenti nel volume 3D. In questo caso la ROI di colore arancione indica l'organo prostata mentre la ROI di colore rosso indica la lesione.

Una differenza che possiamo notare è la dimensione del contorno, molto più piccolo rispetto alle ROI disegnate con l'interfaccia grafica sviluppata. Il formato `rt-struct` non prevede nessuna forma di interazione con le ROI, destinate solamente alla visualizzazione.

5.4 Classificazione di una Regione di Interesse

La fase di *training* del modello SVM è stata suddivisa in diversi addestramenti, secondo quanto progettato nella fase precedente (Capitolo 3, paragrafo 3.4.5) utilizzando:

- un numero di iterazioni (N) pari a 100;
- la *K-Fold Cross Validation* sia con $K = 3$ che con $K = 5$;
- *Random Search* e *Grid Search* come ottimizzatori. Entrambi richiedono come input un insieme di valori per ogni iper-parametro da ottimizzare. Quello utilizzato nei vari addestramenti è il parametro C (fattore di regolarizzazione). I valori forniti sono: 0.1, 1, 10, 100, 200, e 500.

Al variare di K e dei vari ottimizzatori, è stato selezionato il modello più performante. Nella versione originale, quella in `Matlab`, l'SVM è stato addestrato utilizzando l'ottimizzatore Bayesiano. Nel caso della classificazione in `Python` non è stato possibile utilizzare questo ottimizzatore poichè non fornito dalla libreria *Scikit-learn* e dato il poco tempo a disposizione, non è stato possibile implementarlo manualmente.

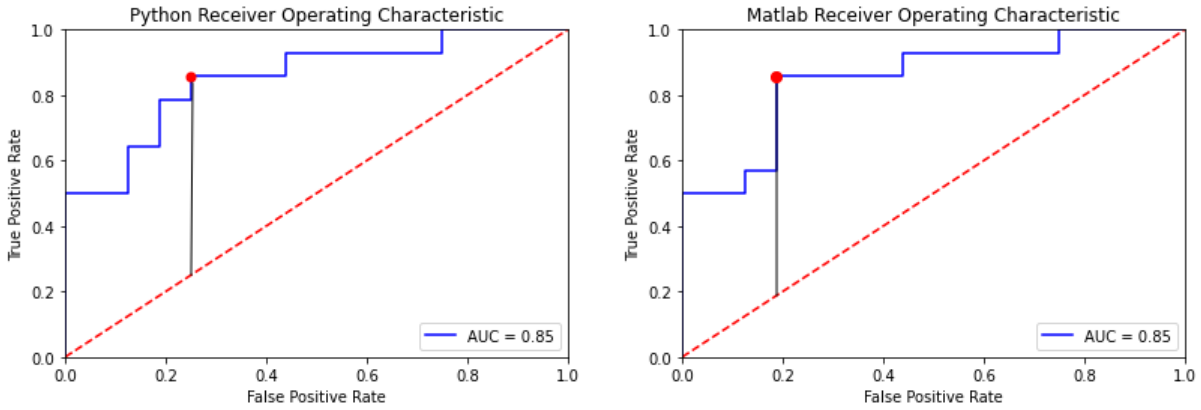
La tabella 5.1 confronta i modelli migliori sul *Test Set* tra quello ottenuto in `Python` (con soglia di decisione pari a 0, valore di default della libreria *Scikit-learn*) e quello già realizzato in `Matlab`. Vengono mostrate le metriche determinanti nella scelta del modello migliore tra quelli ottenuti nei vari addestramenti.

	SVM (`Python`)	SVM (`Matlab`)
Folds (K)	3	3
Ottimizzatore	Grid Search	Bayesiano
TP	12	12
TN	11	13
FP	5	3
FN	2	2
Accuratezza	76,66 %	83,33 %
TNR	0,6875	0,8125
TPR	0,8571	0,8571
AUC	0,8527	0,8527
BM	0,5446	0,6696

Tabella 5.1: Migliori modelli a confronto.

Analizzando i risultati ottenuti dal modello in `Python` è possibile notare come la sensibilità (TPR) ottenuta risulta essere in linea con il modello implementato in `Matlab` mentre la specificità (TNR) risulta essere più bassa. Pertanto il modello sarà in grado di identificare circa lo stesso numero di TP (lesioni clinicamente significative) rispetto al modello in `Matlab` ma un numero inferiore di TN (lesioni non clinicamente significative). Inoltre l'AUC risulta essere la stessa mentre la BM più bassa, data dai valori di sensibilità e specificità. Infine l'ultima differenza riguarda il diverso ottimizzazione utilizzato.

Un altro confronto può essere fatto tra i due modelli sulla base della ROC. Quest'ultima viene creata considerando il TPR nell'asse dell'ordinata e il FPR nell'asse delle ascisse a diversi valori di soglia. L'immagine 5.7 mostra le due ROC di colore blu: a sinistra è presente quella relativa al modello in `Python` e a destra quella relativa al modello in `Matlab`. L'area sotto la ROC è l'AUC che indica la prestazione media di un modello ed è utilizzata per selezionare il classificatore ottimale. Il miglior risultato è rappresentato dal punto in alto a sinistra di coordinata (0,1) che rappresenta il 100% di sensibilità e il 100% di specificità. La diagonale tratteggiata di colore rosso è chiamata linea di discriminazione (*line of no-discrimination*) e divide lo spazio ROC: i punti che ricadono sopra alla diagonale rappresentano buoni risultati mentre i punti che ricadono sotto sono detti 'punti di inversione', in cui il modello stima al 'contrario'. Sulla diagonale, invece, il modello stima con la stessa probabilità di un random *guess* e questo caso rappresenta quello peggiore. Dall'immagine, a parità di AUC, è possibile evidenziare che, fissata la soglia, il modello in `Python` è caratterizzato da un FPR più alto (e quindi una specificità più bassa) rispetto alla sensibilità (TPR). La metrica utilizzata per individuare il punto nella ROC ottimale (cerchio rosso) è l'indice di *Youden* (o *Youden's Index*) che indica la massima distanza tra il punto della ROC e la *line of no-discrimination*. L'obbiettivo in questo caso è di determinare il punto di *cut-off* ovvero il giusto compromesso tra il numero di TP e il numero di FP.



(a) ROC del modello in `Python`.

(b) ROC del modello in `Matlab`.

Figura 5.7: ROC a confronto tra i due modelli.

La tabella 5.2 mostra i valori delle metriche TNR, TPR e BM aggiornati rispetto alla precedente tabella 5.1, sulla base dell'indice di *Youden* individuato nella ROC (immagine 5.7a). La nuova soglia di decisione relativa all'indice di *Youden* è pari a 0,05.

TNR	TPR	BM
0,75	0,8571	0,6071

Tabella 5.2: Metriche riviste sulla base dell'indice di *Youden*.

La classificazione delle segmentazioni prevede l'esecuzione della fase di predizione da parte del software AlizaMS. Dopo la terminazione del processo, è possibile visualizzare l'output aprendo la *Features Color Map View*. Verrà visualizzata una nuova finestra (figura 5.8) in cui, se sono presenti delle segmentazioni relative alle lesioni, la *Features Color Map View* mostrerà una tra le quattro mappe colorimetriche disponibili, che l'utente può scegliere tramite l'apposito *box* (in basso a sinistra), altrimenti verrà mostrata l'immagine originale. La mappa colorimetrica mette in evidenza tutta l'area segmentata appartenente alla lesione e all'organo di riferimento. Inoltre mostra tutte le ROI create appartenenti alle lesioni, per la corrente *slice*.

A destra della *Features Color Map View* è stata creata la barra colorimetrica che indica il *range* dei valori, reali o normalizzati, che ogni specifica *feature* può assumere. Cinque sono i valori mostrati tra cui il minimo, il medio e il massimo. Ad ogni valore è associato un determinato colore in modo tale da poter apprezzare l'area colorata. Il valore massimo e minimo sono associati rispettivamente al colore rosso e blu. La figura 5.9 mostra le quattro mappe colorimetriche relative alle quattro *features* selezionate. L'utente può scegliere quale mappa colorimetrica visualizzare tramite il menù dedicato in basso a sinistra.

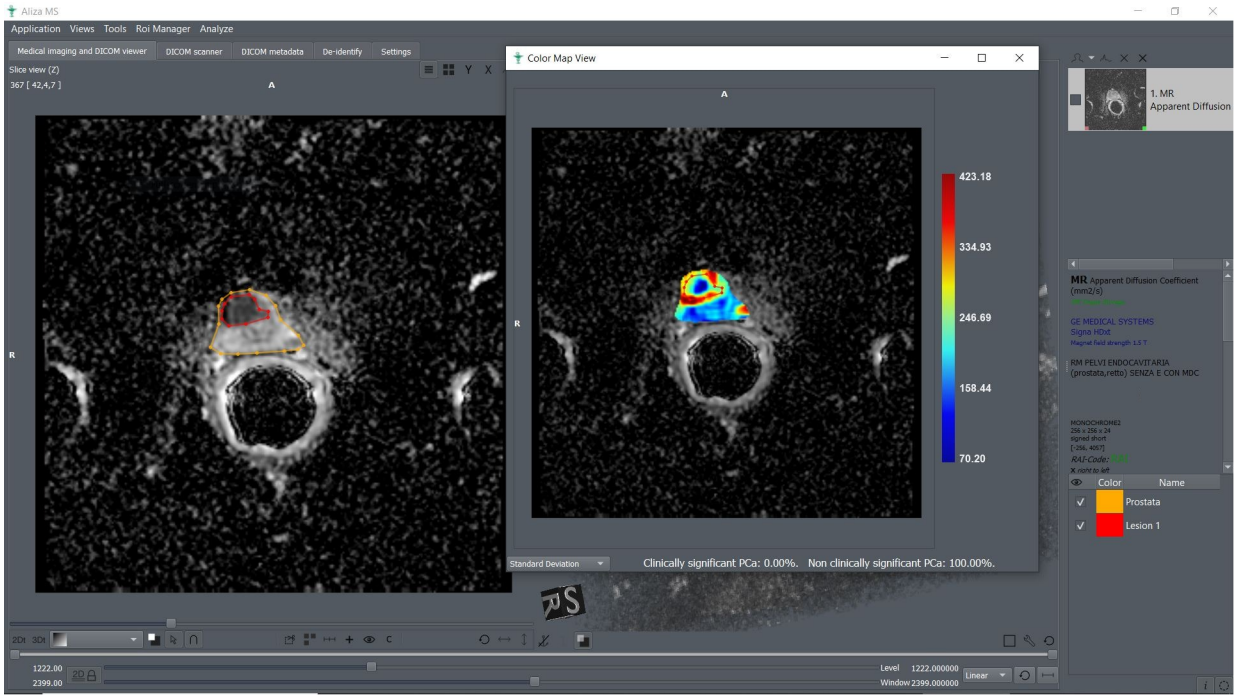
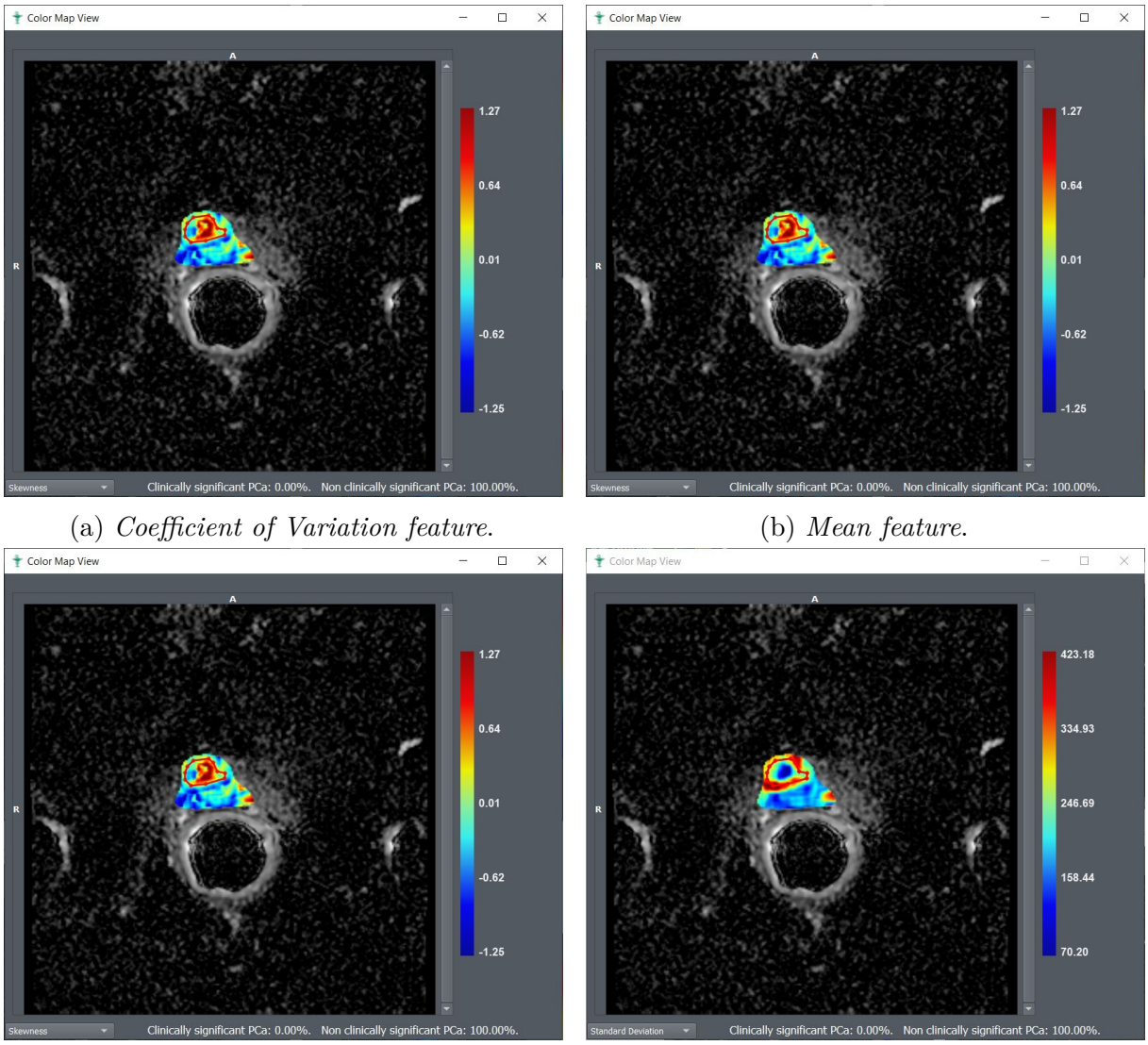


Figura 5.8: Visualizzazione della mappa colorimetrica dopo l'analisi delle ROI.

Nella *Features Color Map View*, vengono anche mostrate le probabilità che una specifica segmentazione può appartenere alla classe positiva (*Clinically Significant PCa*) o alla classe negativa (*Not Clinically Significant PCa*) del problema. Nel caso specifico, il classificatore ha predetto la classe negativa con una probabilità del 100% attribuendo all'area tumorale un valore di *Gleason Score* uguale o inferiore a 7A.



(a) *Coefficient of Variation feature.*

(b) *Mean feature.*

(c) *Skewness feature.*

(d) *Standard Deviation feature.*

Figura 5.9: Mappe colorimetriche relative alle quattro *features*. In ordine di visualizzazione: la *Coefficient of Variation*, la *Mean*, la *Skewness* e la *Standard Deviation*.

6 Conclusioni

La segmentazione ricopre un processo molto importante in ambito medico poichè permette di identificare e isolare una lesione rispetto al *background* dell'immagine. In questa Tesi è stato presentato il software AlizaMS in grado di eseguire la segmentazione assistita su immagini medicali, in particolare su files Dicom. Dispone di un'interfaccia grafica semplice e chiara, sviluppata per permettere al medico di facilitarne l'uso, riducendo gli errori inter-operatore e intra-operatore. Le funzionalità progettate permettono di creare ed interagire con le ROI. Inoltre, sono stati definiti diversi formati con cui esportare ed importare le ROI per aumentare l'interoperabilità con altri software di *imaging* medico. Il software prevede anche la possibilità di analizzare le segmentazioni dal punto di vista clinico. Un modello di intelligenza artificiale (SVM) è stato addestrato per perseguire questo obiettivo. In questo modo il medico dispone di un secondo parere, quello di AlizaMS, che può mettere in discussione o confermare la sua diagnosi.

6.1 Applicazioni

Lo scopo di questo software è di fornire uno strumento ai medici e allo staff sanitario in grado di eseguire la segmentazione assistita di immagini biomedicali nonché la diagnosi delle ROI individuate. Il software può essere anche un valido strumento per una futura elaborazione ed analisi delle segmentazioni create, nonché visualizzazione. Permette inoltre di tenere traccia delle dimensioni di una lesione nel corso del tempo, funzione utile per confrontare un'area pre e post-intervento.

6.2 Lavori Futuri

I lavori futuri che possono essere considerati riguardano eventuali miglioramenti sull'interfaccia grafica e sull'importazione delle ROI negli stessi formati in cui è possibile esportarle. Interessante è anche la possibilità di introdurre un *retraining* del classificatore per aumentarne l'accuratezza: ogni volta che il modello commette un errore evidente, classificando una lesione come clinicamente significativa o viceversa, il medico ha la possibilità di migliorare il modello, aggiungendo anche quest'ultimo caso ai dati del classificatore.

Bibliografia e Sitografia

1. *Visione Artificiale. Wikipedia, L'enciclopedia libera* https://it.wikipedia.org/wiki/Visione_artificiale.
2. *Global Computer Vision in Healthcare Market: Industry Analysis and Forecast (2018-2026) – by Type, Application, End User, Region. Mynewdesk, 19 Luglio 2019* https://www.mynewsdesk.com/in/maximize/pressreleases/global-computer-vision-in-healthcare-market-industry-analysis-and-forecast-2018-2026-by-type-application-end-user-region-2899137.
3. *Laparoscopia con realtà aumentata, Rai News.* https://www.rainews.it/rubriche/tg2medicina33/video/2022/02/Tg2-Medicina-33-del-01022022-60878afa-52f5-462b-845f-5a02dbb0ac1d.html.
4. *Sean Nolan, "How a Singapore hospital uses holograms to assist surgery". GovInsider, 24 Agosto 2021* https://govinsider.asia/citizen-centric/how-a-singapore-hospital-uses-holograms-to-assist-surgery-nuhs-ngiam-kee-yuan/.
5. *Workstation medicali per diagnostica per immagini, Medical Expo.* https://www.medicalexpo.it/fabbricante-medico/workstation-medicale-diagnostica-immagini-40615.html.
6. *Workstation medicali per diagnostica per immagini, Philips.* https://www.philips.it/healthcare/resources/landing/enterprise-imaging/diagnostics.
7. *Display medicali per diagnostica per immagini, Barco.* https://www.barco.com/it/products/medical-display-controllers.
8. *Alberto Signoroni, Paolo Gibellini, Imaging digitale applicato al medicale, 2011.* https://elearning.unito.it/scuole_specializzazione/pluginfile.php/17001/mod_resource/content/1/imagingdigitaleapplicatoalmedicale-140507105626-phpapp02.pdf.
9. *Guida agli esami, AIRC.* https://www.airc.it/cancro/affronta-la-malattia/guida-agli-esami.
10. *Mario Bianchi, Francesco Rivara, Simone Rusca, Caratterizzazione della radiografia computerizzatab (CR/DR) con schermi al fosforo e confronto con la radiografia convenzionale a film (FR), 2009.* https://www.ndt.net/article/aipnd_journal/3_09%20-%20caratterizzazione%20della%20radiografia.pdf.

11. *Radiografia Digitale. Wikipedia, L'enciclopedia libera.* https://it.wikipedia.org/wiki/Radiografia_digitale.

12. *Tomografia Computerizzata. Wikipedia, L'enciclopedia libera.* https://it.wikipedia.org/wiki/Tomografia_computerizzata.

13. *Tomografia Computerizzata. AIRC.* https://www.airc.it/cancro/affronta-la-malattia/guida-agli-esami/tc-tomografia-computerizzata.

14. *Scala Hounsfield. Wikipedia, L'enciclopedia libera.* https://it.wikipedia.org/wiki/Scala_Hounsfield.

15. *Tomografia a Emissioni di Positroni. Wikipedia, L'enciclopedia libera.* https://it.wikipedia.org/wiki/Tomografia_a_emissione_di_positroni.

16. *Imaging a Risonanza Magnetica. Manuale MSD.* https://www.msdmanuals.com/it-it/professionale/argomenti-speciali/principi-di-imaging-radiologico/imaging-a-risonanza-magnetica.

17. *Imaging a Risonanza Magnetica. Wikipedia, L'enciclopedia libera.* https://it.wikipedia.org/wiki/Imaging_a_risonanza_magnetica.

18. *Ecografia. Wikipedia, L'enciclopedia libera.* https://it.wikipedia.org/wiki/Ecografia.

19. *Standard Dicom* https://www.dicomstandard.org/.

20. *DICOM: uno sguardo al suo ambito e alla sua utilità* https://www.postdicom.com/it/blog/dicom-a-look-into-its-scope-and-utility.

21. *Dicom. Wikipedia, L'enciclopedia libera.* https://it.wikipedia.org/wiki/DICOM.

22. *Lista dei tag presenti in un file Dicom* https://dicom.nema.org/medical/dicom/current/output/chtml/part06/chapter_6.html.

23. *L'intelligenza Artificiale nell'Ambito Medico.* https://pmf-research.eu/lintelligenza-artificiale-nellimaging-medico/.

24. *ImageJ* https://imagej.nih.gov/ij/.

25. *MedSeg* https://www.medseg.ai/.

26. *Medviso* https://medviso.com/segment/.

27. *Aliza medical Software* https://www.aliza-dicom-viewer.com/.

28. *Sana K. Ardestani, Combination of texture, color and shape operators to describe image content: a survey.* https://www.preprints.org/manuscript/202012.0479/v1.

29. *Machine Learning: principi di funzionamento e applicazioni in Medicina, Giorgio De Nunzio.* http://ithaca.unisalento.it/nr-19_2022/articolo_IIp_11.pdf.

30. *Algoritmo Mean shift. Wikipedia, L'enciclopedia libera.* https://en.wikipedia.org/wiki/Mean_shift.

31. *Algoritmo K-means. Wikipedia, L'enciclopedia libera.* https://it.wikipedia.org/wiki/K-means.

32. *Rete Neurale U-Net. Wikipedia, L'enciclopedia libera.* https://en.wikipedia.org/wiki/U-Net.

33. *Support Vector Machine. Wikipedia, L'enciclopedia libera.* https://en.wikipedia.org/wiki/Support_vector_machine.

34. *Corinna Cortes, Support Vector Network, 1995.* https://link.springer.com/article/10.1007/BF00994018.

35. *Radial Basis Function. Wikipedia, L'enciclopedia libera.* https://en.wikipedia.org/wiki/Radial_basis_function.

36. *Overfitting. Wikipedia, L'enciclopedia libera.* https://en.wikipedia.org/wiki/Overfitting.

37. *Receiver operating characteristic. Wikipedia, L'enciclopedia libera.* https://en.wikipedia.org/wiki/Receiver_operating_characteristic.

38. *CyberKnife. Wikipedia, L'enciclopedia libera.* https://it.wikipedia.org/wiki/Cyberknife.

39. *Intervento al colon con la realtà aumentata, ospedale Papardo pioniere in Italia. Messina Today.* https://www.messinatoday.it/cronaca/ospedale-papardo-intervento-colon-realta-aumentata.html.

40. *All'ospedale Koelliker si opera con l'innovativa tecnica della laparoscopia 3D, Ospedale Koelliker.* https://www.osp-koelliker.it/allospedale-koelliker-si-opera-con-linnovativa-tecnica-della-laparoscopia-3d.

41. *Microsoft Hololens* https://www.microsoft.com/en-us/d/hololens-2/91pnzzznzwcp?activetab=pivot:overviewtab.

42. *Diagnosi automatizzata. Wikipedia, L'enciclopedia libera.* https://it.wikipedia.org/wiki/Diagnosi_automatizzata.

43. *CAD, Dott. Raffaele Leuzzi.* https://raffaeleleuzzi.it/malattia/cad.php.

44. *Il CAD come aiuto nella diagnosi precoce del tumore del polmone, Medici Oggi.* https://medicioggi.it/contributi-scientifici/il-cad-come-aiuto-nella-diagnosi-precoce-del-tumore-del-polmone/.

45. *Tumore alla Prostata, AIRC.* https://www.airc.it/cancro/informazioni-tumori/guida-ai-tumori/tumore-della-prostata.

46. *Gleason Score. Dr. Matteo Giglio, 2016.* https://www.urologo-genova.it/articoli/gleason-score-tumore-prostata-punteggio-prognosi.htm.

47. *Formato MetaIO. Wikipedia, L'enciclopedia libera.* https://en.wikipedia.org/wiki/Metaio.

48. *Formato Nifti. Wikipedia, L'enciclopedia libera.* https://en.wikipedia.org/wiki/Neuroimaging_Informatics_Technology_Initiative.

49. *Formato Nrrd. Wikipedia, L'enciclopedia libera.* https://en.wikipedia.org/wiki/Nrrd.

50. *Codice sorgente di Aliza Medical Software* https://github.com/AlizaMedicalImaging/AlizaMS.

51. *Licenza GPL-3.0. Wikipedia, L'enciclopedia libera.* https://it.wikipedia.org/wiki/GNU_General_Public_License.

52. *Lireria QT* https://www.qt.io/.

53. *Libreria OpenGL* https://www.opengl.org/.

54. *Libreria ITK* https://itk.org/.

55. *Computer Vision Group* https://cvg.deis.unibo.it/italiano/index.html.

56. *Automatically Extracted Machine Learning Features from Preoperative CT to Early Predict Microvascular Invasion in HCC: The Role of the Zone of Transition (ZOT), 2022.* https://www.mdpi.com/2072-6694/14/7/1816.

57. *Dipartimento di Medicina Specialistica, Diagnostica e Sperimentale* https://dimes.unibo.it/it/index.html.

58. *Trello, software per la gestione del lavoro* https://trello.com/it.

59. *Bacheca di Trello per tracciare il lavoro* https://trello.com/invite/b/hFRVBjNe/a493f79d160c7efca78ba53dbd7644ba/aliza-medical-imaging.

60. *Doxygen* https://doxygen.nl/.

61. *Smart Pointers* https://learn.microsoft.com/en-us/cpp/cpp/smart-pointers-modern-cpp?view=msvc-170.

62. *Classe QGraphicsItem* https://doc.qt.io/qt-6/qgraphicsitem.html.

63. *Classe Roi Encoder del progetto di ImageJ* https://github.com/imagej/ImageJ/blob/master/ij/io/RoiEncoder.java.

64. *Libreria PyDicom* https://pydicom.github.io/.

65. *Progetto rt-utils* https://github.com/qurit/rt-utils.

66. *Libreria Scikit-learn in Python.* https://scikit-learn.org/stable/.

67. *API di Windows per la creazione di un processo di sistema.* https://learn.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-createprocessa.

68. *Teorema della curva di Jordan Curve. Wikipedia, L'enciclopedia libera.* https://en.wikipedia.org/wiki/Jordan_curve_theorem.

69. *PNPOLY - Point Inclusion in Polygon Test, WRFranklin.* `https://wrfranklin.org/Research/Short_Notes/pnpoly.html`.

70. *XML. Wikipedia, L'enciclopedia libera.* `https://it.wikipedia.org/wiki/XML`.

71. *Libreria Tinyxml2* `https://github.com/leethomason/tinyxml2`.

72. *RadiAnt DICOM Viewer* `https://www.radiantviewer.com/it/`.