Alma Mater Studiorum · Università di Bologna

**SCUOLA DI SCIENZE**
**Corso di Laurea in Informatica**

# Denoising Di Immagini Tomografiche Con Reti Neurali

**Relatrice:**
**Chiar.ma Prof.**
**Elena Loli Piccolomini**

**Correlatrice:**
**Dott.**
**Elena Morotti**

**Presentata da:**
**Camilla Vescovi**

**III Sessione**
**Anno Accademico 2023/2024**

# Introduzione

L'intelligenza artificiale sta rivoluzionando sempre più settori, sia nell'ambito della vita quotidiana che nel campo della ricerca scientifica. In particolare, le reti neurali convoluzionali per l'elaborazione di immagini trovano applicazione ottimale nel perfezionamento delle immagini di origine tomografica. Queste ultime, che per loro natura necessitano di un'elaborazione digitale, presentano nella loro acquisizione vari ostacoli di origine fisica, matematica e informatica. Sebbene molti di questi siano stati superati grazie all'avanzamento dell'ingegneria medica, la bassa dose di radiazioni a cui un paziente può essere soggetto durante un esame continua a porre dei limiti alla qualità delle acquisizioni. Un'immagine tomografica affetta da rumore tende a nascondere e confondere i dettagli rilevati dalla TC e rende complessa, se non impossibile, una diagnosi accurata. Questa tesi esplora i metodi di acquisizione di immagini tomografiche e le cause del rumore, procedendo con l'analisi delle tecnologie basate su reti neurali convoluzionali, in particolare quelle dedicate all'*imaging* medico e all'eliminazione del rumore (*denoising*). L'obiettivo principale è di individuare un approccio basato su reti neurali che si riveli in grado di eseguire l'operazione di denoising in modo efficace, distinguendo in maniera accurata tra il rumore e i dettagli più fini dell'immagine. A questo scopo, si presenta nella tesi un nuovo framework basato su reti neurali, insieme ai risultati ottenuti a seguito del training e del testing. Questo framework presenta una struttura caratterizzata dalla presenza di due reti convoluzionali di tipo U-Net Residuale e dalla divisione del dataset in *patch*, tecnica per affrontare il problema della scarsa dimensione del dataset, tipico dei dati di origine medica. I dataset di training e testing usati nello studio della rete sono stati gentilmente forniti dall'azienda SeeThrough S.r.l[1]. L'obiettivo è migliorare i risultati ottenuti da framework più semplici ma più aggressivi, che rimuovendo il rumore eliminano anche dettagli fondamentali per la diagnosi. La tesi si divide su tre capitoli, di cui il primo è dedicato ad un'introduzione sulla storia e le tecniche della tomografia computerizzata, con un approfondimento sulle cause originanti del rumore. Il secondo capitolo è rivolto allo studio delle reti neurali convoluzionali, con un focus sui concetti e modelli necessari per l'introduzione della rete presa in esame. Si procede poi con l'illustrazione del framework proposto, soffermandosi sulla sua struttura e sull'architettura della rete utilizzata. Il terzo e ultimo

capitolo espone i risultati ottenuti, con un'analisi basata principalmente sull'esame visivo dei volumi tomografici ricavati.

# Indice

# Capitolo 1

# Tomografia computerizzata a fascio conico

## 1.1 Introduzione alla tomografia computerizzata

La tomografia computerizzata è una tecnica di *Imaging* che attraverso le proprietà dei tessuti e dei raggi X acquisisce e riproduce immagini e volumi anatomici. Il metodo sfrutta conoscenze di vari campi, quali la fisica, la medicina, la geometria e l'informatica.

La tomografia computerizzata (o TC) nasce negli anni sessanta dello scorso secolo, ma il principio matematico su cui si basa risale al 1917, quando l'austriaco Johann Radon dimostrò possibile ricostruire un volume mediante un numero infinito di proiezioni bidimensionali dell'oggetto stesso. L'idea, adattata al limite concreto dell'avere un numero finito di proiezioni, dovette attendere decenni prima di essere applicata alla disciplina medica. Negli anni cinquanta, durante i trattamenti di radioterapia, il fisico Allan Cormack osservò una variazione della distribuzione della dose di radiazioni, causata dalla disomogeneità dei tessuti. Ciascuno di questi assorbe o riflette la radiazione secondo un coefficiente di attenuazione specifico. Il fisico intuì che conoscendo i coefficienti delle aree in esame si potevano prevedere le irregolarità. Cormack riuscì nella misurazione di tali coefficienti solo nel 1963.

Fu un ingegnere inglese, sir Godfrey Hounsfield a costruire il primo prototipo di tomografo sperimentale. Questi faceva parte dello staff della EMI (Electrical Music Industry), azienda per la quale anni prima aveva già sviluppato EMIDEC 1100, il primo computer a diffusione commerciale della Gran Bretagna. Grazie al sostegno economico del governo britannico e ai grossi introiti che in quegli anni la EMI otteneva per il successo mondiale dei Beatles, Hounsfield riuscì a produrre un primitivo esemplare di TC: era costituito da una sorgente di isotopo $^{241}$Am e da un detector montati su una struttura capace di movimenti sia di traslazione che di rotazione. A

causa della bassa intensità della radiazione emessa dall'isotopo, il dispositivo impiegava circa nove giorni per l'acquisizione dell'oggetto e successivamente il computer elaborava per due ore e mezza il grosso numero di immagini acquisite. Per i loro sviluppi pionieristici nel campo della tomografia computerizzata Cormack e Hounsfield ricevettero il premio Nobel per la medicina nel 1979. Infatti, i principi su cui si basava il prototipo del 1967 sono li stessi su cui si fonda la moderna tomografia.

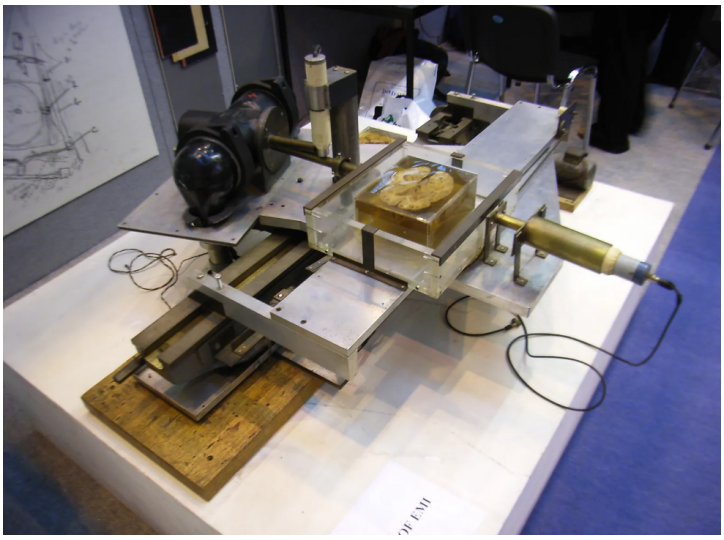


**Figura 1.1:** Il prototipo di TC scanner ideato da Godfrey Hounsfield, in esposizione allo U.K. Radiological Congress del 2005.

## 1.2 La legge di Lambert-Beer e il sinogramma

La struttura essenziale di uno scanner TC comprende una sorgente di radiazioni e un detector, posizionati uno di fronte all'altro, che ruotano o traslano intorno al paziente. Le geometrie e gli angoli di acquisizione si diversificano per scopo e periodo di sviluppo dello scanner, ma tra le varie disposizioni spiccano le TC di terza generazione. In questa architettura il detector non è singolo, ma composto da vari moduli disposti su un arco il cui centro è la sorgente dei raggi X. La sorgente e il detector rimangono fermi in relazione l'uno a l'altro, mentre la struttura ruota intorno al paziente. Questa architettura rappresenta la maggior parte dei TC scanner in commercio oggi e verrà assunta come modello nell'esposizione dei principi fisici e matematici su cui si basa la tomografia computerizzata.

Per semplificazione si illustra il processo in due dimensioni; inoltre, lo scanner si può assumere rettilineo anziché che curvo, poiché le unità di misura con cui si

lavora sono microscopiche ed è quindi lecito approssimare un tratto dell'arcata del detector ad un segmento.

Si consideri la sorgente $S$ che trasmette un raggio X indicato con $R$, la cui intensità è $I_0$. I raggi X hanno frequenza d'onda molto piccola, e di conseguenza, per la legge di Plank-Einstein, un'energia altrettanto alta che permette loro di attraversare la maggior parte della materia. Quando ciò accade, come durante un esame tomografico, l'oggetto attenua il raggio X con cui è stato inondato. La legge di Lambert-Beer descrive la quantità di raggio X iniziale che viene attenuata.

$$I = I_0 e^{-\mu x} \tag{1.1}$$

Dove $I$ è l'intensità del raggio dopo aver attraversato il corpo, $\mu$ è il coefficiente di attenuazione lineare del materiale e $x$ è lo spessore. Applicando il logaritmo naturale ad entrambi i membri di (1.1) si può ricavare l'espressione seguente.

$$-\ln\left(\frac{I}{I_0}\right) = \mu x \tag{1.2}$$

Nel caso in cui il coefficiente $\mu$ vari lungo il raggio $R$ è necessario definire una versione più generale della legge di Lambert-Beer. Si consideri per esempio un esame tomografico, in cui i raggi X attraversano sia diversi tessuti all'interno del corpo sia la zona d'aria presente attorno al paziente. Allora lo spessore $x$ percorso dal raggio diventa l'intera distanza $R$ tra la sorgente e il detector. Occorre quindi sommare i vari effetti che ogni segmento di materia ha sul raggio. Calcolando l'assorbenza sull'intero spazio di percorrenza $R$ si ottiene la seguente equazione della legge di Lambert-Beer:

$$\int_R \mu \, dx = -\ln\left(\frac{I_0}{I}\right) \tag{1.3}$$

Nella realtà l'applicazione di tale formula richiede un'inevitabile discretizzazione e il volume esaminato è suddiviso in piccolissime sezioni chiamate voxel, la più piccola sotto unità di un volume digitale. Ciascun voxel ha il proprio coefficiente $\mu_i$ e spessore $x_i$. La formula nel discreto è quindi:

$$\sum_{i=1} \mu_i x_i = -\ln\left(\frac{I_0}{I}\right) \tag{1.4}$$

L'esame tomografico procede ripetendo la proiezione per diversi angoli $\theta$ e salvando le misure effettuate dal detector in una matrice. Tali valori, elaborati e scalati per essere rappresentati sulla scala dei grigi, coincidono con i pixel dell'immagine acquisita in questa fase, il sinogramma, le cui righe corrispondono alle proiezioni ottenute ai diversi angoli e le colonne ai valori registrati dai sensori che compongono il detector. Per ottenere l'immagine tomografica dal sinogramma è necessario

utilizzare un algoritmo di ricostruzione tomografica, capace di generare la sezione bidimensionale dell'oggetto dati i valori di attenuazione misurati durante le varie proiezioni effettuate durante l'esame.

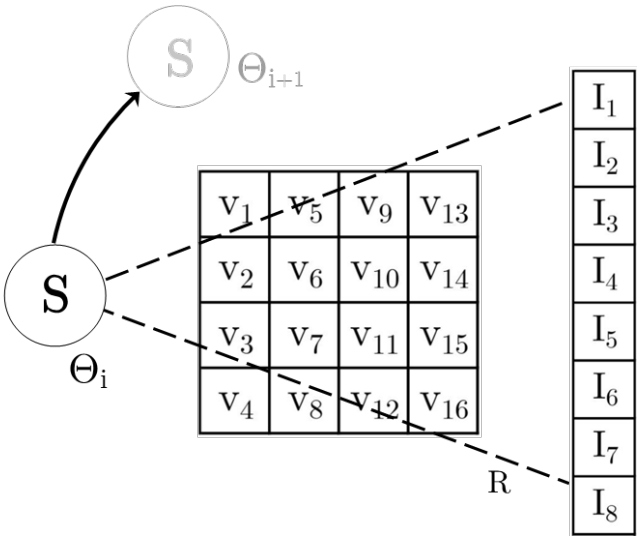



**Figura 1.2:** Modello essenziale di un tomografo. In figura si possono osservare la sorgente S, il volume in esame suddiviso in voxel e i vari moduli del detector, ciascuno rilevante un'intensità diversa che dipende dal coefficiente di attenuazione della materia attraversata. In seguito ad una prima acquisizione la struttura ruota e procede all'analisi da un secondo angolo.

## 1.3 CBCT: Cone Beam Computed Tomography

La tomografia appena descritta è detta *fan beam*, poiché il raggio emesso dalla sorgente ha forma a ventaglio. Esistono diversi tipi di tomografie, tra cui la Cone Beam Computed Tomography (CBCT), in cui il fascio utilizzato è conico e il detector ad esso opposto è di forma circolare. Questo tipo di esame è molto usato nel campo dell'odontoiatria grazie alla possibilità di convergere il raggio sulla zona di interesse, riducendo l'esposizione non necessaria del paziente. Un altro vantaggio della CBCT è la breve durata dell'esame: la forma conica del raggio permette di acquisire centinaia di immagini in una sola rotazione, diminuendo il tempo di scansione a meno di un 5% di quello della tomografia tradizionale.[MZGG09] Per quanto riguarda la ricostruzione delle immagini, se nella tomografia a ventaglio ogni sinogramma è indipendente e può essere trasformato in una fetta bidimensionale del volume finale, la *cone beam* necessita delle proiezioni dell'intero oggetto. Gli algoritmi di ricostruzione per la CBCT possono essere di due tipi: iterativi, ideali quando i dati sono incompleti o poco affidabili, e analitici, solitamente più appropriati per l'elaborazione di dati più esaurienti. Esistono però algoritmi analitici approssimativi in grado di gestire dati

moderatamente troncati. Questi metodi sono meno precisi, ma molto più veloci e leggeri computazionalmente rispetto alle tecniche iterative. A questa categoria appartiene l'algoritmo di Feldkamp-Davis-Kress (FDK).

### 1.3.1 L'algoritmo di Feldkamp-Davis-Kress

L'idea dell'algoritmo di Feldkamp-Davis-Kress è di estendere alla CBCT il metodo di ricostruzione tomografica Filtered Back-Projection (FBP) usato per le acquisizioni di tipo *fan beam*. Infatti, il fascio conico può essere pensato come multipli raggi a ventaglio inclinati di un angolo diverso, mentre ogni linea sul detector corrisponde ad un'acquisizione bidimensionale di un *fan beam*. La FBP si basa principalmente su due operazioni: l'applicazione di un kernel convolutivo (una specie di filtro matematico che sarà approfondito nel prossimo capitolo) e la retro-proiezione del valore di attenuazione acquisito. Quest'ultima si basa sul Teorema della slice di Fourier, che afferma la possibilità di ricostruire un'immagine bidimensionale date la sue proiezioni lungo diverse angolazioni. La retro-proiezione semplice consiste nel

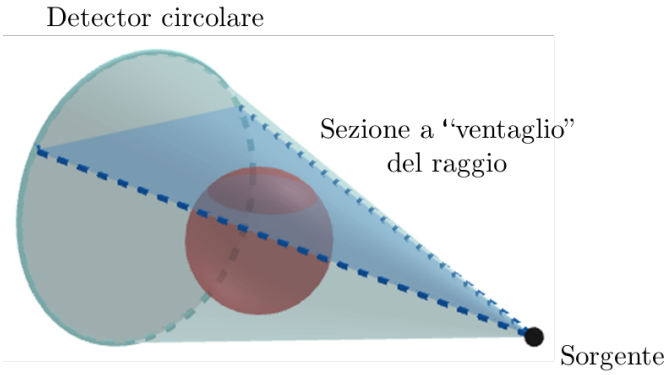



**Figura 1.3:** Schema semplificato di una Cone Beam Computed Tomography. In evidenza una delle sezioni *fan beam* usate nell'algoritmo FDK.

distribuire in parti uguali ai pixel lungo la traiettoria del raggio corrispondente il valore di attenuazione misurato dai vari detector, una specie di operazione opposta alla proiezione. Questa distribuzione è poi ripetuta per ogni angolo di acquisizione, sommando per ogni pixel i vari contributi di ciascuna retro-proiezione. Purtroppo la distribuzione uniforme dei valori, se non opportunamente corretta, introduce nell'immagine una sfocatura. Intuitivamente, i pixel più centrali alla figura sono irradiati da ogni angolazione e avranno quindi valore più alto rispetto a quelli periferici. Per correggere questa sovrapposizione, prima della retro-proiezione viene applicato al sinogramma un filtro di frequenza, detto *ramp filter*. Questa fase, detta di *filtering*, sopprime le componenti a bassa frequenza spaziale dei valori di attenuazione e ac-

centua invece i cambiamenti rapidi tra alta e bassa frequenza. Questa operazione riduce la sfocatura e mette in evidenza i confini delle strutture anatomiche.

L'algoritmo di Feldkamp-Davis-Kress prevede l'uso del metodo FBP, ma adattato alla natura tridimensionale del fascio di raggi X usato nella CBCT. Per questo motivo, la prima fase di FDK è la correzione prospettica: la maggior parte dei *fan beam* arrivano al detector non in maniera perpendicolare come nella tomografia classica, ma con una certa inclinazione che dipende dalla distanza dal centro della circonferenza. Occorre quindi trasformare i valori di attenuazione ottenuti in base alla posizione del raggio corrente, in modo da avere la componente perpendicolare della proiezione sul detector. Una volta eseguita questa operazione si procede all'applicazione dell'algoritmo Filtered Back-Projection sui *fan beam* e al calcolo dei valori dei pixel.

In questo elaborato saranno esaminate immagini di origine tomografica, in particolare ottenute tramite CBCT di tipo dentale. Esse sono memorizzate sotto forma di *stack*, cioè una pila tridimensionale di immagini (dette *slice* o fette) che in successione l'una all'altra vanno a definire un volume anatomico.

### 1.3.2 Il rumore

Con il termine rumore si intende l'alterazione casuale di alcuni valori durante una misurazione e può disturbare vari tipi di informazione, come un segnale acustico o elettrico. In questa tesi si fa riferimento al rumore (o *noise*) come ad una variazione presente nell'immagine rispetto alla sorgente reale ed è solitamente visibile sotto forma di struttura granulosa che copre la figura. Nell'immagine tomografica acquisita esiste intrinsecamente una quantità aleatoria di rumore e può essere di vari tipi. Il rumore statistico è causato dalla natura quantistica dei raggi X. Ai fini della semplicità, nel modello appena discusso si assumono alcune nozioni che risultano più complesse nella realtà: i raggi X non viaggiano perfettamente in linea retta, non tutti i fotoni del fascio posseggono la stessa quantità di energia e anche la misura dell'attenuazione in base al materiale è di natura statistica. Inoltre, al contatto con il corpo del paziente, i fotoni variano in energia e possibilmente di traiettoria. Questo causa la presenza di fotoni sparpagliati al di fuori del fascio, che se catturati dal detector, possono disturbare la misurazione. Poiché questi tipi di rumore dipendono dalla quantità di fotoni nel fascio, la diminuzione della dose di radiazione riduce la presenza di disturbo. Un altro tipo di noise è quello di origine strutturale e dipende dalle piccole variazioni che possono interessare la misura tramite l'attrezzatura per l'esame tomografico, come il detector. Non è una componente rilevante del rumore totale e solitamente può essere ridotta grazie ad una buona calibrazione degli strumenti. Esiste anche una quantità di disturbo causata dalla natura elettronica dei circuiti del detector. Questo tipo di rumore, detto elettronico, è un segnale costante di bassa frequenza e diventa rilevante quando anche i raggi rilevati dal detector sono

di piccola ampiezza, come nel caso di esami a dose molto ridotta. In particolare, l'operazione di filtering durante l'applicazione dell'algoritmo di ricostruzione FBP intensifica il rumore. Essa aumenta il contrasto tra i valori adiacenti tra loro nel sinogramma, accentuando i rumori assimilati in precedenza.

Un'altra forma di disturbo dell'informazione di origine simile al rumore, e che viene spesso affrontata nella tomografia computerizzata, è la presenza di artefatti. Un artefatto è un elemento visivo anomalo o scorretto che compare nell'immagine e che non rappresenta una struttura reale del soggetto analizzato. A differenza del rumore, l'artefatto non è un fenomeno diffuso di natura aleatoria, ma segue strutture e pattern specifici. Ad esempio, a causa della natura conica delle acquisizioni, nel caso della CBCT si tratta principalmente di striature o coni di luce e possono comparire più facilmente a seguito di alti dosaggi. Il rumore, insieme agli artefatti, può compromettere l'interpretazione visiva e la diagnosi. Si rende quindi necessario un lavoro di post-processing mirato al *denoising*, cioè l'eliminazione degli elementi di disturbo, sia sotto forma di rumore, sia sotto forma di artefatto. A tale scopo si sono rivelate molto promettenti alcune tecniche basate su reti neurali, capaci di apprendere in modo efficace le caratteristiche del rumore e rimuoverlo preservando i dettagli e le strutture anatomiche importanti.

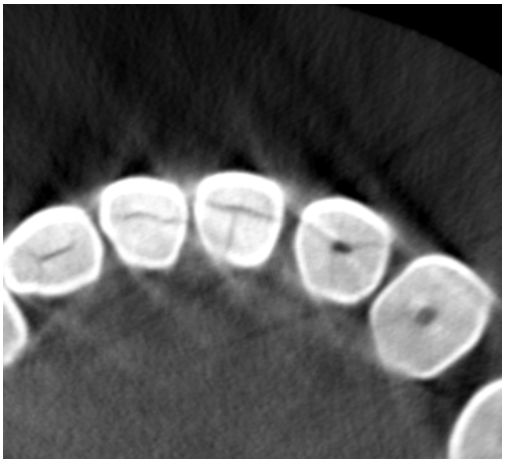


**Figura 1.4:** Le striature bianche in figura sono artefatti tipici della CBCT.

# Capitolo 2

# Reti neurali convoluzionali in 3D

Le reti neurali convoluzionali (o CNN, *Convolutional Neural Networks*) sono un tipo specializzato di rete neurale particolarmente adatto per la classificazione, generazione e modifica di immagini. Una rete neurale è composta da vari livelli, detti *layer*, che includono un primo livello di input, uno o più *hidden layer* responsabili dell'elaborazione dei dati, e infine un livello di output che restituisce il risultato finale. Nel caso in cui vi sia più di un layer nascosto la rete neurale è detta *deep*. Ogni livello è composto da un numero di neuroni artificiali, o nodi, che compiono un'operazione sui dati in input e inviano l'output ad altri nodi nel layer successivo.

## 2.1 Introduzione alle reti neurali convoluzionali

Una rete convoluzionale contiene tre tipi di hidden layer: convoluzionali, di *pooling* e completamente connessi (FC, *Fully Connected*). Il layer convoluzionale è sempre il primo step di elaborazione, seguito poi da altre convoluzioni o da livelli di pooling, mentre i layer FC chiudono la sequenza. Un esempio celebre di rete neurale convoluzionale è la *LeNet* (figura 2.1), ideata nel 1989 da Yann LeCun per il riconoscimento dei caratteri scritti a mano. La rete fu allenata su dataset di codici postali forniti dal U.S Postal Service ed è composta da due cicli di convoluzione e pooling, seguiti da tre livelli FC.

Lo scopo di una convoluzione è l'individuazione dei pattern distintivi e delle principali caratteristiche di un'immagine. Un livello convolutivo restituisce infatti una *feature map*, cioè una versione dell'immagine in input che ne metta in evidenza i 'punti salienti'. Ciò avviene tramite l'applicazione di un filtro convoluzionale detto *kernel*, una matrice che viene sovrapposta ai pixel dell'immagine in input. Ciascuno di questi pixel viene allineato con il centro del filtro e viene calcolata la somma pesata dei valori adiacenti ad esso con i pesi contenuti nel kernel. Il risultato va a sostituire il pixel centrale nella feature map in output dal livello stesso.

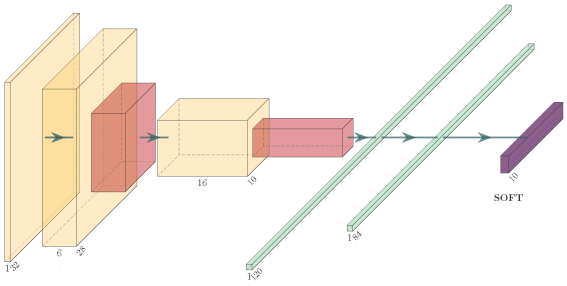



**Figura 2.1:** La *LeNet*, una struttura di rete convoluzionale risalente al 1989. I volumi gialli corrispondono a layer convoluzionali, seguiti dai layer arancioni di pooling e gli ultimi livelli FC.

A vari kernel sono associate operazioni diverse. Ad esempio, un filtro convolutivo in cui tutti i valori sono nulli tranne quello centrale uguale a uno, corrisponde alla funzione identità. Un altro kernel famoso è il filtro Laplaciano per il rilevamento dei bordi: l'applicazione di questa convoluzione rileva le transizioni improvvise di intensità nell'immagine e ne evidenzia i contorni. Lo scopo dell'allenamento è far imparare alla rete quali valori dei pesi sono i migliori affinché l'immagine in input, dopo la convoluzione, sia il più simile possibile all'immagine obiettivo (detta *ground truth*). Nell'esempio in figura 2.2 la convoluzione è stata calcolata traslando il ker-

Laplacian Filter

| | | |
|---|---|---|
| 0 | 1 | 0 |
| 1 | -4 | 1 |
| 0 | 1 | 0 |

Source Image

| | | | | |
|---|---|---|---|---|
| 6 | 5 | 2 | 7 | 6 |
| 9 | 3 | 1 | 5 | 9 |
| 2 | 3 | 3 | 4 | 4 |
| 1 | 2 | 4 | 5 | 1 |
| 5 | 6 | 1 | 9 | 8 |

Feature Map

| | | |
|---|---|---|
| 6 | 9 | 1 |
| -2 | 0 | 1 |
| 6 | -5 | -2 |

**Figura 2.2:** Esempio di applicazione del filtro di Laplace per la *edge detection*.

nel in verticale o in orizzontale di un solo passo: questo valore può essere scelto a seconda dell'obiettivo dell'operazione e prende il nome di *stride*. Si noti anche che i pixel sorgente, quelli posti in corrispondenza con il centro del filtro, appartengono solo alla sezione centrale dell'immagine. Di conseguenza i pixel sui bordi della figura contribuiscono in maniera minore al calcolo della feature map, in quanto vengono considerati un numero minore di volte durante la convoluzione. Per ovviare a

questo problema può essere aggiunto all'immagine un *padding*, ovvero un contorno supplementare di pixel intorno alla figura, solitamente di valore nullo. La presenza o meno del padding, insieme alla dimensione del kernel e alla stride, va a definire la grandezza della feature map in relazione alla figura iniziale. Tale misura è data dalla relazione:

$$\frac{W_{in} + P - K}{S} + 1 = W_{out} \tag{2.1}$$

Dove $W_{in}$ e $W_{out}$ sono rispettivamente le dimensioni dell'input e dell'output, $P$ è il padding, $K$ è la dimensione del kernel (spesso un numero dispari), e $S$ è la stride. I risultati di un'operazione di convoluzione, prima di essere passati al livello successivo, sono sottoposti ad una funzione di attivazione, solitamente una ReLU che permette solo ai valori positivi di essere trasmessi, portando a zero quelli negativi.

A seguito dei livelli di convoluzione si applica un layer di pooling. Si tratta di un'operazione di *downsampling*, cioè che ha il compito di diminuire la dimensione della feature map, in modo da evidenziare le caratteristiche dominanti e migliorare le prestazioni della rete. In particolare l'operazione di pooling suddivide la feature map in piccole zone e procede a selezionare un valore singolo per l'intera area, solitamente il massimo (*Max Pooling*). Infine, i Fully Connected layer hanno il compito di raccogliere tutte le informazioni presenti nelle feature map calcolate e combinarli. Infatti è importante ricordare che ogni livello è composto da diversi nodi, ciascuno di essi avente una visione ristretta dell'immagine in input. Occorre quindi raggruppare gli output dei vari neuroni per ottenere una feature map completa, capace di astrarre le caratteristiche globali dell'immagine.

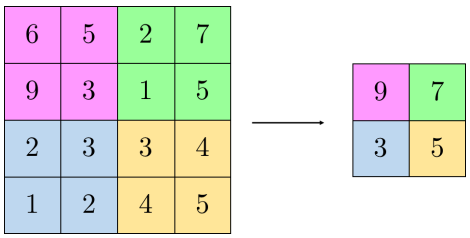



**Figura 2.3:** Esempio di operazione di Max Pooling.

### 2.1.1 Reti convoluzionali in 3D

Fino ad ora si è fatto riferimento a reti capaci di elaborare immagini bidimensionali, ma è possibile predisporre le CNN anche per lo studio di dati in tre dimensioni, come nel caso delle immagini tomografiche. È sufficiente aggiungere una dimensione a molti dei parametri utilizzati per le reti convoluzionali a due dimensioni. Ad esempio, l'input di una rete convoluzionale 2D è la tupla $(w, h, c)$, dove $w$ indica la larghezza dell'input, $h$ la sua altezza e $c$ il numero di canali, cioè la quantità di

valori necessari per descrivere un pixel. Ad esempio, nel caso di immagini in bianco e nero il numero di canali è uno, in quanto ogni pixel è descritto da un solo valore sulla scala dei grigi. Se si lavorasse con immagini colorate RGB, i canali sarebbero 3, poiché il valore di un pixel sarebbe legato alla quantità di rosso, verde e blu. Per effettuare il passaggio ad una rete 3D si aggiunge una dimensione all'input, cioè la profondità $d$ del volume in esame: $(d, w, h, c)$. Nel caso di immagini tomografiche, la terza dimensione è data dal numero di fette sovrapposte nella stack. Per quanto riguarda i layer convoluzionali, anche la dimensione del kernel viene incrementata: l'utilizzo di un filtro in tre dimensioni permette alla rete di osservare i pattern all'interno dei volumi e le relazioni tra slice consecutive. Il calcolo del risultato della convoluzione è eseguito analogamente al caso bidimensionale, cioè sostituendo ciascun voxel con il prodotto scalare tra i valori dei voxel adiacenti in tutte le direzioni e gli elementi del kernel tridimensionale. Anche la stride può assumere un valore multidimensionale, per indicare che il kernel debba essere traslato di passi diversi su direzioni diverse. Ad esempio, non è detto che in un'immagine tomografica la distanza reale tra una slice e l'altra corrisponda alla risoluzione laterale dei pixel di una singola slice. Ovviamente, a causa dell'incremento del numero dei parametri, le reti convoluzionali 3D necessitano di maggiori risorse computazionali. Come si è visto in precedenza, una tecnica per mitigare l'alto costo computazionale ed evitare la dispersione delle informazioni è l'applicazione di un layer di pooling. Lo stesso principio di downsampling si applica anche in tre dimensioni, per cui un solo valore sostituisce una quantità di voxel adiacenti, che può attraversare una o più slice, oppure rimanere bidimensionale se non si vuole ridurre la profondità della pila.

### 2.1.2 Reti U-Net Residuali

Varie architetture di reti neurali convoluzionali sono state proposte con l'obiettivo di eliminare il rumore dalle immagini. Infatti, rispetto ai metodi tradizionali basati su modelli matematici, le CNN offrono un'inferenza veloce e buone prestazioni. Ciononostante queste architetture richiedono grandi quantità di dati specifici per l'allenamento e in campo sanitario raramente è possibile collezionare dataset così abbondanti. Si pensi ad un esame tomografico, in cui si vuole sottoporre il paziente alla minor quantità possibile di radiazioni e per il più breve tempo possibile: ciò implica poche angolazioni di irradiazione e una qualità dell'acquisizione ridotta. Per ovviare il problema, nel 2015 è stato proposto da Ronneberger et al. [RFB15] il concetto di rete U-Net, un tipo di CNN che grazie alla sua architettura è capace di ottenere risultati molto precisi anche con dataset scarsi o aumentati artificialmente. La U-Net è stata originariamente pensata per la segmentazione di immagini mediche, ovvero per l'individuazione e distinzione automaticamente degli oggetti e delle strutture presenti in una figura, ma successivamente è stata adattata al problema del denoising con ottimi risultati.

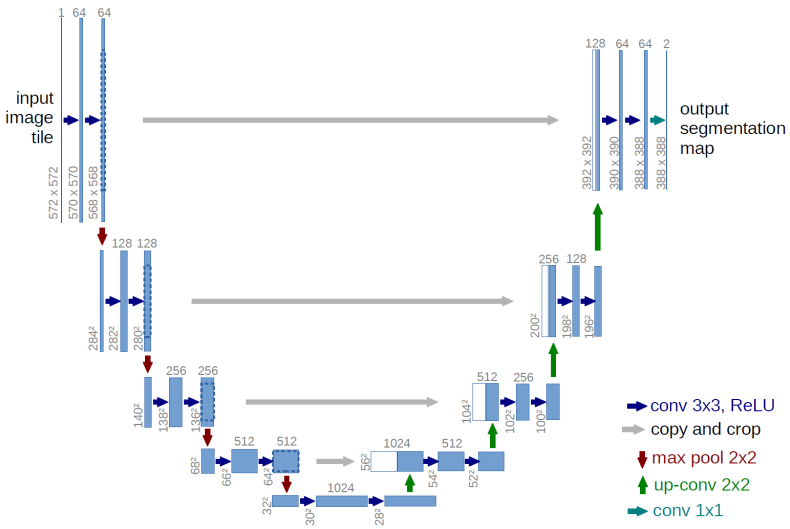



**Figura 2.4:** Architettura originale della U-Net proposta da Ronneberger et al.

L'architettura della rete U-Net è caratterizzata da una struttura simmetrica a "U" composta da due parti principali: un percorso di contrazione (*encoder*) e uno di espansione (*decoder*). La prima fase estrae le caratteristiche significative e il contesto generale dell'immagine, comprimendo la figura attraverso layer di convoluzioni e di pooling. La successiva fase di decoding ricostruisce l'immagine utilizzando operazioni di *upsampling* e concatenando le informazioni ottenute con le feature map risultanti dalla prima fase di contrazione. Questa doppia natura permette alla rete di ottenere risultati molto precisi, unendo le informazioni generali sul contesto, provenienti dall'encoding, con i dettagli sulla localizzazione dei pixel catturati dal percorso di decoding.

Si noti che l'upsampling è l'operazione opposta al downsampling e permette di aumentare la dimensione dell'immagine in input. Viene realizzata nella U-Net tramite layer di deconvoluzione. Come fa intendere il nome, questi livelli compiono il lavoro inverso alla convoluzione, espandendo ogni pixel dell'input in un'area maggiore. L'applicazione di connessioni skip (rappresentate dalle frecce grigie nella figura 2.4) concatena direttamente l'output di ogni livello dell'encoder al layer corrispondente nel decoder. Questa operazione consente al percorso di espansione di recuperare informazioni dettagliate che sono state estratte nella fase di contrazione, migliorando la precisione della ricostruzione dell'immagine denoised.

In lavori successivi[DWCW20] a quello di Ronneberger sono state introdotte le U-Net Residuali. Questo concetto unisce l'architettura U-Net alla tecnica delle connessioni residuali, che permette di allenare reti molto profonde senza perdita di informazioni (problema conosciuto in letteratura come *vanishing gradient*, o scomparsa del gradiente). L'obiettivo di una rete residuale consiste nell'imparare il re-

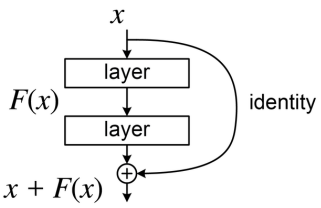



**Figura 2.5:** Applicazione di una connessione residuale all'interno di una rete neurale.

siduo, cioè la differenza tra l'input e l'output desiderato (la ground truth). Questo tipo di rete è molto efficace quando i due tipi di immagini sono molto simili, come nel problema del denoise. Le reti residuali si differenziano da altri tipi di reti neurali perché non tentano di imparare direttamente una mappatura complessa dell'input verso l'output, come avviene invece nell'operazione di segmentazione. Nel caso della rete presa in esame, dedicata al denoising, essa impara la modellazione del rumore stesso, cioè il residuo tra l'immagine di input e l'immagine obiettivo. In particolare, le convoluzioni si concentrano sull'apprendimento delle caratteristiche che identificano e rimuovono il rumore, mentre le connessioni skip preservano i dettagli rilevanti dell'immagine originale. A livello architetturale, la rete residuale è implementata tramite l'uso delle omonime connessioni, il cui principio risiede nel far 'saltare' ad alcuni dati in input un numero a scelta di layer convolutivi, risommandoli successivamente al flusso di informazioni che attraversa la rete (figura 2.5).

Il modello di rete proposto in questa tesi è una ResU-Net.

## 2.2 Analisi del framework utilizzato

In questa sezione viene presentata la struttura del framework su cui è stato condotto il lavoro di ricerca. Si consideri che, come verrà approfondito in seguito, una delle particolarità di tale percorso è l'addestramento della rete neurale su due fasi, una di downsampling e una di super resolution, con l'intenzione di imitare la struttura stessa di una U-Net. Il modello della ResU-Net 3D rimane invariato nelle due fasi, se non per piccole differenze di iperparametri (i valori degli argomenti che prende in input il modello), come il numero di canali e la dimensione dell'input.

La rete è stata allenata su un dataset di training composto da immagini stack tomografiche gentilmente fornite dall'azienda SeeThrough[1]. Esso verrà descritto più dettagliatamente nel prossimo capitolo, ma per ora basta sapere che durante la preparazione del dataset ciascuna immagine viene scomposta in un certo numero di sotto-volumi uguali, detti *patch*.

### 2.2.1 Architettura della ResUNet

Il framework proposto fa uso di una rete di tipo ResU-Net. Questa è stata scelta per combinare la capacità del modello U-Net di costruire una mappatura dei dettagli dell'immagine in input, con l'obiettivo delle connessioni residuali di apprendere la differenza tra input e ground truth.

I parametri in input alla rete appartengono alla tupla

$$(batches, channels, depth, height, width)$$

dove:

- **batches:** letteralmente 'lotto' o 'gruppo', indica il numero di sottoinsiemi in cui è suddiviso il dataset; gli elementi appartenenti a ciascuno di questi insiemi vengono elaborati dal modello nella stessa iterazione di training. Quando il numero di batch è uguale ad uno, tutto il dataset viene elaborato contemporaneamente.

- **channels:** indica il numero dei canali in input alla rete; può variare a seconda della codifica dell'immagine (B&W o RGB) o a seconda dell'architettura: ad esempio, nel percorso di espansione di una U-Net, alcuni livelli di convoluzione possono avere canali di input in più se elaborano sia la feature map del livello precedente sia i dati riportati da una *skip connection*. Nei layer più interni, il numero di canali rappresenta quante sono le caratteristiche (o *feature*) rilevate dalla rete: ogni canale contiene informazioni su un particolare pattern o struttura.

- **depth, height, width:** rispettivamente profondità, altezza e larghezza; rappresentano le tre dimensioni del volume patch in input; nella rete presa in esame, durante il training le misure sono sempre $128 \times 128 \times 128$ voxel.

La figura 2.6 illustra la struttura del modello ResU-Net: le frecce indicano le operazioni effettuate dai layer, mentre i volumi azzurri le feature map risultanti. L'input è elaborato da un doppio strato di livelli convolutivi: il kernel applicato ha dimensione $(3 \times 3 \times 3)$ con padding e stride uguali a 1. La feature map risultante da ciascun layer è sottoposta alla funzione di attivazione non lineare ReLU. Questa esamina i valori del risultato della convoluzione, azzerando quelli negativi e lasciando invariati quelli positivi. A questo punto del modello, il risultato del doppio layer convolutivo è memorizzato per la realizzazione di una skip connection. Esso verrà successivamente reinserito in input nel primo layer convolutivo della fase di decoding. Al risultato intermedio è applicato un livello di Max Pooling con kernel e stride di dimensione $(1 \times 2 \times 2)$. Si noti che le aree in cui viene suddiviso il volume per il downsampling sono bidimensionali, perciò il patch diminuisce in dimensione solo lungo l'altezza e la lunghezza, mantenendo lo stesso numero di slice.

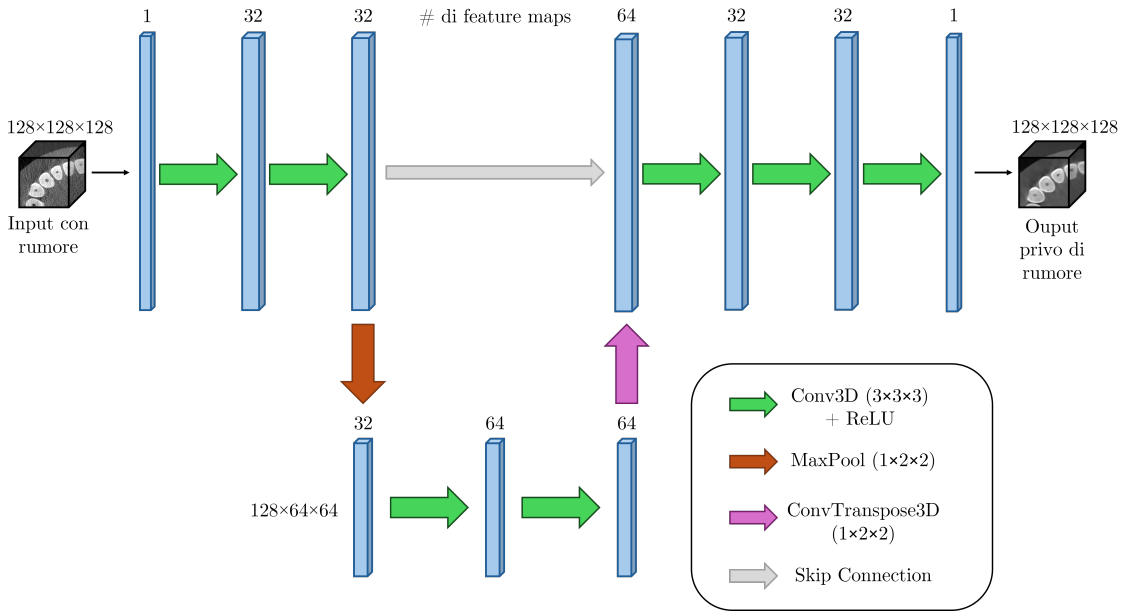



**Figura 2.6:** Modello della U-Net in esame.

Termina così la parte di encoding della U-Net, seguita da un percorso di *bottleneck* (che tradotto significa letteralmente 'a collo di bottiglia'). Questa fase rappresenta un ponte tra i percorsi di encoding e decoding, preparando i dati provenienti dai livelli di contrazione per l'espansione successiva. I layer di bottleneck sono due convoluzioni con kernel uguale a 1, che non vanno a modificare la dimensione delle immagini. Servono a elaborare i dati più astratti dell'intera rete e a catturare con grande precisione le caratteristiche globali dell'input, eliminando ridondanze e dettagli non necessari. In seguito, nella fase di decoding, i patch vengono riportati alla dimensione naturale da un livello di deconvoluzione finalizzato all'upsampling. Nel percorso di espansione, i volumi vengono poi processati da altri due livelli di deconvoluzione e relative ReLU, prima di attraversare un ultimo layer di convoluzione finale. Tale livello ha un ruolo fondamentale, in quanto unisce le feature map intermedie in un unico output con un solo canale. Questa operazione ha il compito di raccogliere tutte le caratteristiche elaborate nei passaggi precedenti e restituire in output il volume patch senza presenza di rumore.

### 2.2.2 Struttura del framework

Come anticipato nella sezione precedente, la struttura della rete neurale presa in esame si sviluppa in due fasi. La prima fase è detta di downsampling. Si consideri il dataset di training, composto da volumi tomografici di due tipi: le immagini di input molto rumorose e le immagini obiettivo, le ground truth, caratterizzate da

un'alta risoluzione e dalla scarsa presenza di rumore (rispettivamente $N$ e $GT$ in figura 2.7). Prima di essere divisi in patch ed elaborati dalla rete, ad entrambi i

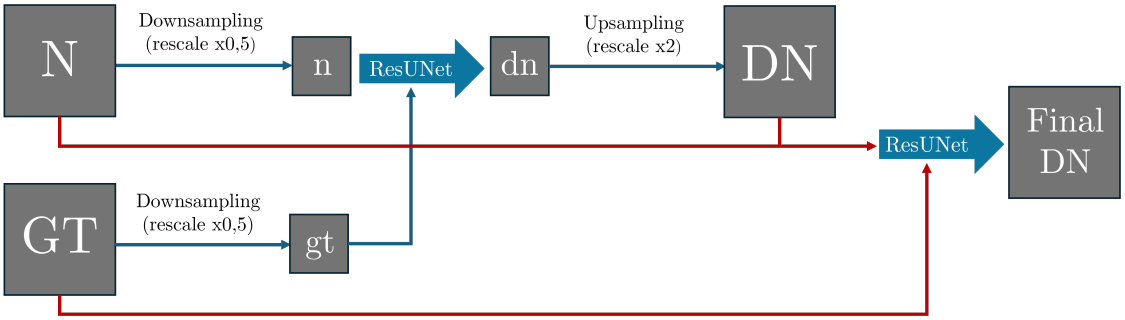



**Figura 2.7:** Struttura complessiva della rete neurale proposta.

dataset viene applicata una funzione di ridimensionamento, che dimezza larghezza e altezza dei volumi, ma mantiene invariato il numero di slice. L'obiettivo è eseguire la U-Net descritta in precedenza su immagini che, essendo ridotte in risoluzione, sono più scarne di dettagli, ma conservano la struttura generale e i contorni. La natura convoluzionale della rete amplifica nell'output l'assenza di particolari e ne sottolinea la struttura globale, rimuovendo il rumore in modo aggressivo. I volumi risultanti, di dimensione ridotta e privi di rumore, sono poi ripristinati alla risoluzione iniziale con un'operazione di *upscale*, che effettua un ulteriore lavoro di sfumatura e rimozione dei particolari dalle immagini. Le immagini tomografiche ora ottenute risultano appiattite e senza dettagli ($DN$ in figura 2.7).

Si passa quindi alla seconda fase del programma, detta di *super resolution*. L'obiettivo è unire le informazioni dettagliate dei volumi originali di input ($N$), con i pattern generali ricavati nel primo stadio ($DN$). Infatti, le immagini originali, seppure molto rumorose, contengono i particolari eliminati durante la prima applicazione della rete. Al contrario, i volumi denoised appena ottenuti guidano la rete nell'imparare cosa non è rumore, cioè quali strutture e contorni appartengono correttamente alle immagini. Il risultato ($Final\ DN$) è un volume che combina la ricchezza di dettagli delle immagini originali con la chiarezza strutturale ottenuta dai volumi denoised, restituendo immagini prive di rumore e artefatti.

### 2.2.3 Iperparametri

Con il termine iperparametri si indicano tutti i valori ed argomenti che configurano la rete prende in input e ne regolano il comportamento.

### Funzione di Loss

La funzione di loss è utilizzata dalla rete per calcolare, durante e dopo l'allenamento, quanto l'immagine denoised sia distante dall'immagine obiettivo, con il fine di minimizzare tale differenza. Nel training della rete patch e nell'analisi dei seguenti risultati è stata usata la funzione *MSELoss* [Con23] per il calcolo dell'errore quadratico medio (o MSE, *Mean Squared Error*), fornita dal framework di PyTorch. Questa funzione è molto utilizzata nelle tecniche di predizione di dati come le reti neurali e viene scelta per la sua semplicità e facilità di calcolo. Dati due tensori $x$ e $y$ di dimensione variabile ma con un totale di $N$ elementi, la funzione MSELoss calcola la seguente funzione:

$$l(x, y) = L = \{l_1, \ldots, l_N\}^T \quad con \quad l_i = (x_i - y_i)^2 \tag{2.2}$$

In particolare, è impostata la modalità *mean* della funzione, per cui MSELoss restituisce la media dell'errore calcolato in 2.2, cioè:

$$l(x, y) = mean(L) = \frac{1}{N} \sum_{i=1}^{N} l_i = \frac{1}{N} \sum_{i=1}^{N} (x_i - y_i)^2 \tag{2.3}$$

### Epoche

Con il termine *epoca* si intende un ciclo completo di addestramento della rete, durante il quale il modello elabora i batch di dati. Un batch viene analizzato durante un'iterazione, alla fine della quale la rete calcola l'errore e aggiorna i pesi di conseguenza. Una volta analizzati tutti i batch in cui è stato suddiviso il dataset, l'epoca termina. Nel caso della rete patch, il numero di batch è impostato a 1, per cui l'intero dataset di training è analizzato in una sola iterazione per epoca. Sono necessarie più epoche per l'addestramento efficace di una rete, in particolare entrambe le ResUNet della rete patch sono state addestrate per 25 epoche. É consigliabile non esagerare con il numero di epoche, in quanto più a lungo una rete viene allenata, maggiore è il rischio che essa sia soggetta ad *overfitting*. Questo termine indica la spiacevole situazione in cui la rete impara con troppa fedeltà i dati di training e manca di capacità di generalizzazione. Ciò porta ad ottime prestazioni sul dataset di training, ma peggiori risultati sul dataset di testing, contenente immagini che la rete non ha mai visto.

Per quanto riguarda il training della rete patch, è possibile osservare nel grafico 2.8 come, all'aumentare delle epoche, diminuisca l'errore quadratico medio tra la previsione fatta dalla rete e le ground truth, sia per la rete di downsampling che per la super resolution. In particolare, si può notare come l'errore della fase di super resolution sia sempre minore di quello del downsampling.

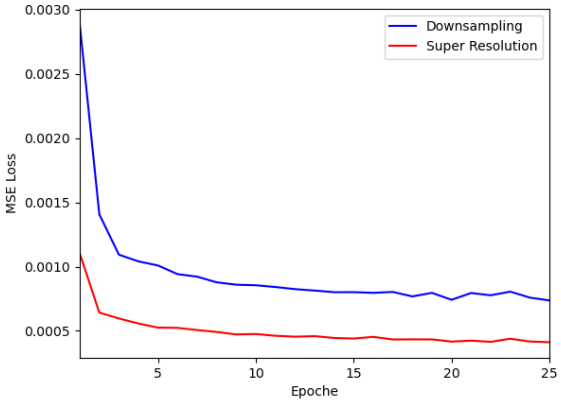



**Figura 2.8:** Confronto sulla loss MSE durante i training della rete di downsampling e della rete super resolution.

### Learning Rate

Il learning rate, o tasso di apprendimento, è la dimensione del passo compiuto ad ogni iterazione dall'algoritmo di ottimizzazione, usato nella ricerca del minimo della funzione di loss. In altre parole, il learning rate decide quanto velocemente i pesi nei kernel del modello vengono aggiornati in risposta agli errori commessi. Nel caso in cui il tasso di apprendimento sia troppo basso il calcolo del minimo impiegherebbe troppo tempo, mentre con un learning rate troppo alto l'algoritmo di ottimizzazione rischierebbe di non convergere correttamente. Per l'allenamento della rete patch, è stato scelto un learning rate di $1 \times 10^{-4}$. L'algoritmo di ottimizzazione utilizzato è *Adam* (*Adaptive Moment Estimation*), che aggiorna i pesi della rete in modo iterativo. Adam può essere visto come una variante avanzata della discesa del gradiente stocastico ed è ampiamente utilizzato nell'addestramento di reti neurali profonde, grazie alla sua efficienza e stabilità anche in presenza di dati rumorosi.

# Capitolo 3

# Risultati Numerici

## 3.1 Dataset

I dataset utilizzati sono formati da immagini tomografiche dentali, catturate da una tomografia computerizzata a *cone beam* e ricostruite con l'algoritmo di Feldkamp-David-Kress. Le scansioni sono state effettuate su tre fantocci, prima di tutto perché questi non generano artefatti da movimento, ma soprattutto perché è possibile inondarli di qualsiasi quantità di radiazione senza conseguenze per la salute dei pazienti. I volumi utilizzati non rappresentano una visione complessiva dell'arcata, come è solito per le tomografie dentali, ma si tratta di *patch*, ritagli dei volumi interi che si concentrano sulle aree contenenti i denti. Non bisogna confondere questo tipo di patch, pochi e più grandi, con i sotto-volumi patch in cui la rete suddivide questi ultimi per l'elaborazione del dataset. L'approccio basato su patch è stato scelto per vari motivi e rappresenta, insieme al doppio uso di una ResUNet, la caratteristica fondamentale della rete in esame. La scelta è stata fatta con lo scopo di aumentare la dimensione del dataset di training e di generare varietà tra le forme e gli oggetti presenti in ciascun volume. Infatti, un dataset più vario e numeroso migliora le prestazioni della rete e la aiuta a imparare rappresentazioni e pattern più generali, riducendo il rischio di overfitting. Un altro vantaggio non indifferente è la gestione della memoria: volumi grandi occupano molto spazio e suddividerli in patch, prima e durante l'allenamento, aiuta ad alleggerire l'elaborazione.

### 3.1.1 Suddivisione dei volumi in patch

Come anticipato nel capitolo precedente, la rete durante l'allenamento di entrambe le fasi non utilizza il dataset di training così come è, ma suddivide i volumi in piccoli patch di dimensione $128 \times 128 \times 128$ voxel. Questi sotto-volumi hanno lo stesso obiettivo dei patch più grandi descritti sopra: allargare e variare il dataset di training. I sotto-volumi hanno sempre la stessa dimensione, sia nella fase di down-

sampling che di super resolution: nel primo caso, ogni volume del dataset, essendo ridimensionato, contribuirà con un numero inferiore di patch rispetto alla seconda fase.

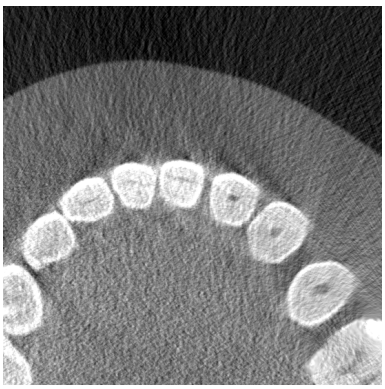
I patch non sono disgiunti, infatti il parametro *patch stride* indica ogni quanto, nelle tre dimensioni, inizia un nuovo sotto-volume di *patch size* uguale a 128. La stride è stata impostata a 64, il che significa che ciascun patch si sovrappone al successivo su ogni asse. Ciò permette di aumentare il numero di dati elaborati e di includere a pieno anche i bordi dei volumi, migliorando la continuità tra i patch. Il numero di patch per volume è calcolabile sapendo le sue dimensioni e i parametri stride e size. Ad esempio, lungo la larghezza $x$ i volumi sono:

$$\text{numPatchX} = \left\lfloor \frac{\text{totalSizeX} - \text{patchSize}}{\text{patchStride}} \right\rfloor + 1 \tag{3.1}$$

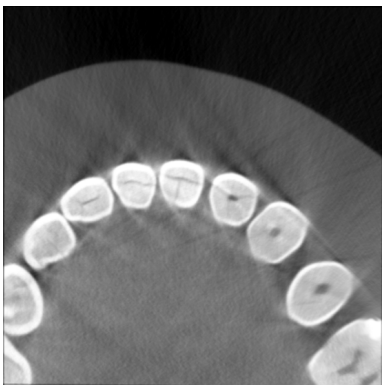
I calcoli per le rimanenti dimensioni sono analoghi. Ad esempio, dato il volume di training di dimensioni $(579, 514, 494)$, il numero di patch per le dimensioni $z$, $y$ e $x$ è rispettivamente 8, 7 e 6, per un totale di 336 patch.

### 3.1.2 Dataset di training

Come anticipato nel capitolo precedente, il dataset di training è composto da due tipi di immagini: le immagini in input rumorose e le immagini obiettivo, le ground truth. Il primo tipo di volume è stato ricavato applicando una normale dose di radiazioni durante la CBCT, mentre il secondo è stato ottenuto aumentando di 100 volte la dose standard. Le ground truth risultano essere delle immagini molto precise e prive di rumore granulare, seppure disturbate dagli artefatti classici di una cone beam, rafforzati dalla potenza dei raggi X. Ad esempio, in 3.1b sono più marcate le striature coniche. Ciononostante, il volume è ricco di dettagli e texture essenziali, che la rete mira a replicare.



**(a)** Immagine rumorosa in input

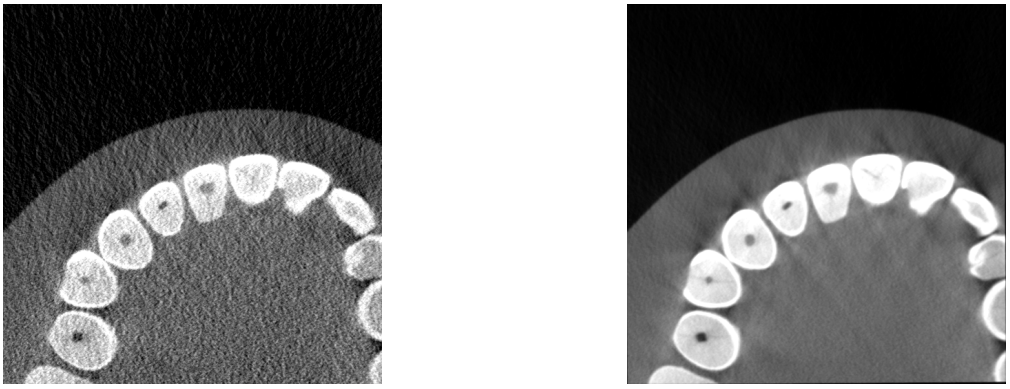


**(b)** Immagine obiettivo (o ground truth)

**Figura 3.1:** Confronto tra la stessa immagine nei due dataset di training.

### 3.1.3 Dataset di test

Dai patch del dataset appena descritto, un volume è stato escluso dallo stesso, in modo da poter ottenere, durante il testing, risultati su immagini che la rete non ha mai visto prima. Al fine di calcolare la funzione di loss, anche il volume di test ha una corrispondente ground truth.



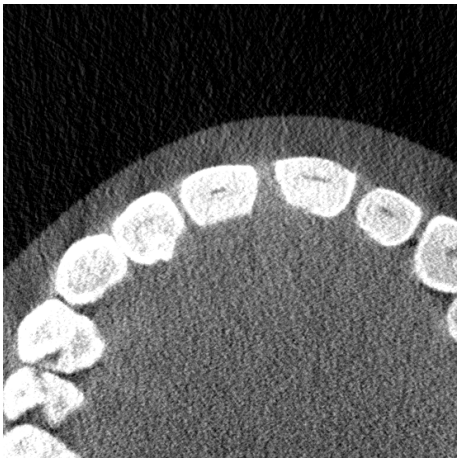
**(a)** Immagine di test rumorosa **(b)** Immagine obiettivo di test

**Figura 3.2:** Dataset di test.

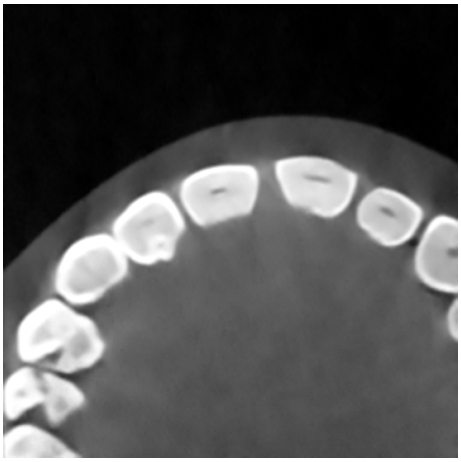
## 3.2 Risultati

Si analizzano ora i risultati della rete patch sul dataset di test. Poiché si tratta di un volume di dimensione $(475, 550, 550)$, verranno scelte delle slice significative per l'analisi visiva del risultato. Lo scopo della rete è la rimozione di rumore e artefatti causati dall'acquisizione ed elaborazione dei volumi, mantenendo un'alta qualità e risoluzione dei dettagli, essenziali per la diagnosi. Si noti che per la visualizzazione e modifica delle immagini che seguono è stato usato il software open-source *ImageJ* del National Institute of Health statunitense.

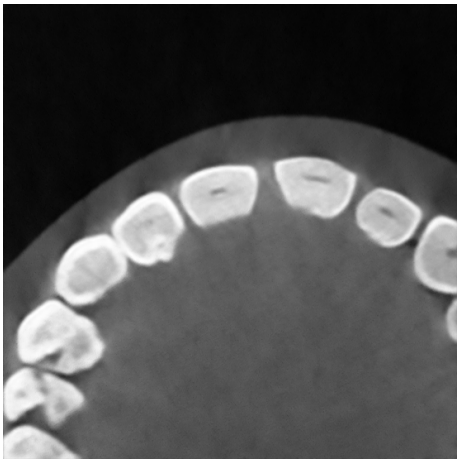
Procedendo ad una prima ispezione qualitativa visiva della figura 3.3, risalta subito all'occhio il filtro granuloso dell'immagine originale, assente nelle immagini elaborate. L'immagine intermedia 3.3b risulta totalmente priva di rumore, ma anche di dettagli. Si può notare ad esempio l'assenza di texture e particolari nei denti, come le varie sfumature di grigio che ne compongono la parte interna, che ritornano invece nell'immagine finale 3.3c. Il risultato ottenuto si presenta immediatamente più definito, persino rispetto all'immagine obiettivo, che a causa della forte intensità del raggio X utilizzato presenta artefatti e un'alta luminosità che va a coprire i dettagli. Ad esempio, il terzo dente da sinistra nella ground truth 3.3d compare quasi completamente bianco. Nonostante l'elaborato offra un migliore bilanciamento del contrasto, il confronto con la ground truth è cruciale per garantire che i risul-



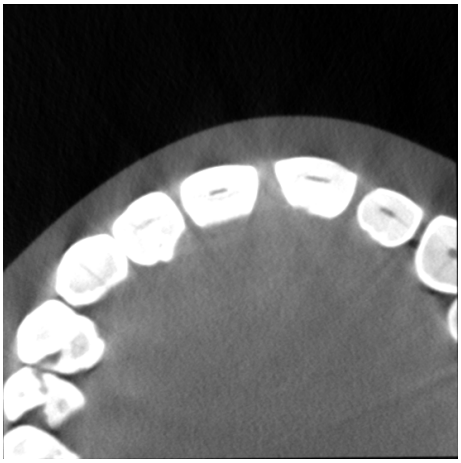
**(a)** Immagine di test rumorosa



**(b)** Immagine intermedia dopo il downsampling



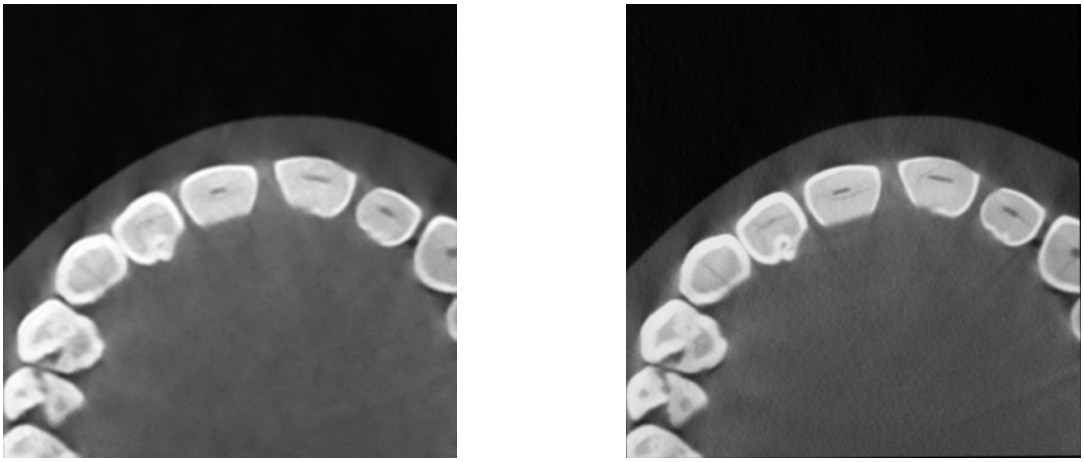
**(c)** Immagine finale



**(d)** Immagine obiettivo

**Figura 3.3:** Confronto qualitativo tra le varie fasi sull'immagine di test.

tati siano coerenti con il riferimento stabilito. Basta utilizzare uno strumento per l'aggiustamento del contrasto e il confronto con l'immagine obiettivo diventa più interessante. Si può visivamente notare dalla figura 3.4 la somiglianza tra il risultato finale della rete patch e la ground truth a contrasto ridotto, evidenziando la qualità della ricostruzione compiuta dalla rete.

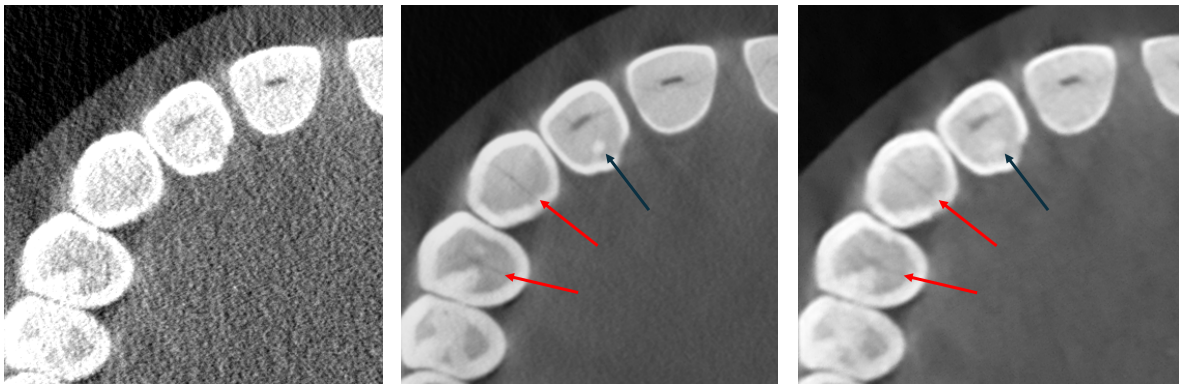


**(a)** Immagine finale

**(b)** Immagine obiettivo con aggiustamento di contrasto

**Figura 3.4:** Confronto qualitativo tra la ground truth e l'immagine elaborata.

La rete non ha solo effettuato l'operazione di rimozione del rumore, che avviene già nella prima fase di downsampling, ma ha anche recuperato i dettagli, altrimenti invisibili nel rumoroso volume di input. Ad esempio, nella figura 3.5, si può notare la macchia nel quarto dente da sinistra (indicata dalla freccia blu in figura), che nonostante compaia più risoluta nella ground truth, risulta comunque ben visibile anche nell'immagine finale. Questo tipo di macchie, presenti anche in altri denti in figura, insieme alle linee nel secondo e terzo dente (indicate dalle frecce rosse), sono perse nel rumore dell'immagine in input.



**(a)** Dettaglio immagine in input

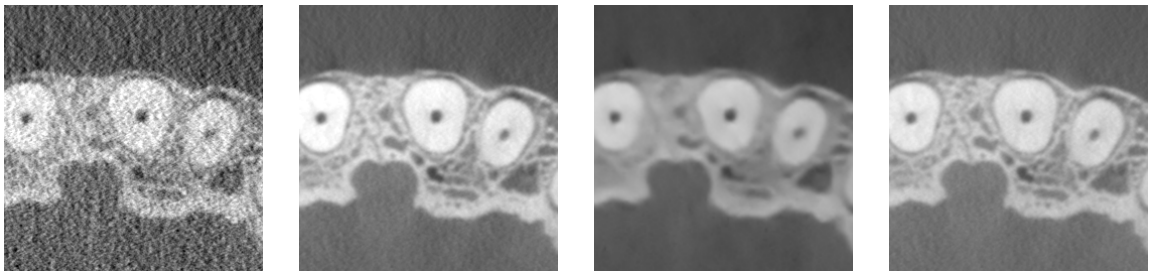
**(b)** Dettaglio immagine obiettivo con aggiustamento di contrasto

**(c)** Dettaglio immagine finale

**Figura 3.5:** Recupero dei dettagli tra la ground truth e l'immagine elaborata.

Un altro tipo di confronto sfrutta la possibilità data da ImageJ di sottrarre le immagini tra di loro, permettendo di ottenere un risultato visivo di quella che

è la differenza tra i valori dei pixel. Date le prime quattro immagini in figura 3.6, raffiguranti un dettaglio dell'arcata dentale, si procede nella riga successiva a calcolare la differenza tra il risultato finale e le altre immagini. L'immagine 3.6e, cioè la sottrazione tra immagine in input e output finale, raffigura il filtro granuloso del rumore che è stato rimosso, ma nessuna forma o pattern dentale è discernibile al suo interno, a prova che non sono andate perse informazioni chiave sull'aspetto generale dell'immagine. Invece la figura 3.6g, rappresentante la differenza tra i risultati intermedio e finale, mette in risalto i dettagli che la super resolution ha in più rispetto al processo di downsampling. Il risultato migliore è sicuramente la figura 3.6f, cioè la sottrazione tra la ground truth e il volume finale della super resolution, dato che la differenza è completamente nera, cioè non esistono pixel di valore diverso tra le due aree in esame.



**(a)** Dettaglio immagine in input



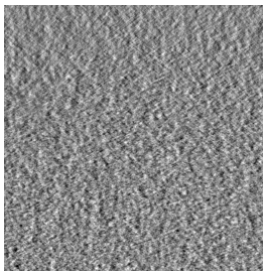
**(b)** Dettaglio immagine obiettivo



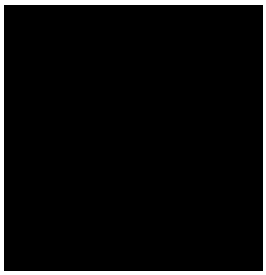
**(c)** Dettaglio immagine intermedia



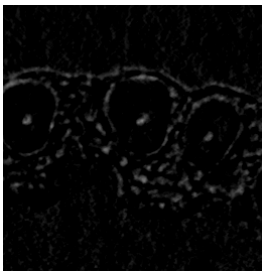
**(d)** Dettaglio immagine finale



**(e)** Sottrazione tra input e super resolution



**(f)** Sottrazione tra ground truth e super resolution



**(g)** Sottrazione tra immagine intermedia e super resolution

**Figura 3.6:** Calcolo della differenza tra le varie immagini e il risultato finale. L'operazione di sottrazione è stata eseguita con il software *ImageJ*.

Si procede ora ad una analisi quantitativa dei risultati sul volume di test. Uno dei metodi possibili è attraverso lo studio dell'istogramma, il grafico che mostra la distribuzione dei valori dei pixel in un'area o in un segmento selezionati nell'immagine. Per brevità, si considerano solo gli istogrammi della ground truth e del risultato finale. Il segmento preso in esame è rappresentato in figura 3.7 su entrambe le immagini, ed è stato scelto per la varietà di valori e dettagli sul suo cammino.

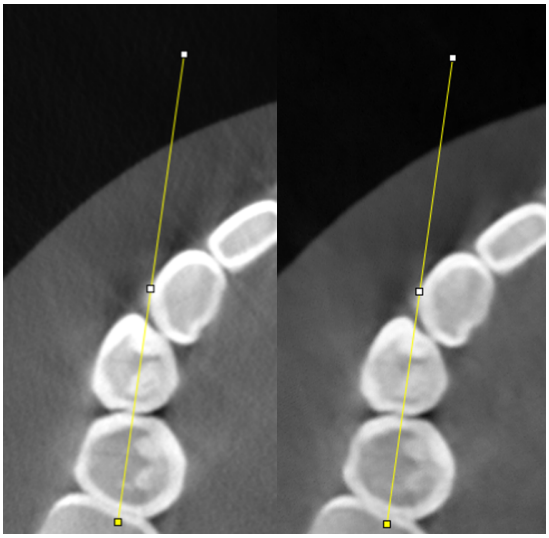
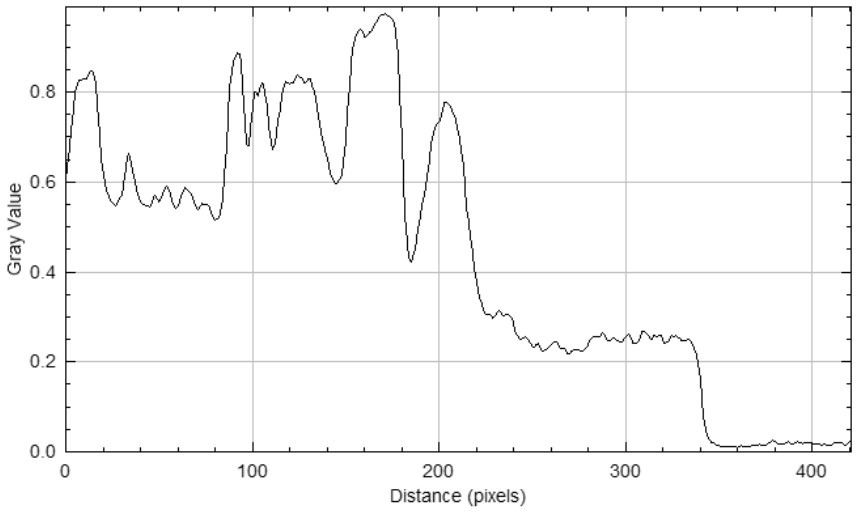
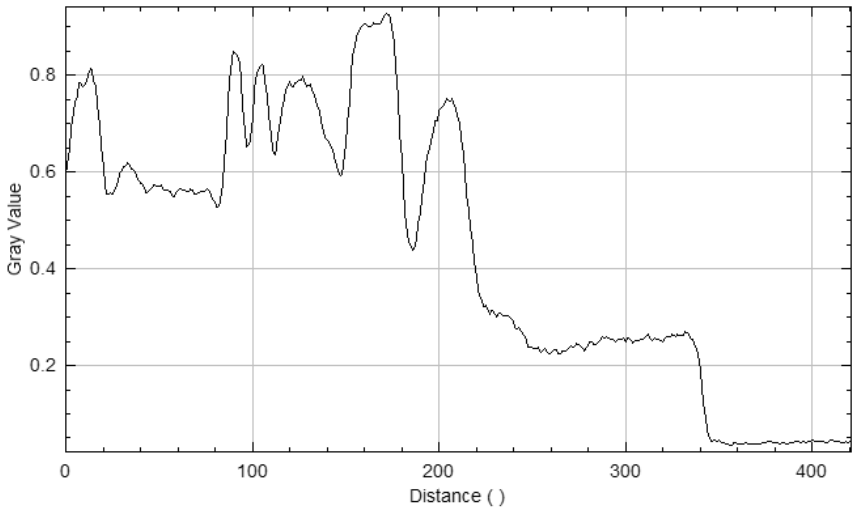


**Figura 3.7:** Area selezionata per il confronto sull'immagine obiettivo (sinistra) e sul risultato finale (destra).

I due istogrammi risultano subito molto simili, sintomo che la rete ha eseguito un buon lavoro. Si possono osservare solo un paio di differenze. La prima riguarda il tratto di segmento che attraversa la zona di tessuto molle in figura (rappresentata in 3.8 dalla gamma dei valori tra i 250 e 350 pixel di distanza circa), dove si può notare un appiattimento del grafico dovuto alla rimozione del rumore e degli artefatti conici visibili nella ground truth. La seconda differenza riguarda l'area interna al primo dente attraversato dal segmento (in figura la zona di bassa intensità tra i 25 e 75 pixel circa), che presenta un abbassamento e allargamento dei picchi del grafico, dovuto alla minor risoluzione del risultato della rete rispetto alla ground truth.




**(a)** Profilo del grafico dell'istogramma sull'immagine ground truth




**(b)** Profilo del grafico dell'istogramma sull'immagine risultato

**Figura 3.8**

### 3.2.1 Confronto con una rete End-to-End

Per affermare l'efficacia del framework proposto e l'utilità del suo percorso a doppia ResUNet, è stato fatto un confronto con una rete end-to-end. Con questo termine si indica una rete semplice, che sfrutta lo stesso modello di ResUNet, senza applicare però alcun downsampling o tecnica di super resolution. Il dataset

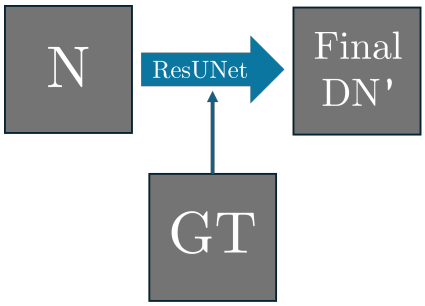



**Figura 3.9:** Struttura della rete end-to-end.

di training rimane invariato e viene diviso in sotto-volumi come nella rete patch. Per chiarezza espositiva, nomineremo *DN* i risultati finali della rete patch e *DN'* i risultati della rete end-to-end. Quest'ultima è stata addestrata con gli stessi iperparametri della rete patch, quindi stessa funzione di loss, learning rate e numero di epoche.

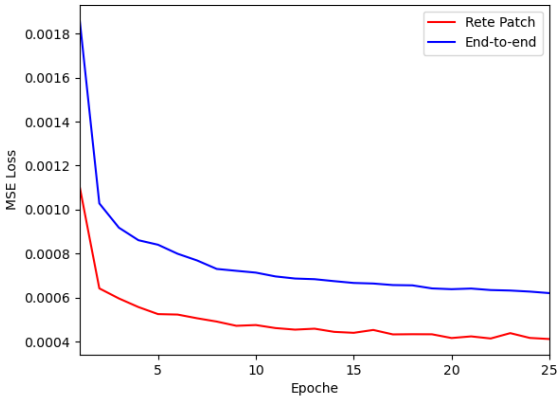
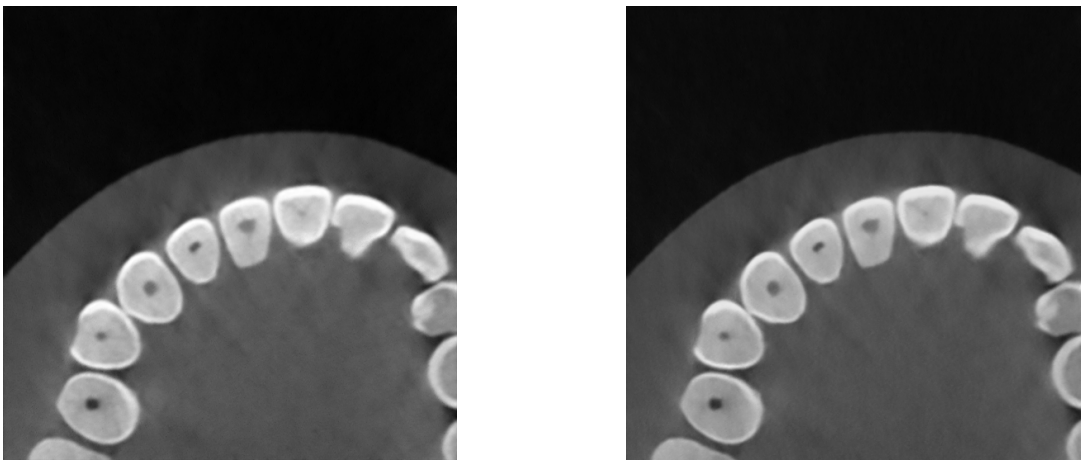



**Figura 3.10:** Confronto tra la rete Patch e la più semplice end-to-end: MSE Loss per epoca sul dataset di training.

Eseguendo un confronto qualitativo si può notare l'assenza di alcuni dettagli nell'immagine DN' che invece sono visibili nella N, come le linee sul terzo e settimo dente da sinistra, già leggere nell'output della rete patch ma invisibili nel risultato

della end-to-end. In generale il denoising e il delineamento dei bordi risulta meno aggressivo nell'immagine DN, il che permette la visione di più dettagli.



**(a)** Immagine finale ottenuta con rete patch (DN)

**(b)** Immagine ottenuta con rete end-to-end (DN')

**Figura 3.11:** Confronto qualitativo tra rete patch e end-to-end sull'immagine di test.

Si può procedere anche in questo caso alla sottrazione tra le due immagini per sottolineare i particolari diversi tra i due output.

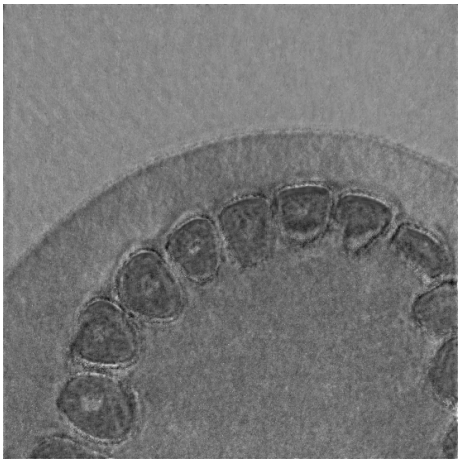


**Figura 3.12:** Sottrazione tra l'immagine end-to-end e il risultato della rete patch.

# Conclusioni

In questa tesi, è stato analizzato un framework innovativo basato su reti neurali convoluzionali per il denoising di immagini mediche, con l'obiettivo di migliorare la qualità delle immagini e, di conseguenza, la loro interpretabilità diagnostica. I risultati sulle immagini di training e di testing confermano l'efficacia del framework proposto nella rimozione del rumore, con una significativa capacità di conservare i dettagli utili per la diagnosi rispetto ad altri modelli. In particolare, il modello si è rivelato più preciso rispetto all'approccio end-to-end classico, evidenziando il suo potenziale anche in assenza di dataset abbondanti. Tuttavia il modello presenta alcune limitazioni operative, come la necessità di ogni volume di attraversare entrambe le reti prima di poter essere ricostruito, rallentando il processo complessivo. Nonostante queste sfide, il nuovo framework proposto risulta particolarmente efficace al denoising delle immagini mediche, offrendo una base promettente per futuri miglioramenti. Tra le possibili direzioni di ricerca, si potrebbero esplorare strategie per ottimizzare il processo di allenamento dal punto di vista computazionale. Inoltre, la rete si apre ad un'ulteriore validazione su dataset di testing più ampi e diversificati, composti ad esempio da volumi di pazienti reali.

# Riferimenti bibliografici


[1] See through s.r.l. `https://www.seethrough.one/`.

[Che17] Omar Chehaimi. Parallelizzazione dell'algoritmo di ricostruzione di feldkamp-davis-kress per architetture low-power di tipo system-on-chip. Master's thesis, Alma Mater Studiorum, Bologna, 2017. Laurea Magistrale in Physics.

[Con23] PyTorch Contributors. Mseloss - pytorch 2.5 documentation. `https://pytorch.org/docs/stable/generated/torch.nn.MSELoss.html`, 2023. Accessed: 2024-12-06.

[DWCW20] Foivos Diakogiannis, Francois Waldner, Peter Caccetta, and Chen Wu. ResUNet-a: A deep learning framework for semantic segmentation of remotely sensed data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 16:94–114, 02 2020.

[Hsi09] Jiang Hsieh. *Computed Tomography: Principles, Design, Artifacts, and Recent Advances*. Wiley, 2nd edition, 2009.

[LF10] Emanuele Neri Lorenzo Faggioni, Fabio Paolicchi. *Elementi di tomografia computerizzata*. Springer, 2010.

[Mad21] Samaya Madhavan. Introduction to convolutional neural networks. `https://developer.ibm.com/articles/introduction-to-convolutional-neural-networks/`, 12 July 2021. [Accessed 01-12-2024].

[MZGG09] Hui Miao, Hui-juan Zhao, Feng Gao, and Shao-run Gong. Implementation of fdk reconstruction algorithm in cone-beam ct based on the 3d shepp-logan model. In *2009 2nd International Conference on Biomedical Engineering and Informatics*, pages 1–5, 2009.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[SKT+19] Rebecca Schofield, L. King, U. Tayal, I. Castellano, J. Stirrup, François Pontana, James Earls, and Edward Nicol. Image reconstruction: Part 1 – understanding filtered back projection, noise and image acquisition. *Journal of Cardiovascular Computed Tomography*, 14, 04 2019.

[VE17] Elluru Venkatesh and Snehal Elluru. Cone beam computed tomography: basics and applications in dentistry. *Journal of Istanbul University Faculty of Dentistry*, 51, 11 2017.

[Wan23] Chuqi Wang. A review on 3d convolutional neural network. In *2023 IEEE 3rd International Conference on Power, Electronics and Computer Applications (ICPECA)*, pages 1204–1208, 2023.