

UNIVERSITÀ DEGLI STUDI DI CASSINO E DEL LAZIO  
MERIDIONALE

Dipartimento di Ingegneria Elettrica e dell'Informazione



**Corso di Laurea Magistrale in Ingegneria Informatica**

# **Segmentazione del seno in immagini MRI**

**Relatore:** Alessandro Bria

**Correlatore:** Marco Cantone

**Studente:** Achille Cannavale

**Matricola:** 58721

# Indice

<b>1</b>	<b>Introduzione</b>	<b>4</b>
1.1	Contesto . . . . .	4
1.2	Attività di Tirocinio e obiettivi . . . . .	5
1.3	Strumenti Utilizzati . . . . .	5
<b>2</b>	<b>Background e Stato dell'Arte</b>	<b>8</b>
2.1	L'Evoluzione delle Tecniche di Segmentazione . . . . .	8
2.2	Architettura U-Net per la Segmentazione di Immagini Mediche . . . . .	10
2.3	Strumenti e Dataset Moderni . . . . .	11
2.4	Stato dell'Arte nella Segmentazione del Seno in MRI 3D . . . . .	11
<b>3</b>	<b>Dataset: Duke Breast Cancer MRI</b>	<b>13</b>
3.1	Composizione del Dataset . . . . .	13
3.1.1	Annotazioni e Ground Truth . . . . .	14
3.1.2	Preprocessing e Integrazione con MONAI . . . . .	14
3.1.3	Suddivisione del Dataset . . . . .	15
<b>4</b>	<b>Configurazione Sperimentale</b>	<b>16</b>
4.1	Il file di configurazione YAML . . . . .	17
4.1.1	Struttura generale . . . . .	17
4.1.2	Configurazione del training . . . . .	18
4.1.3	Trasformazioni dei dati . . . . .	18
4.1.4	Definizione del modello . . . . .	18
4.1.5	Ottimizzazione e loss . . . . .	19
4.1.6	Post-processing e valutazione . . . . .	19
4.2	Gestione del Logging . . . . .	20
<b>5</b>	<b>Eperimenti</b>	<b>22</b>
5.1	Introduzione . . . . .	22
5.1.1	Nota sulle Valutazioni . . . . .	23
5.2	Impostazione degli Eperimenti . . . . .	23
5.3	Evoluzione Architetturale e Tuning degli Iperparametri . . . . .	23
5.4	Strategia Alternativa: Approccio 2D e Architettura SliceUNet . . . . .	26
5.5	Ritorno all'approccio 3D: Affinamento e Validazione . . . . .	27

5.6	<b>Nuovo approccio di Valutazione</b>	27
5.6.1	L'Attention U-Net	28
5.7	<b>Esperimento Finale</b>	29
5.7.1	Post-processing morfologico: operazione di Closing	31
<b>6</b>	<b>Conclusioni</b>	<b>33</b>
6.1	Sintesi del Percorso Sperimentale	33
6.2	Risultato Finale e Implicazioni Cliniche	34
6.3	Possibili Downstream Task	34
6.4	Prospettive di Sviluppo Futuro	34
6.5	Riflessione Conclusiva	35
	<b>Bibliografia</b>	<b>40</b>

# Capitolo 1

## Introduzione

### Contents

---

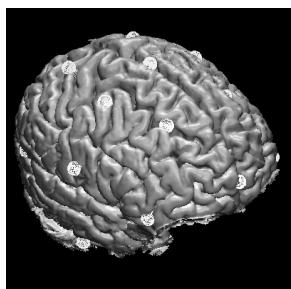
1.1	Contesto . . . . .	4
1.2	Attività di Tirocinio e obiettivi . . . . .	5
1.3	Strumenti Utilizzati . . . . .	5

---

### 1.1 Contesto

La segmentazione di strutture anatomiche in immagini mediche 3D rappresenta un task importante nell'ambito clinico e diagnostico. In particolare la segmentazione del seno da immagini **MRI (Magnetic Resonance Imaging) 3D** è un task complesso, ma essenziale per applicazioni come la pianificazione di interventi chirurgici o l'analisi di anomalie tessutali.

In passato, la segmentazione del seno in immagini MRI 3D veniva effettuata principalmente con **metodi classici**, spesso semi-automatici o manuali, basati su:



**Figura 1.1:** Immagine 3D di un cervello catturata tramite una MRI. [1]

- **Approcci a soglia** (thresholding) e **region-growing**, che sfruttavano differenze di intensità tra tessuti ma richiedevano regolazioni manuali e fallivano in presenza di rumore o basso contrasto.
- **Deformable models** (es.: Active Contours) e **algoritmi a clustering** (es.: k-means), sensibili all'inizializzazione e poco robusti alla variabilità anatomica.
- Metodi **atlas-based**, che allineavano immagini a template pre-annotati, limitati però dalla diversità inter-paziente.

Con l'avvento del **deep learning**, in particolar modo delle **reti convoluzionali** e di architetture come **U-Net 3D** [6], la segmentazione del seno ha raggiunto livelli di accuratezza più alti rispetto al passato.

## 1.2 Attività di Tirocinio e obiettivi

L'attività si è inserita nel contesto di un progetto di ricerca mirato allo sviluppo e alla valutazione di modelli di segmentazione automatica per immagini in MRI del seno, con l'obiettivo di migliorare la diagnosi precoce e la pianificazione terapeutica.

L'obiettivo principale del tirocinio è stato quello di sperimentare e ottimizzare pipeline basate su **deep learning** per la segmentazione semantica tridimensionale, utilizzando framework open-source moderni come **MONAI** (Medical Open Network for AI) [5] e **PyTorch**.

Le principali attività svolte sono state:

- **Pre-processing** dei dati in formato **DICOM**, con conversione in formato leggibile da **MONAI** e normalizzazione delle immagini.
- Composizione e validazione di un dataset bilanciato, con partizionamento in **training**, **validation** e **test set**.
- Studio e implementazione di modelli di segmentazione basati su architetture tra cui **UNet** e **AttentionUNet**.
- Configurazione degli esperimenti, **tuning degli iperparametri** e addestramento dei modelli in ambiente GPU.
- Valutazione delle prestazioni mediante metriche standard come **Dice Score**



**Figura 1.2:**  
Logo Py-  
Torch.  
[10]

Sono state inoltre affrontate e risolte diverse problematiche tecniche legate alla gestione dei metadati **DICOM**, alla compatibilità tra i formati di input, e alla gestione efficiente della memoria in fase di training. L'intero lavoro è stato documentato e riproducibile mediante script **Python** e configurazioni **YAML**.

## 1.3 Strumenti Utilizzati

Sono stati impiegati diversi strumenti software e librerie fondamentali per la gestione, il pre-processing e l'elaborazione di immagini medicali, nonché per lo sviluppo e il training di modelli di deep learning.

- Il linguaggio principale di lavoro è stato **Python**, scelto per la sua flessibilità e per l'ampio ecosistema di librerie scientifiche.
- Per la manipolazione delle immagini **DICOM** è stata utilizzata la libreria **MONAI** (**Medical Open Network for AI**), un framework open-source basato su **PyTorch** e

progettato specificamente per applicazioni di imaging medicale. MONAI ha permesso di gestire agevolmente il caricamento dei dati, le trasformazioni e la normalizzazione delle immagini, grazie a un sistema modulare di trasformazioni componibili.

- Il framework di deep learning impiegato è stato **PyTorch**, scelto per la sua semplicità d'uso, il supporto attivo della community e le sue prestazioni elevate, specialmente in combinazione con l'accelerazione GPU fornita da **CUDA**.
- Lo sviluppo del codice è stato realizzato su una macchina **Linux**, con accesso remoto e con l'ausilio dell'editor **Visual Studio Code**.

Per sperimentare le varie pipeline, ho avuto accesso alla potenza di calcolo delle **GPU universitarie**, dedicate a progetti di ricerca in **deep learning**. Per sfruttare queste risorse, ho utilizzato il protocollo **SSH** (Secure Shell) 1.3 per connettermi in remoto ai server e gestire l'esecuzione dei miei esperimenti.

Il sistema era dotato di **4 GPU NVIDIA** (modello Tesla V100 o A100, a seconda della disponibilità), condivise tra diversi utenti del dipartimento. Per evitare conflitti nell'allocatione delle risorse, è stato necessario prenotare in anticipo le GPU tramite un sistema di schedulazione interno, basato su code di priorità.

Nel dettaglio, il nodo su cui ho lavorato disponeva delle seguenti GPU:

- **3x NVIDIA Tesla V100-PCIE-16GB**, GPU ad alte prestazioni con 16 GB di memoria, molto diffuse in ambito scientifico per il training di deep neural networks.
- **1x NVIDIA A100 80GB PCIe**, una delle GPU più potenti attualmente disponibili per il calcolo scientifico, con ben 80 GB di memoria dedicata.

Le informazioni mostrate da `nvidia-smi` includevano anche temperatura, consumo energetico, utilizzo della memoria e livello di attività di ciascuna GPU.

Questa infrastruttura ha permesso di testare i miei modelli su dataset di grandi dimensioni con tempi di addestramento molto più rapidi rispetto all'uso di una macchina locale.

**Figura 1.3:** Esempio di output del comando `nvidia-smi` per monitorare le GPU disponibili.

# Capitolo 2

## Background e Stato dell'Arte

### Contents

---

2.1	L'Evoluzione delle Tecniche di Segmentazione . . . . .	8
2.2	Architettura U-Net per la Segmentazione di Immagini Mediche . . . . .	10
2.3	Strumenti e Dataset Moderni . . . . .	11
2.4	Stato dell'Arte nella Segmentazione del Seno in MRI 3D . . . . .	11

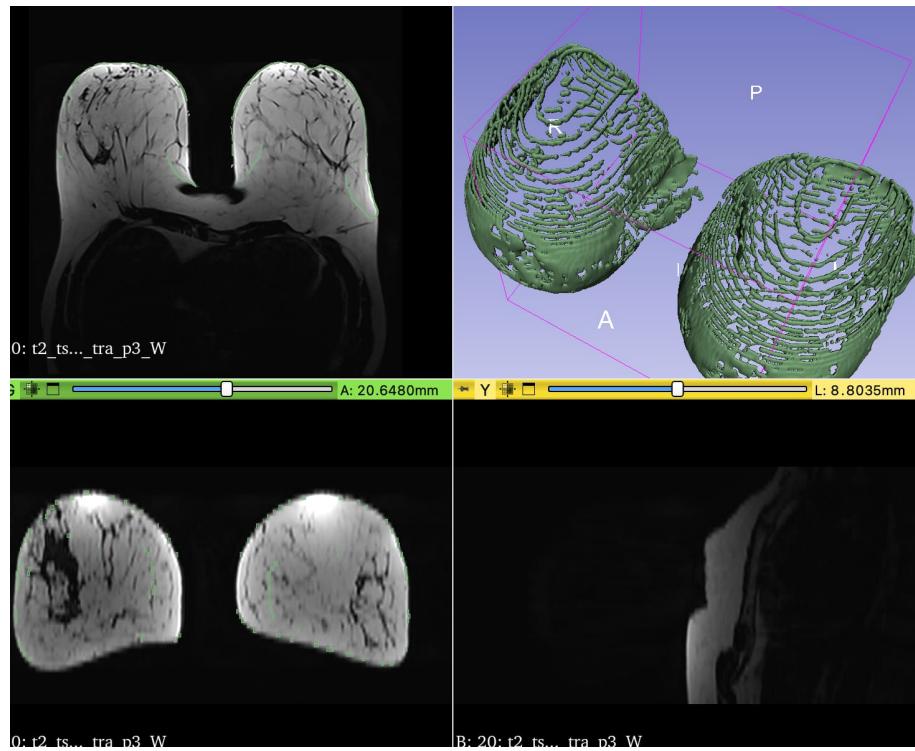
---

La segmentazione semantica di immagini mediche rappresenta una delle sfide più importanti nel campo della diagnostica. Questo compito, che consiste nel delineare con precisione **strutture anatomiche** o **aree patologiche** all'interno di **immagini biomediche**, ha un impatto diretto su applicazioni cliniche cruciali come la pianificazione chirurgica e il monitoraggio terapeutico. Nel contesto specifico della **mammella**, la segmentazione da immagini **MRI 3D** presenta peculiarità che la rendono particolarmente **complessa**, tra cui l'elevata **variabilità anatomica tra pazienti**, la presenza di **artefatti tipici delle risonanze magnetiche** e la necessità di bilanciare **accuratezza** e **tempi di elaborazione** quando si lavora con volumi tridimensionali ad alta risoluzione.

### 2.1 L'Evoluzione delle Tecniche di Segmentazione

Prima dell'avvento del **deep learning**, la segmentazione di immagini mediche si basava principalmente su approcci tradizionali che, pur rappresentando soluzioni pionieristiche per l'epoca, presentavano limiti significativi. Tecniche come il **thresholding** e il **region-growing**, ad esempio, erano ampiamente utilizzate per la loro semplicità concettuale, ma risultavano estremamente sensibili alla qualità dell'immagine, fallendo spesso in presenza di rumore o basso contrasto tra i tessuti. Allo stesso modo, i **deformable models**, che cercavano di adattare contorni attivi alle strutture anatomiche,

richiedevano un'inizializzazione manuale e faticavano a gestire la complessa morfologia della ghiandola mammaria.



**Figura 2.1:** Outer surface segmentation of breast MRI image. [2]

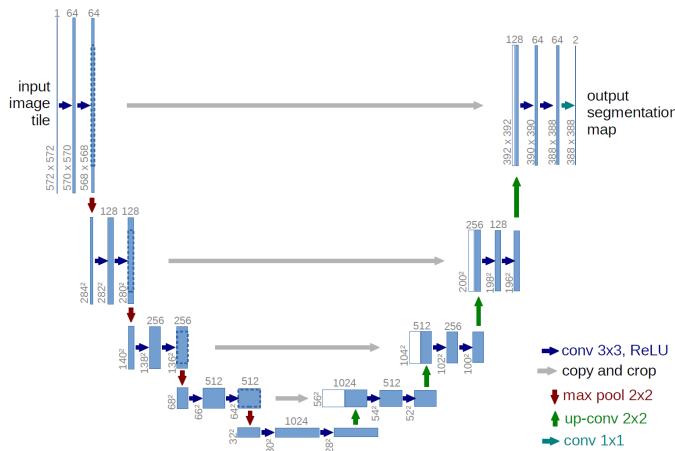
Con l'introduzione delle reti **neurali convoluzionali (CNN)**, il panorama della segmentazione medica è cambiato radicalmente. L'architettura **U-Net**, proposta nel 2015 da **Ronneberger et al.**, ha rappresentato una svolta grazie alla sua struttura **encoder-decoder** e alle **skip connections**, che permettono di combinare informazioni a diversi livelli di risoluzione, preservando i dettagli spaziali fondamentali per una segmentazione precisa. Successivamente, la comunità scientifica ha sviluppato varianti sempre più avanzate, come la **V-Net**, ottimizzata per dati volumetrici, e il framework **nnUNet**, in grado di adattarsi automaticamente alle caratteristiche di diversi dataset medici.

Negli ultimi anni, l'attenzione si è spostata verso i **Transformers**, modelli nati nell'ambito del **Natural Language Processing** e poi adattati con successo all'analisi di immagini. Architetture come **TransUNet** e **Swin UNETR** combinano la capacità delle CNN di estrarre features locali con il potere dei Transformers di modellare relazioni globali, offrendo prestazioni superiori in molti task di segmentazione. Tuttavia, questi modelli richiedono risorse computazionali elevate e grandi quantità di dati annotati, il che ne limita ancora l'applicabilità in alcuni contesti clinici.

## 2.2 Architettura U-Net per la Segmentazione di Immagini Mediche

Tra le architetture più utilizzate nel campo della segmentazione semantica applicata all'imaging biomedico, la **U-Net** occupa senza dubbio un ruolo di primo piano. Introdotta da **Ronneberger et al. nel 2015**, questa rete è stata progettata appositamente per segmentare immagini mediche anche in condizioni di **disponibilità limitata di dati annotati**, un aspetto ricorrente nei contesti clinici reali. Il successo della U-Net è dovuto non solo alla sua efficacia, ma anche alla sua semplicità architetturale, che la rende estremamente versatile e facilmente adattabile a diverse tipologie di dati, tra cui immagini 2D, 3D o multi-canale.

L'architettura della U-Net si sviluppa secondo una struttura simmetrica a forma di U (fig. 2.2), composta da due fasi principali: un percorso di **contrazione**, o encoder, e un percorso di **espansione**, o decoder. La fase di contrazione ha il compito di **estrarre rappresentazioni sempre più astratte** e semantiche dell'immagine in input, riducendone progressivamente la risoluzione attraverso **convoluzioni** e operazioni di **pooling**. Al contrario, la fase di espansione mira a **ricostruire l'informazione spaziale originaria**, riportando l'output alla dimensione dell'immagine di partenza mediante operazioni di **upsampling** e **convoluzioni**.



**Figura 2.2:** Schema concettuale dell'architettura U-Net.

Una caratteristica distintiva della U-Net rispetto ad altre reti di segmentazione è la presenza delle cosiddette **skip connections**. Questi collegamenti diretti tra i livelli corrispondenti **dell'encoder** e del **decoder** permettono di trasportare le informazioni a bassa astrazione, spesso perse durante il downsampling, direttamente nei livelli di ricostruzione. In questo modo, la rete riesce a **combinare efficacemente il contesto globale** dell'immagine con i **dettagli locali**, migliorando significativamente la **precisione dei bordi** e la **definizione delle strutture segmentate**.

Dal punto di vista pratico, la U-Net prende in input un'immagine e produce come output una **maschera segmentata**, in cui ogni pixel (o voxel nel caso 3D) è classificato in una

determinata categoria. Questo rende la rete particolarmente adatta per compiti dove è richiesta una **segmentazione voxel-wise**, come nel caso dell'identificazione di tessuti, lesioni o strutture anatomiche specifiche.

In tal senso, la U-Net si è dimostrata una **scelta solida**, grazie alla sua capacità di generalizzare anche su strutture anatomiche complesse, pur mantenendo una buona efficienza computazionale, ma nonostante la sua efficacia, la U-Net presenta alcune **limitazioni**. In presenza di **rumore, artefatti** o **grandi squilibri tra le classi**, la rete può mostrare difficoltà, ad esempio nel riconoscere strutture piccole o nel distinguere tessuti contigui con caratteristiche simili. Per superare tali difficoltà, sono state sviluppate diverse **estensioni e varianti** dell'architettura originale, come l'**U-Net++** con connessioni più profonde, le **U-Net con meccanismi di attention**, o le versioni 3D per dati volumetrici. Tuttavia, anche nella sua forma standard, la U-Net continua a rappresentare un punto di partenza solido e affidabile per molte applicazioni di segmentazione in ambito medico.

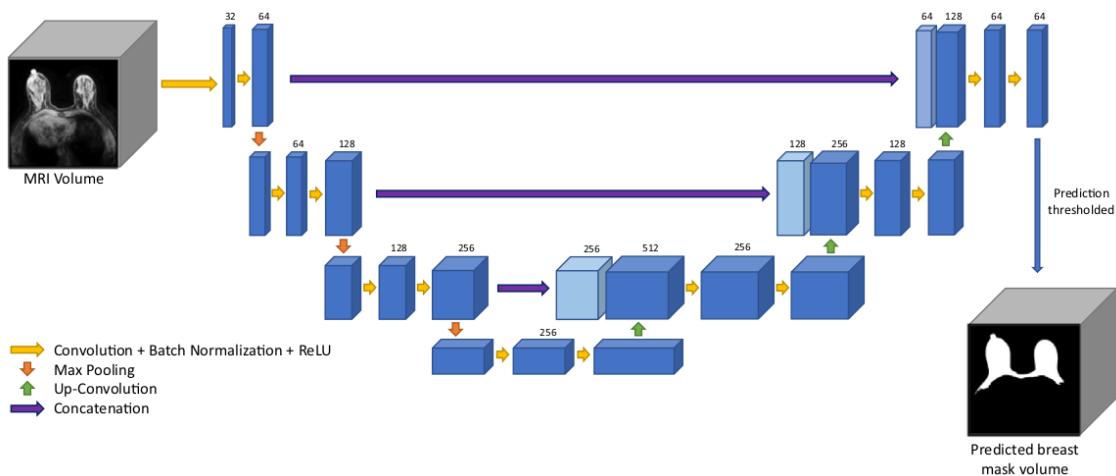
## 2.3 Strumenti e Dataset Moderni

Oggi, lo sviluppo di pipeline per la segmentazione medica si avvale di strumenti sempre più sofisticati. Tra questi, il framework **MONAI (Medical Open Network for AI)** si è affermato come uno standard de facto, grazie alla sua vasta raccolta di trasformazioni specifiche per immagini mediche, modelli predefiniti e metriche di valutazione. Basato su **PyTorch**, MONAI semplifica notevolmente la gestione di dati complessi come quelli DICOM e supporta l'implementazione di workflow riproducibili, essenziali per la ricerca in ambito medico.

Per quanto riguarda i dataset, il **Duke-Breast-Cancer-MRI** [7], disponibile su **The Cancer Imaging Archive (TCIA)**, rappresenta una risorsa preziosa per lo studio della segmentazione mammaria. Questo dataset include volumi MRI multiparametrici annotati manualmente da radiologi esperti, catturando tutta la complessità anatomica e le sfide tipiche delle immagini reali, come la presenza di lesioni e la variabilità nella densità del tessuto.

## 2.4 Stato dell'Arte nella Segmentazione del Seno in MRI 3D

Come già accennato, negli ultimi anni, la segmentazione automatica delle immagini mediche ha ricevuto un crescente interesse grazie all'utilizzo di reti neurali convoluzionali (CNN), in particolare per compiti complessi come l'analisi della densità mammaria. Uno studio di riferimento in questo ambito è quello di **Lew et al. (2024)** [8], che ha proposto un modello basato su U-Net per la segmentazione automatica del seno, del tessuto fibroghiandolare (FGT) e dei vasi sanguigni su risonanza magnetica (MRI) pre-contrasto (fig. 2.3).



**Figura 2.3:** Overview dei dati di input e la U-NET usata per la segmentazione del seno. [8]

Il lavoro si distingue per l'approccio rigoroso e riproducibile all'annotazione dei dati: **100 studi MRI**, selezionati dal già citato dataset pubblico *Duke Breast Cancer MRI* [7], sono stati annotati manualmente seguendo criteri ben definiti e validati da radiologi specializzati. Le annotazioni sono tridimensionali e comprendono il **tessuto mammario**, il **FGT** e i **vasi**, un elemento spesso trascurato in studi precedenti.

Dal punto di vista architettonale, il sistema utilizza **due reti U-Net in cascata**:

- **Breast U-Net**, che segmenta il tessuto mammario nell'intero volume MRI.
- **FGT-Vessel U-Net**, che utilizza sia il volume MRI che la segmentazione del seno come input per individuare il FGT e i vasi sanguigni.

Le performance sono state valutate tramite il **coefficiente di similarità di Dice (DSC)**, ottenendo i seguenti risultati medi sul test set:

- **Breast segmentation: DSC = 0.92 (3D), 0.95 (2D)**
- **FGT segmentation: DSC = 0.86 (3D), 0.84 (2D)**
- **Blood vessels: DSC = 0.65 (3D), 0.53 (2D)**

Un elemento innovativo del lavoro è la **segmentazione esplicita dei vasi**, che ha mostrato impatto significativo nella stima della densità mammaria. In effetti, il **volume dei vasi rappresentava in media il 5.7% del totale dei voxel etichettati come FGT o vasi**. Trascurare questo aspetto, come avveniva in studi precedenti, può portare a una sovrastima della densità mammaria.

# Capitolo 3

## Dataset: Duke Breast Cancer MRI

### Contents

---

3.1	Composizione del Dataset	13
3.1.1	Annotazioni e Ground Truth	14
3.1.2	Preprocessing e Integrazione con MONAI	14
3.1.3	Suddivisione del Dataset	15

---

### 3.1 Composizione del Dataset

Il dataset utilizzato proviene dal database **Duke Breast Cancer MRI** [7], un archivio pubblico messo a disposizione dalla **Duke University Medical Center** e accessibile tramite il portale **The Cancer Imaging Archive (TCIA)**. Questo dataset è stato sviluppato per promuovere la ricerca nel campo della segmentazione automatica della mammella e contiene immagini di **risonanza magnetica (MRI)** acquisite da pazienti con diagnosi confermata di carcinoma mammario invasivo.

Ogni studio è composto da sequenze MRI in formato **DICOM**, ottenute in posizione prona utilizzando scanner da **1.5 T o 3.0 T (prodotti da GE Healthcare o Siemens)**. Le immagini sono acquisite in **proiezione assiale**, con una **risoluzione spaziale variabile** compresa tra **0.6×0.6×1.0 mm<sup>3</sup>** e **1.1×1.1×1.2 mm<sup>3</sup>**, rendendole adatte all'elaborazione tridimensionale.

Le sequenze utilizzate nel progetto corrispondono alle immagini **T1-weighted fat-suppressed pre-contrast**, selezionate per l'elevato contrasto tra il **tessuto adiposo** e quello **fibroghiandolare (FGT)**, senza l'interferenza di mezzi di contrasto. Questa scelta metodologica



**Figura 3.1:** Fast Breast MRI Available to Women with Average Breast Cancer Risk — Duke Health [3]

ca è coerente con quanto proposto da **Lew et al. (2024)**, che hanno dimostrato l'efficacia di tale sequenza per la segmentazione automatica del FGT e dei vasi sanguigni.

### 3.1.1 Annotazioni e Ground Truth

Una caratteristica distintiva del dataset è la disponibilità di **annotazioni manuali tridimensionali**, elaborate da **annotatori formati** e successivamente validate da **radiologi esperti in senologia**. Ogni caso include **segmentazioni voxel-wise** delle seguenti strutture:

- **Seno (breast)**: delimitazione del tessuto mammario escludendo parete toracica, sternale e linfonodi;
- **Tessuto fibroghiandolare (FGT)**: zone a maggiore densità che costituiscono l'indicated primario per la valutazione del rischio;
- **Vasi sanguigni**: rami dell'arteria mammaria interna e dell'arteria toracica laterale, spesso iperintensi anche in assenza di contrasto.

### 3.1.2 Preprocessing e Integrazione con MONAI

Durante la fase di preparazione, le immagini DICOM sono state convertite in un formato compatibile con **MONAI (Medical Open Network for AI)** [5].

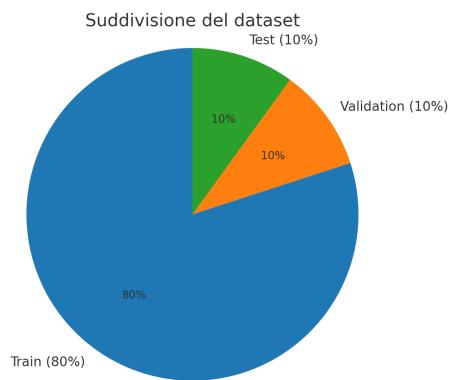
La pipeline di preprocessing ha incluso le seguenti operazioni:

- il **caricamento** delle immagini e delle relative maschere;
- la **normalizzazione** dell'intensità per migliorare la stabilità del training;
- il **ridimensionamento** delle immagini a una risoluzione uniforme per l'input nei modelli di rete neurale;
- la conversione in **dizionari Python** contenenti l'immagine e la maschera;
- il **salvataggio** del dataset preprocessato su disco, pronto per essere utilizzato nel training.

### 3.1.3 Suddivisione del Dataset

Al fine di garantire una valutazione robusta e riproducibile dei modelli, l'intero dataset è stato suddiviso in tre sottoinsiemi:

- **Training set (80%)**: utilizzato per l'addestramento del modello, comprendente la maggior parte dei dati e delle variabilità cliniche;
- **Validation set (10%)**: impiegato durante l'addestramento per monitorare l'overfitting e ottimizzare gli iperparametri;
- **Test set (10%)**: utilizzato esclusivamente per la valutazione finale delle prestazioni del modello, senza interferenze in fase di sviluppo.



**Figura 3.2:** Grafico a torta della suddivisione del dataset

# Capitolo 4

## Configurazione Sperimentale

### Contents

---

4.1	Il file di configurazione YAML . . . . .	17
4.1.1	Struttura generale . . . . .	17
4.1.2	Configurazione del training . . . . .	18
4.1.3	Trasformazioni dei dati . . . . .	18
4.1.4	Definizione del modello . . . . .	18
4.1.5	Ottimizzazione e loss . . . . .	19
4.1.6	Post-processing e valutazione . . . . .	19
4.2	Gestione del Logging . . . . .	20

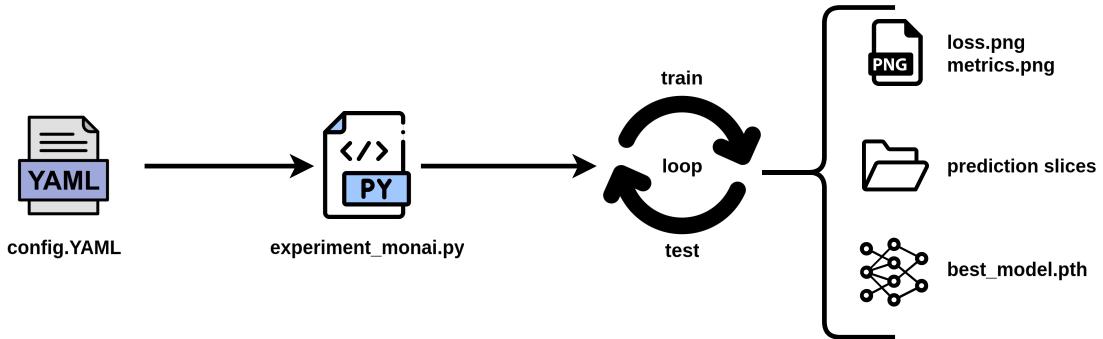
---

Per effettuare gli esperimenti ho avuto la possibilità di utilizzare un **framework sperimentale** sviluppato dall'ingegner **Marco Cantone** [4], correlatore di questa tesi. Il cuore del framework risiedeva nella sua capacità di **astrarre la complessità tipica degli esperimenti di deep learning** attraverso una struttura di configurazione **gerarchica e intuitiva**. Ogni aspetto cruciale del processo, dal **preprocessing** dei dati alla **definizione dell'architettura**, dagli **iperparametri di training** alle **metriche di valutazione**, trovava una precisa collocazione nel file **YAML**.

Uno dei maggiori vantaggi di questo approccio emergeva nella **fase di validazione incrociata delle ipotesi**. La possibilità di confrontare direttamente diverse varianti architettoniche, mantenendo invariati tutti gli altri parametri, ha permesso di isolare con precisione l'impatto di ciascuna scelta progettuale.

Sebbene il framework fornisse una struttura organizzata per gli esperimenti, **non era dotato di sistemi intelligenti in grado di verificare automaticamente la coerenza logica delle configurazioni**. Spettava quindi a me, durante la definizione dei parametri

nel file YAML, assicurarmi che le scelte fossero **compatibili** tra loro e adatte al contesto. Ad esempio, se impostavo un'architettura progettata per input volumetrici, dovevo verificare manualmente che anche le trasformazioni di preprocessing (come il random padding o il resizing) fossero configurate per operare su tre dimensioni, evitando incongruenze che avrebbero compromesso l'addestramento.



**Figura 4.1:** Esempio di pipeline sperimentale con configurazione YAML.

Allo stesso modo, quando sperimentavo ottimizzatori diversi, dovevo prestare attenzione alla scelta del **learning rate** e dello **scheduler**, poiché valori troppo aggressivi potevano **destabilizzare il training**, mentre quelli troppo conservativi **rallentavano inutilmente la convergenza**.

Questo processo richiedeva una costante **analisi degli errori**: se un esperimento falliva o produceva risultati insoliti, dovevo riesaminare la configurazione YAML per identificare possibili discrepanze, come **dimensioni di crop incompatibili** con la **risoluzione del volume** o **parametri di augmentazione eccessivamente distorti**. La mancanza di validazione automatica ha reso il lavoro più impegnativo, ma mi ha spinto a **sviluppare una profonda comprensione delle dipendenze tra i vari componenti del sistema**.

## 4.1 Il file di configurazione YAML

Il file di configurazione YAML è organizzato in **sezioni logiche** che definivano ogni aspetto del processo di **addestramento** e **valutazione**. Di seguito, analizziamo le componenti principali del file, suddividendolo in sottoinsiemi funzionali.

### 4.1.1 Struttura generale

Il file YAML è organizzato in 6 sezioni principali:

```
# Esempio di struttura generale
TRAINING:
  # parametri di addestramento
TRAIN_TRANSFORM:
  # trasformazioni per il training
```

```

TEST_TRANSFORM:
    # trasformazioni per il test
MODEL:
    # architettura del modello
LOSS:
    # funzione di loss
OPTIMIZER:
    # ottimizzatore

```

#### 4.1.2 Configurazione del training

La sezione TRAINING definiva i parametri fondamentali:

```

TRAINING:
    workspace: /path/to/experiment_results
    dataset_root: /path/to/dataset
    valid_size: 0.1    # 10% validation set
    test_size: 0.1    # 10% test set
    train_batch_size: 4
    test_batch_size: 1
    num_workers: 8      # thread per data loading
    device: cuda:0      # GPU da utilizzare
    epochs: 50          # numero di epoche
    roi_size: [256, 256, 32]  # dimensione regioni di interesse

```

#### 4.1.3 Trasformazioni dei dati

Le sezioni TRAIN\_TRANSFORM e TEST\_TRANSFORM specificavano il preprocessing:

```

TRAIN_TRANSFORM:
    - class: monai.transforms.LoadImaged
        params:
            keys: ['post_2', 'breast']
    - class: monai.transforms.RandSpatialCropSamplesd
        params:
            keys: ['img', 'seg']
            num_samples: 4
            roi_size: [512, 512, 8]

```

#### 4.1.4 Definizione del modello

La sezione MODEL configurava l'architettura della rete:

```

MODEL:
    class: monai.networks.nets.Unet

```

```

params:
    channels: [16, 32, 64]      # canali per ogni livello
    in_channels: 1              # canali input
    out_channels: 2             # canali output
    norm: INSTANCE            # normalizzazione
    spatial_dims: 3             # dimensione spaziale
    strides: [2, 2, 2]          # stride dei livelli

```

#### 4.1.5 Ottimizzazione e loss

Le sezioni LOSS e OPTIMIZER controllavano l'addestramento, includendo la funzione di loss e l'ottimizzatore:

**LOSS:**

```

class: monai.losses.DiceFocalLoss
params:
    weight_dice: 0.7
    weight_ce: 0.3

```

**OPTIMIZER:**

```

class: torch.optim.AdamW
params:
    lr: 1e-3                  # learning rate
    weight_decay: 1e-3        # regolarizzazione

```

#### 4.1.6 Post-processing e valutazione

Le sezioni finali gestivano la valutazione:

**METRIC:**

```

class: monai.metrics.DiceMetric
params:
    include_background: false
    reduction: mean

```

**POST\_PRED:**

- **class**: monai.transforms.AsDiscrete
- params**:
- argmax**: true
- to\_onehot**: 2

**POST\_LABEL:**

- **class**: monai.transforms.AsDiscrete
- params**:

```
to_onehot: 2
```

## 4.2 Gestione del Logging

Per ogni esperimento è stato adottato un sistema di **logging automatico** organizzato in modo da raccogliere tutti gli output generati in una **cartella dedicata**, con l'obiettivo di garantire tracciabilità, riproducibilità e analisi approfondita dei risultati.

La struttura delle cartelle seguiva uno schema gerarchico, con una cartella principale per ogni esperimento, identificata da un nome univoco basato sulla data e sull'ora di inizio. All'interno di questa cartella, venivano create sottocartelle per ogni fase del processo di addestramento e valutazione, consentendo una facile navigazione e gestione dei risultati. Ad esempio, la struttura poteva apparire come segue:

```
--- exp_i/
|   --- log.txt
|   --- loss_curve.png
|   --- metric_curve.png
|   --- best_model.pth
|   --- predictions/
|       --- case_001/
|           |   --- slice_01.png
|           |   --- slice_02.png
|           |   ...
|       --- case_002/
|           |   --- slice_01.png
|           |   ...
|       ...
|   ...
```

Ogni cartella conteneva le seguenti informazioni:

- Un file `log.txt` contenente tutte le stampe console (`print`) prodotte durante il training. In questo file venivano riportati:
  - la **loss di training** e **validazione** per epoca;
  - le **metriche di valutazione** (es. **Dice score**);
  - eventuali **messaggi diagnostici**.
- Due **grafici** in formato immagine (`.png`):
  - il grafico della **curva della loss** per epoca;
  - il grafico della **metrica** per epoca.
- Il file del **modello salvato** (`.pth`), che rappresenta lo **stato della rete al miglior checkpoint**.

- Una sottocartella `predictions/` contenente, per ogni caso del **test set**, le **immagini salvate slice per slice**. Ogni immagine mostrava sovrapposte:
  - l'**immagine originale della MRI dell'i-esima slice**
  - la **ground truth** dell'i-esima slice
  - la **predizione generata dal modello** dell'i-esima slice

Questa struttura ha permesso non solo di **monitorare l'andamento dell'addestramento**, ma anche di effettuare **confronti qualitativi tra modelli diversi** e analizzare visivamente i punti di **forza** e **debolezza** della rete. Inoltre, la separazione per cartelle ha reso semplice la gestione di esperimenti multipli e il confronto sistematico tra diverse configurazioni di training.

# Capitolo 5

## Esperimenti

### Contents

---

5.1	Introduzione . . . . .	22
5.1.1	Nota sulle Valutazioni . . . . .	23
5.2	Impostazione degli Esperimenti . . . . .	23
5.3	Evoluzione Architetturale e Tuning degli Iperparametri . . . . .	23
5.4	Strategia Alternativa: Approccio 2D e Architettura SliceUNet . . . . .	26
5.5	Ritorno all'approccio 3D: Affinamento e Validazione . . . . .	27
5.6	Nuovo approccio di Valutazione . . . . .	27
5.6.1	L'Attention U-Net . . . . .	28
5.7	Esperimento Finale . . . . .	29
5.7.1	Post-processing morfologico: operazione di Closing . . . . .	31

---

### 5.1 Introduzione

Sono stati condotti diversi esperimenti, cercando di variare gli aspetti della configurazione del modello e del processo di addestramento, al fine di ottimizzare le prestazioni nella segmentazione delle immagini mammografiche. Si è cercato di **cambiare gli iperparametri uno alla volta**, mantenendo costanti gli altri, per isolare l'impatto di ciascuna modifica.

Lo sviluppo di un sistema di segmentazione automatica efficace richiede non solo un'adeguata progettazione architettonica, ma anche un attento processo di **ottimizzazione sperimentale**, che tenga conto delle molteplici variabili che influenzano l'apprendimento del modello. In questo capitolo viene descritto in dettaglio il percorso iterativo seguito, che ha portato a una **progressiva evoluzione del modello** fino a raggiungere **performance soddisfacenti**.

### 5.1.1 Nota sulle Valutazioni

Nei primi **16 esperimenti**, la valutazione sul **test set** è stata effettuata utilizzando le **immagini trasformate**, ovvero mantenendo le dimensioni e le modifiche introdotte durante il preprocessing (incluso il resize). A partire dall'esperimento **17**, invece, è stata applicata una **trasformazione inversa** (in particolare il resize inverso) alle ground truth, riportandole alle dimensioni originali. Questo ha permesso una valutazione più corretta e realistica delle performance, evitando possibili distorsioni dovute alle trasformazioni di preprocessing.

EXP	Architettura	Parametri Chiave
1-3	U-Net 3D	Canali [16,32,64,128]
4-10	U-Net 3D	Canali [16,32,64,128,256]
12,13	SliceU-Net 2D	Approccio 2D slice-based
14-21	U-Net 3D	Varianti profondità/canali
20	Attention U-Net 3D	Meccanismo di attention

**Tabella 5.1:** Configurazioni principali dei modelli sperimentalni con architetture e parametri chiave.

## 5.2 Impostazione degli Esperimenti

Gli esperimenti sono stati condotti in **maniera incrementale**, seguendo una filosofia di **ottimizzazione guidata da evidenze**, nella quale ciascuna modifica introdotta agli iperparametri o alla pipeline di preprocessing è stata motivata da osservazioni emerse negli esperimenti precedenti. Tutte le prove sono state **documentate** e **monitorate** con precisione.

Per garantire la coerenza e individuare possibili problemi, è stato anche necessario implementare un sistema di **debugging interno** tramite Python, includendo **stampe delle dimensioni dei tensori nei momenti chiave** del forward pass e salvataggi intermedi delle immagini durante il training. Questo approccio ha **permesso di identificare rapidamente errori legati a mismatch dimensionali, problemi nella gestione della normalizzazione, o trasformazioni incoerenti nei dati di input/output**.

## 5.3 Evoluzione Architetturale e Tuning degli Iperparametri

Il **primo esperimento** (tab. 5.2) ha costituito la **base di partenza**, con una configurazione **standard della rete U-Net tridimensionale**, con **4 blocchi convoluzionali** (`channels: [16, 32, 64, 128]`), **normalizzazione batch**, e un **resize uniforme** applicato a tutte le immagini. Con questa configurazione, è stato ottenuto un **Dice score sul test set pari a 0.7894**.

Parametro	Valore
<b>Architettura</b>	U-Net 3D standard con 4 blocchi convoluzionali
<b>Canali</b>	[16, 32, 64, 128]
<b>Normalizzazione</b>	Batch normalization
<b>Resize</b>	Uniforme su tutte le immagini
<b>Dice score (test set)</b>	0.7894

**Tabella 5.2:** Configurazione del primo esperimento (baseline) con architettura U-Net 3D e risultati ottenuti.

Il **passo successivo** ha previsto un **aumento del learning rate** dell'ottimizzatore **Adam**, passando da **0.0005** a **0.005**. Questo semplice cambiamento ha prodotto un **incremento della metrica di circa +1.17%**, suggerendo che la rete era in grado di **convergere più rapidamente in presenza di un gradiente iniziale più marcato**. Da qui è emersa l'intuizione che la rete potesse beneficiare anche di una **maggior profondità**: il **terzo** e il **quarto** esperimento (tab. 5.3) **hanno confermato questa ipotesi, mostrando miglioramenti crescenti in validazione e test** con l'introduzione di un ulteriore blocco (channels: [16, 32, 64, 128, 256]), fino a raggiungere **0.8597 sul test set** e **0.8569 sul validation set**.

Parametro	Valore
<b>Architettura</b>	U-Net 3D profonda con 5 blocchi convoluzionali
<b>Canali</b>	[16, 32, 64, 128, 256]
<b>Normalizzazione</b>	Batch normalization
<b>Dice score (test)</b>	0.8597
<b>Dice score (validazione)</b>	0.8569

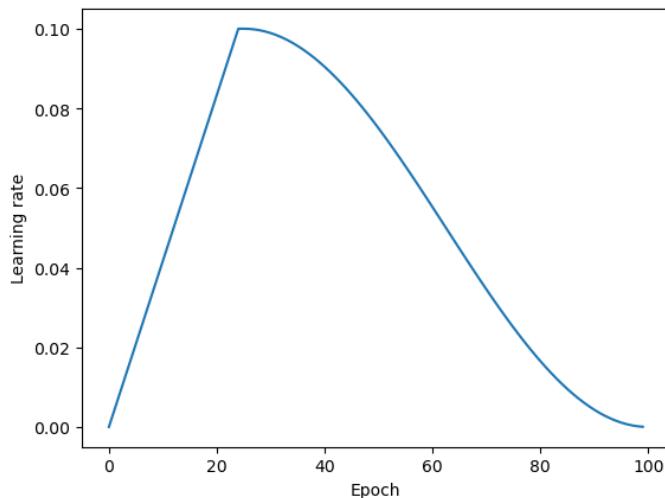
**Tabella 5.3:** Risultati degli esperimenti 3-4 con architettura U-Net 3D più profonda. L'aggiunta di un quinto blocco convoluzionale ha portato a significativi miglioramenti nelle metriche.

A partire dal **quinto esperimento** (tab. 5.4) è stata esplorata una **combinazione di modifiche strutturali**, tra cui il passaggio da **Adam** ad **AdamW**, il cambiamento della **normalizzazione** da **Batch** a **Instance** e l'utilizzo di una **loss composita (Focal + Dice Loss)**, ciascuna ponderata per gestire in modo bilanciato gli squilibri tra le classi. Questa configurazione ha prodotto **ottimi risultati in validazione (0.8623) e (0.8597) sul test set**.

Componente	Modifica
Ottimizzazione	Sostituzione Adam → AdamW
Normalizzazione	Batch → Instance
Loss function	Dice → DiceFocal ( $\lambda = 0.7/0.3$ )
Metriche	$\Delta$ Valid +3.1%, $\Delta$ Test +0.32%
Risultati	Valid: 0.8623, Test: 0.8597

**Tabella 5.4:** Analisi comparativa delle modifiche introdotte dall'EXP 5. Tutti i cambiamenti hanno contribuito al miglioramento delle performance.

Nei **successivi esperimenti**, si è indagata ulteriormente la sensibilità del modello al **learning rate**, che era stato inizialmente portato troppo in basso (0.0001). Ripristinando un valore intermedio (0.001), il **Dice score sul test è aumentato di oltre il 3%**. Parallelamente, l'introduzione di una strategia di **learning rate scheduling (LinearWarmupCosineAnnealingLR)** (fig. 5.1) ha portato a **miglioramenti più modesti ma stabili**.



**Figura 5.1:** Andamento del learning rate durante il training, con warmup iniziale e successiva discesa coseno.

Una seconda linea di sperimentazione ha riguardato il **trattamento dimensionale** delle immagini. Portando la dimensione da **(384x384)** a **(512x512)**, si è osservato un ulteriore **miglioramento della metrica**. Al contrario, strategie basate esclusivamente sul **padding**, o su un **mix di resize e padding**, hanno avuto un effetto **negativo** sulle performance, portando a un crollo significativo della metrica nel **nono e decimo** esperimento fino a **0.7576 sul test set e 0.0816 in validazione**.

Configurazione	Dimensione	Test Dice	Valid Dice
Baseline	384×384	0.7894	0.7920
Full Resize	512×512	<b>0,8878</b>	<b>0,8934</b>
Solo Padding	512×512	0.7576	0.8160
Mix Strategie	Variabile	0,8857	0,8848

**Tabella 5.5:** Confronto sistematico degli approcci dimensionali. I valori mostrano come il resize completo produca i migliori risultati, mentre il padding peggiora le performance.

## 5.4 Strategia Alternativa: Approccio 2D e Architettura SliceUNet

In seguito a questi risultati, è stata sperimentata una **nuova strategia**, volta a esplorare un **approccio 2D** invece che 3D. Questo ha richiesto l'adozione di una **ROI size** di tipo **(512, 512, 1)**, accompagnata da una trasformazione `RandSpatialCropSamplesd` e da un'opportuna gestione del padding. Per rendere l'intera pipeline compatibile con questo formato, è stata definita un'architettura **SliceUNet personalizzata**, progettata per operare su **singole slice bidimensionali** mantenendo il supporto volumetrico per **l'aggregazione del risultato finale**.

```
class SliceUNet(nn.Module):
    def __init__(self, in_channels, out_channels,
                 channels, norm, strides, spatial_dims=2):
        super(SliceUNet, self).__init__()
        self.Unet2D = monai.networks.nets.UNet(
            spatial_dims=spatial_dims,
            in_channels=in_channels,
            out_channels=out_channels,
            channels=channels,
            strides=strides,
            norm= norm,
        )
        def forward(self, x):
            x = self.Unet2D(x[...,0])

            return x[..., None]
```

I primi esperimenti con **SliceUNet** hanno ottenuto **risultati interessanti**: partendo da **0.628**, si è passati a **0.7808, in test**, modificando la strategia di preprocessing da **padding** a **resize**. Tuttavia, nonostante **l'eleganza** e la **semplicità computazionale** di questo approccio, i risultati **non hanno raggiunto i livelli di accuratezza ottenibili con la versione 3D**.

## 5.5 Ritorno all'approccio 3D: Affinamento e Validazione

Per questo motivo, l'attenzione è tornata su modelli tridimensionali, testando configurazioni diverse di `roi_size` (ad esempio,  $(512 \times 512 \times 16)$ ) e nuove combinazioni architettoniche. **Esperimenti successivi hanno mostrato che aumentando la profondità lungo l'asse z** e spingendo il numero di **epocha a 150**, si potevano ottenere performance vicine a **0.8863 in test**, che **costituivano il massimo assoluto fino a quel punto**.

Tipologia	Parametri Chiave	Test Dice	Δ
2D Base	SliceUNet, padding, 50 epocha	0.628	-
2D Migliorato	SliceUNet, resize $512 \times 512$	0.7808	↑24.3%
3D Intermedio	U-Net, ROI $512 \times 512 \times 16$	0.8668	↑38.0%
3D Ottimale	+150 epocha, asse z profondo	<b>0.8863</b>	↑41.1%

**Tabella 5.6:** Progressione prestazionale con scala cromatica: dal rosso (baseline) al blu (miglior risultato). I Δ verdi mostrano il miglioramento cumulativo, mentre il blu evidenzia il picco prestazionale (+41.1% rispetto alla baseline).

## 5.6 Nuovo approccio di Valutazione

Un **aspetto critico** che è emerso in questa fase è stato **il modo in cui le predizioni venivano valutate**. In effetti, l'adozione di tecniche di preprocessing alterava la struttura delle label, rendendo la metrica poco rappresentativa. È stata quindi introdotta una **pipeline di trasformazioni inverse** nella fase di test, per riportare le predizioni allo spazio originale e confrontarle correttamente con le etichette intonse.

```
if isinstance(test_loader.dataset.transform, Compose)
    and any(isinstance(tr, Resized)
        for tr in test_loader.dataset.transform.transforms):
    val_outputs = [Resized(keys="img", spatial_size=original_shape,
                           mode="nearest")
                  ("img": i})["img"] for i in val_outputs]
```

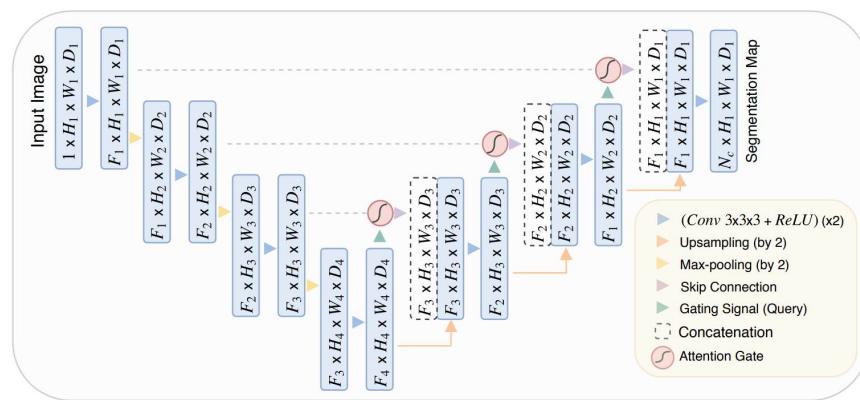
Con questa nuova metodologia di test, sono stati ripetuti alcuni esperimenti precedenti, mantenendo **invariati l'architettura, la configurazione degli iperparametri e il preprocessing**, ma modificando il modo in cui veniva eseguita la valutazione sul test set. L'introduzione delle **trasformazioni inverse**, ha evidenziato una **differenza sostanziale** tra le metriche riportate in precedenza e quelle ricalcolate in condizioni più rigorose. Ad esempio, l'esperimento **16**, che con il metodo di valutazione originale aveva raggiunto un **Dice score di 0.8863**, è stato ripetuto come esperimento **20**, ottenendo un valore **ridotto di 0.8567**.

Nonostante questa lieve flessione nei valori assoluti, la nuova metodologia **ha reso i confronti tra modelli molto più affidabili**. Alcune architetture, che in precedenza sembravano promettenti ma erano in realtà favorite da un confronto distorto, hanno rivelato prestazioni inferiori alle attese.

### 5.6.1 L'Attention U-Net

L'**Attention U-Net** (fig. 5.2) rappresenta una naturale estensione della classica architettura **U-Net**, progettata per migliorare la capacità del modello di **concentrarsi sulle regioni di interesse più rilevanti all'interno dell'immagine**. Introdotta da **Oktay et al. nel 2018** [9], questa variante si basa sull'integrazione di **meccanismi di attenzione spaziale** nei percorsi di **skip connection** tra **encoder** e **decoder**. L'obiettivo principale è quello di guidare la rete **nell'enfatizzare le strutture salienti** e nel **sopprimere attivamente le attivazioni inutili** o irrilevanti, soprattutto in contesti caratterizzati da alta complessità anatomica o forte squilibrio tra classi.

Nella **U-Net tradizionale**, le informazioni a bassa risoluzione estratte dall'encoder vengono concatenate direttamente con i feature map corrispondenti del decoder. Tuttavia, questo approccio assume che tutte le regioni abbiano pari rilevanza informativa, ignorando il fatto che non tutte le porzioni dell'immagine contribuiscono in egual misura alla segmentazione finale. L'**Attention U-Net** introduce quindi un **modulo di attenzione** che agisce come un **filtro “intelligente”**: prima della concatenazione, il modello valuta l'importanza di ciascuna regione mediante un **gating mechanism**, che calcola **mappe di attenzione** specifiche per ogni livello di **skip connection**.



**Figura 5.2:** Schema dell'Attention U-Net. Le frecce indicano il flusso di informazioni tra i moduli di encoder e decoder, con l'integrazione del modulo di attenzione.

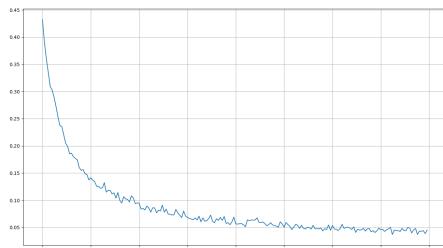
Dal punto di vista implementativo, il modulo di attenzione riceve in input due segnali: da un lato, **le feature provenienti dal livello encoder** (contenenti informazione spaziale locale); dall'altro, **i segnali di gating dal decoder** (contenenti informazione semantica più astratta). L'interazione tra questi due flussi genera una **mappa di attenzione**, applicata tramite moltiplicazione ai tensori encoder, prima della loro trasmissione al deco-

der. In questo modo, il decoder riceve solo le informazioni più rilevanti per la predizione finale.

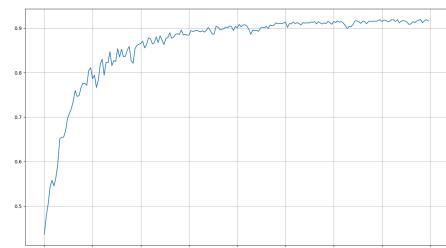
Nel contesto del presente lavoro, **nonostante un aumento moderato della complessità computazionale**, il modello **ha mostrato buone performance**, raggiungendo un **Dice score pari a 0.8415** nell'esperimento dedicato.

## 5.7 Esperimento Finale

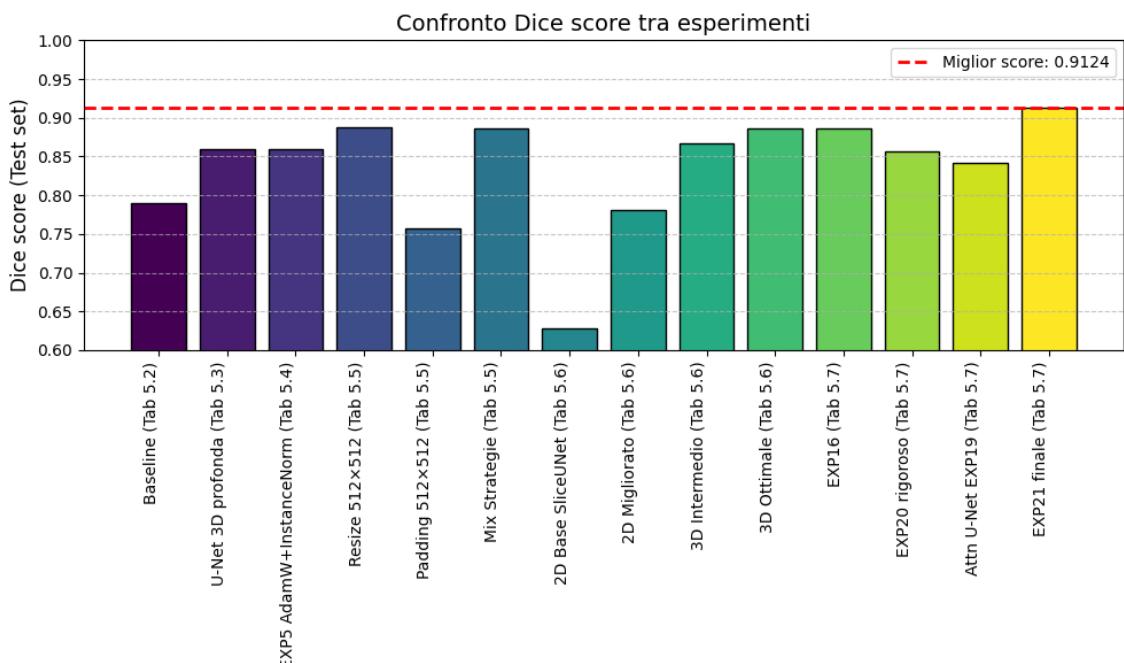
Il punto culminante di questa fase è stato rappresentato dall'esperimento **21**, in cui è stata utilizzata una **U-Net profonda con cinque livelli e stride finale lungo l'asse z pari a 1**. L'input adottava una ROI size tridimensionale pari a (512, 512, 8), e il training è stato prolungato fino a **200 epocha** (fig. 5.3 e fig. 5.4).



**Figura 5.3:** Andamento delle funzioni di loss durante il training dell'esperimento finale(EXP 21).



**Figura 5.4:** Andamento delle metriche di valutazione durante il training dell'esperimento finale(EXP 21).

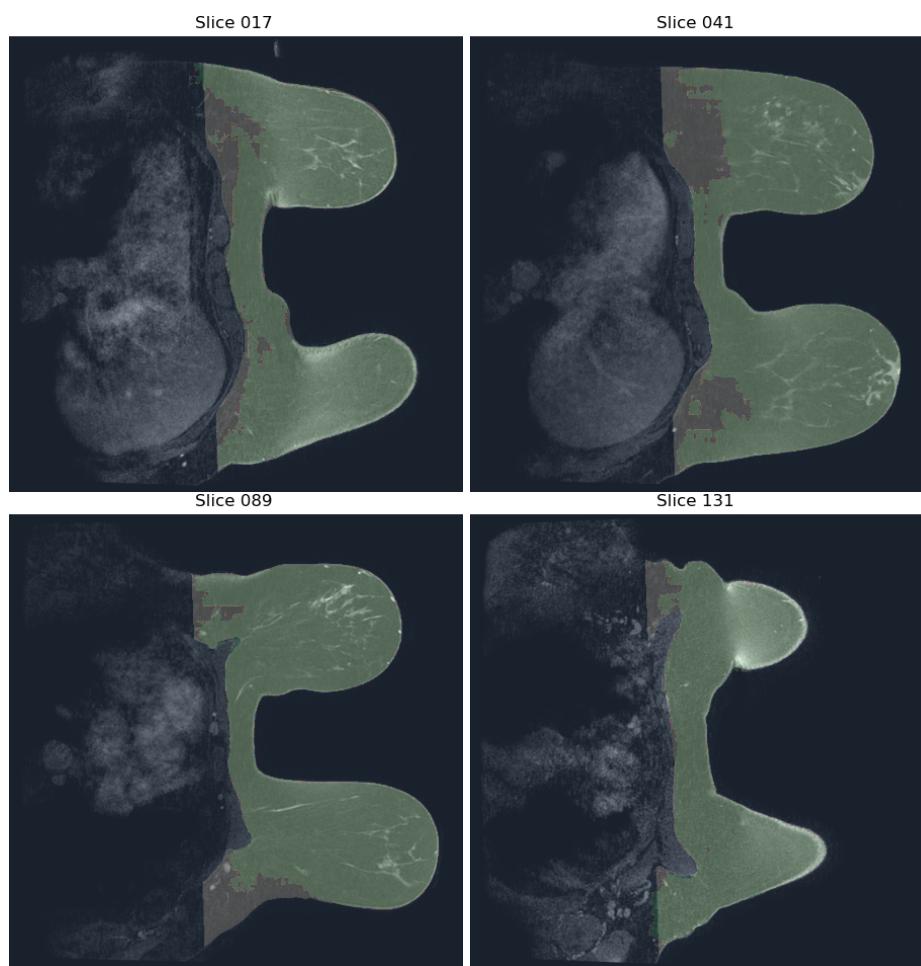


**Figura 5.5:** Confronto Dice score tra tutti gli esperimenti effettuati.

In queste condizioni, **il modello ha raggiunto un Dice score pari a 0.9124** (fig. 5.4), **il valore più alto registrato su tutto il test set** (fig. 5.6). Questo risultato rappresenta non solo il successo di una specifica configurazione, ma anche la validazione dell'intero **processo di ottimizzazione**, culminato in un **modello accurato, robusto** e valutato con **criteri metodologicamente solidi**.

EXP	Configurazione	Dice	$\Delta$	Note
16	Config. originale	0.8863	-3.3%	Sovrastima precedente
20	Stessa config. + val. rigorosa	0.8567	0%	Benchmark corretto
19	Attention U-Net	0.8415	-1.8% vs 20	Robustezza verificata
21	U-Net avanzato	0.9124	+6.5% vs 20	Nuovo state-of-the-art

**Tabella 5.7:** Analisi dettagliata dei risultati finali. La colonna  $\Delta$  mostra: per EXP 16 la sovrastima rispetto alla nuova metodologia, per EXP 19-21 la variazione rispetto al benchmark corretto (EXP 20).



**Figura 5.6:** Esempi di segmentazione ottenuti con il modello finale (EXP 21). Le immagini mostrano le predizioni su diverse slice del test set, evidenziando la capacità del modello di identificare correttamente le aree di interesse.

## 5.7.1 Post-processing morfologico: operazione di Closing

Al termine dell'addestramento del modello finale (EXP 21), è stata introdotta una fase opzionale di **post-processing morfologico** sulle maschere di segmentazione prodotte, con l'obiettivo di correggere eventuali discontinuità e imperfezioni residue lungo i margini delle strutture segmentate.

L'operazione applicata è il **closing morfologico** (*dilatazione seguita da erosione*). Questo tipo di trasformazione è comunemente utilizzato per **colmare piccoli vuoti** all'interno delle regioni segmentate, **uniformare i bordi** delle maschere e **migliorare la connettività** delle strutture anatomiche in presenza di interruzioni puntuali.

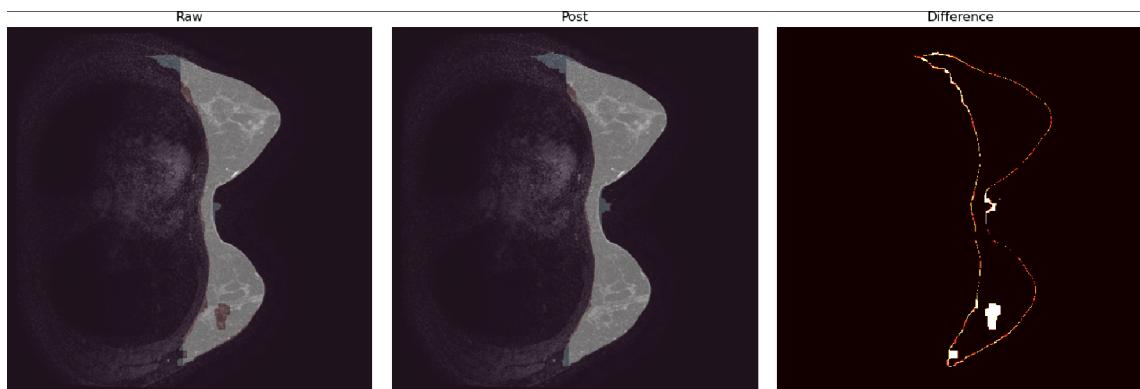
L'implementazione è stata realizzata creando una funzione custom in Python che applica il closing morfologico su ciascuna maschera di segmentazione generata dal modello. La funzione utilizza la libreria scikit-image per eseguire le operazioni morfologiche, specificando un **kernel di dimensione fissata dal parametro in ingresso**:

```
from scipy.ndimage import binary_closing
from monai.transforms import MapTransform
import torch
import numpy as np

class MorphologicalClosing():
    def __init__(self, dim=12):
        self.dim = dim
        self.structure = np.ones((self.dim, self.dim, self.dim), dtype=bool)
    def __call__(self, x):
        device = x.device if isinstance(x, torch.Tensor) else torch.device('cpu')
        # convert to numpy
        if isinstance(x, torch.Tensor):
            x = x.cpu().numpy()
        # remove first dimension
        x = x.squeeze(0) if x.ndim > 3 else x
        closed_np = binary_closing(x, structure=self.structure)
        # re-add first dimension
        closed_np = closed_np[None, ...]
        # convert back to tensor and move to original device
        closed_tensor = torch.tensor(closed_np, dtype=torch.float32,
                                      device=device)
        return closed_tensor
```

Sono state testate diverse dimensioni per il **kernel size** dell'operazione di closing, variando il parametro tra valori compresi tra 6 e 16. L'analisi sistematica ha mostrato che la dimensione ottimale era pari a **10**, che garantiva il miglior compromesso tra **uniformità**

**dei bordi e preservazione dei dettagli anatomici**, evitando sia un'eccessiva erosione sia la formazione di artefatti.



**Figura 5.7:** Esempio di segmentazione prima e dopo l'applicazione del closing morfológico. Le immagini mostrano come il closing migliori la coesione delle strutture segmentate, eliminando piccole discontinuità e rendendo le maschere più uniformi.

L'effetto del closing (fig. 5.7) è stato valutato rieseguendo la metrica di **Dice score** sul *test set*. Il modello finale, che in fase di validazione rigorosa aveva raggiunto un valore di **0.9124**, ha beneficiato di un incremento fino a **0.9161** dopo il post-processing. Questo risultato conferma che, pur trattandosi di un miglioramento numericamente modesto (+0.0037), **l'operazione di closing ha contribuito a rifinire ulteriormente le maschere, portando a un output visivamente più coerente e clinicamente più fruibile.**

# Capitolo 6

## Conclusioni

### Contents

---

6.1	Sintesi del Percorso Sperimentale . . . . .	33
6.2	Risultato Finale e Implicazioni Cliniche . . . . .	34
6.3	Possibili Downstream Task . . . . .	34
6.4	Prospettive di Sviluppo Futuro . . . . .	34
6.5	Riflessione Conclusiva . . . . .	35

---

### 6.1 Sintesi del Percorso Sperimentale

L'analisi sperimentale condotta ha evidenziato che il successo di una pipeline di segmentazione non dipende da un singolo fattore, bensì dall'interazione armonica tra **scelte architettonurali, strategie di preprocessing, criteri di valutazione e robustezza del training**. Ogni esperimento ha rappresentato un passo incrementale verso una maggiore comprensione del comportamento del modello, contribuendo alla costruzione di una soluzione **affidabile, generalizzabile** e potenzialmente **integrabile** in contesti clinici reali.

Il percorso si è rivelato altamente **iterativo** e **evidence-based**, partendo da una configurazione di base e introducendo, con rigore sperimentale, modifiche mirate al **learning rate**, alla **funzione di loss**, al tipo di **normalizzazione**, alla **profondità architetturale**, fino a giungere a una configurazione finale profondamente ottimizzata. Particolarmente significativo è stato l'inserimento di una strategia di **validazione con trasformazioni inverse**, che ha garantito confronti fedeli tra predizioni e maschere originali, migliorando l'affidabilità delle metriche ottenute.

## 6.2 Risultato Finale e Implicazioni Cliniche

Il modello finale, sviluppato **nell'esperimento 21**, ha raggiunto un **Dice score di 0.9124**, **arrivando fino a un Dice score di 0.9161 con il closing morfologico finale**. Dimostrando così un'elevata accuratezza predittiva. Il valore di questa performance risiede non solo nella metrica in sé, ma nella sua coerenza con una pipeline sperimentale **solida, riproducibile e scientificamente fondata**.

Dal punto di vista clinico, i risultati ottenuti hanno implicazioni rilevanti. Automatizzare la **segmentazione del tessuto mammario**, del **fibroghiandolare (FGT)** e dei **vasi sanguigni** consente di ridurre la **variabilità inter-operatori**, migliorare la **velocità di refertazione** e supportare nuove forme di **analisi quantitativa**. In particolare, la stima automatizzata della **densità mammaria** può influenzare in modo diretto la **valutazione del rischio oncologico**, abilitando percorsi diagnostici più personalizzati e precoci.

Inoltre, la presenza di maschere segmentate consente di contestualizzare meglio le eventuali lesioni, potenziando strumenti di **localizzazione assistita** e agevolando la **pianificazione preoperatoria**. Questo può essere utile, ad esempio, per identificare la **relazione tra lesioni e strutture vascolari** o valutare la **distribuzione topografica del FGT** rispetto a margini chirurgici.

## 6.3 Possibili Downstream Task

Le potenzialità applicative del sistema sviluppato non si limitano alla segmentazione primaria, ma si estendono verso numerosi **downstream task**. Tra questi rientrano la possibilità di costruire **modelli predittivi di rischio** oncologico basati su **caratteristiche morfometriche** derivate dalle segmentazioni, l'integrazione del sistema all'interno di pipeline per **registrazione multimodale** (es. MRI + mammografia) e la generazione automatica di **report anatomici strutturati** per supportare la refertazione radiologica.

Un ulteriore ambito di applicazione è rappresentato dalla **quantificazione longitudinale**: in presenza di acquisizioni ripetute, il modello potrebbe consentire il monitoraggio dell'evoluzione del tessuto mammario nel tempo, supportando la valutazione dell'efficacia terapeutica o la diagnosi precoce in soggetti a rischio.

## 6.4 Prospettive di Sviluppo Futuro

Tra gli sviluppi futuri più promettenti, si colloca la possibilità di estendere l'addestramento del modello a **dataset multi-istituzionali**, affrontando il problema della **generalizzazione cross-center**. Per ottenere prestazioni robuste su scanner diversi o su popolazioni eterogenee, sarà necessario introdurre meccanismi di **domain adaptation**, **normalizzazione avanzata** o **data augmentation realistica**.

Dal punto di vista architetturale, una direzione interessante è l'esplorazione di **modelli ibridi CNN-Transformer**, che combinano la capacità di catturare **dipendenze locali** con la modellazione di **relazioni globali**. In parallelo, l'ottimizzazione del modello per ambienti a bassa potenza di calcolo, mediante reti **più leggere e meno profonde**, potrebbe favorire la sua implementazione diretta in contesti clinici.

## 6.5 Riflessione Conclusiva

Il presente lavoro ha rappresentato un'opportunità significativa di formazione e cresciuta all'interno del percorso accademico, consentendomi di confrontarmi in modo diretto con una problematica concreta e di rilevanza clinica: la **segmentazione automatica** di immagini MRI della mammella. Tale esperienza si è rivelata estremamente formativa, **non solo per il consolidamento delle competenze tecniche già acquisite in ambito accademico, ma soprattutto per lo sviluppo di un approccio sperimentale fondato su rigore metodologico, autonomia operativa e capacità critica**.

La progettazione e l'ottimizzazione della pipeline di segmentazione sono state affrontate secondo una prospettiva **graduale** e **iterativa**. A partire da una configurazione di riferimento, sono state progressivamente introdotte modifiche agli iperparametri, all'architettura del modello, alle tecniche di normalizzazione, alle funzioni di loss e alle strategie di validazione. Ogni sperimentazione ha richiesto **un'attenta analisi dei risultati**, la verifica dei dati in output, la progettazione di strumenti di debugging specifici e, spesso, la necessità di **risolvere problematiche non documentate**, tipiche del lavoro su dataset complessi e ambienti computazionali condivisi.

Tali aspetti hanno richiesto non soltanto competenze tecniche, ma anche una progressiva maturazione sul piano dell'organizzazione del lavoro sperimentale, della documentazione e della gestione degli output in modo riproducibile.

Dal punto di vista metodologico, si è trattato di un progetto orientato all'evidenza, in cui ciascuna decisione è stata motivata da osservazioni empiriche e rafforzata da test sistematici. In tal senso, la realizzazione di una pipeline di valutazione basata su **trasformazioni inverse** ha rappresentato un passaggio cruciale, migliorando sensibilmente l'affidabilità delle metriche e consentendo un confronto più aderente alla realtà clinica. La metrica finale, un **Dice score pari a 0.9161, ottenuta nel contesto dell'ultimo esperimento, è il risultato di un lungo processo di ottimizzazione e rappresenta una sintesi significativa delle scelte mature**.

Più in generale, questa tesi ha costituito per me un'esperienza di avvicinamento concreto alla **ricerca applicata**, permettendomi di comprendere più a fondo le potenzialità e le responsabilità legate all'uso **dell'intelligenza artificiale in ambito sanitario**. È emersa in particolare l'importanza della trasparenza, della tracciabilità e della validazione nella costruzione di modelli destinati a interagire con dati clinici sensibili.

**Concludo dunque questa esperienza con la consapevolezza di aver maturato un insieme articolato di competenze, non solo sul piano tecnico, ma anche sotto il profilo metodologico e professionale.** La capacità di analizzare criticamente i risultati, documentare il processo sperimentale e adattare le soluzioni a vincoli reali rappresenta, a mio avviso, uno degli elementi più preziosi di questo percorso.

# *Ringraziamenti*

*Desidero esprimere la mia più sincera gratitudine al Prof. Alessandro Bria, relatore di questo lavoro, per la sua disponibilità, guida costante e preziosi consigli durante tutto il percorso di tirocinio e la redazione della tesi.*

*Un ringraziamento particolare va inoltre all'ing. Marco Cantone, correlatore di questa tesi, per il supporto tecnico, la competenza e la disponibilità dimostrata nell'accompagnarmi lungo le diverse fasi sperimentali. I suoi suggerimenti concreti e il costante confronto hanno contribuito in modo significativo alla buona riuscita di questo lavoro.*

*Colgo infine l'occasione per ringraziare tutte le persone che, direttamente o indirettamente, mi hanno accompagnato con il loro incoraggiamento.*

*A tutti loro, va la mia più profonda riconoscenza.*

*Questo lavoro rappresenta per me non solo la conclusione di un percorso di studi, ma anche la sintesi di impegno, pazienza e costanza che ho cercato di mantenere lungo tutta la strada. Per questo motivo, un ringraziamento va a me stesso, per non essermi arreso di fronte alle difficoltà e per aver creduto fino in fondo nella possibilità di arrivare a questo traguardo.*

# Elenco delle tabelle

5.1	Configurazioni principali dei modelli sperimentalni con architetture e parametri chiave. . . . .	23
5.2	Configurazione del primo esperimento (baseline) con architettura U-Net 3D e risultati ottenuti. . . . .	24
5.3	Risultati degli esperimenti 3-4 con architettura U-Net 3D più profonda. L'aggiunta di un quinto blocco convoluzionale ha portato a significativi miglioramenti nelle metriche. . . . .	24
5.4	Analisi comparativa delle modifiche introdotte dall'EXP 5. Tutti i cambiamenti hanno contribuito al miglioramento delle performance. . . . .	25
5.5	Confronto sistematico degli approcci dimensionali. I valori mostrano come il resize completo produca i migliori risultati, mentre il padding peggiora le performance. . . . .	26
5.6	Progressione prestazionale con scala cromatica: dal rosso (baseline) al blu (miglior risultato). I $\Delta$ verdi mostrano il miglioramento cumulativo, mentre il blu evidenzia il picco prestazionale (+41.1% rispetto alla baseline). . . . .	27
5.7	Analisi dettagliata dei risultati finali. La colonna $\Delta$ mostra: per EXP 16 la sovrastima rispetto alla nuova metodologia, per EXP 19-21 la variazione rispetto al benchmark corretto (EXP 20). . . . .	30

# Elenco delle figure

1.1	Immagine 3D di un cervello catturata tramite una MRI. [1] . . . . .	4
1.2	Logo PyTorch. [10] . . . . .	5
1.3	Esempio di output del comando <code>nvidia-smi</code> per monitorare le GPU disponibili. . . . .	7
2.1	Outer surface segmentation of breast MRI image. [2] . . . . .	9
2.2	Schema concettuale dell'architettura U-Net. . . . .	10
2.3	Overview dei dati di input e la U-NET usata per la segmentazione del seno. [8] . . . . .	12
3.1	Fast Breast MRI Available to Women with Average Breast Cancer Risk — Duke Health [3] . . . . .	14
3.2	Grafico a torta della suddivisione del dataset . . . . .	15
4.1	Esempio di pipeline sperimentale con configurazione YAML. . . . .	17
5.1	Andamento del learning rate durante il training, con warmup iniziale e successiva discesa coseno. . . . .	25
5.2	Schema dell'Attention U-Net. Le frecce indicano il flusso di informazioni tra i moduli di encoder e decoder, con l'integrazione del modulo di attenzione. . . . .	28
5.3	Andamento delle funzioni di loss durante il training dell'esperimento finale(EXP 21). . . . .	29
5.4	Andamento delle metriche di valutazione durante il training dell'esperimento finale(EXP 21). . . . .	29
5.5	Confronto Dice score tra tutti gli esperimenti effettuati. . . . .	29
5.6	Esempi di segmentazione ottenuti con il modello finale (EXP 21). Le immagini mostrano le predizioni su diverse slice del test set, evidenziando la capacità del modello di identificare correttamente le aree di interesse.	30
5.7	Esempio di segmentazione prima e dopo l'applicazione del closing morfologico. Le immagini mostrano come il closing migliori la coesione delle strutture segmentate, eliminando piccole discontinuità e rendendo le maschere più uniformi. . . . .	32

# Bibliografia

- [1] *Brain MRI 3D*. URL: [https://www.researchgate.net/figure/D-image-of-a-brain-obtained-from-MRI-The-markers-show-the-end-points-of-the-rays\\_fig4\\_221197073](https://www.researchgate.net/figure/D-image-of-a-brain-obtained-from-MRI-The-markers-show-the-end-points-of-the-rays_fig4_221197073).
- [2] *Breast MRI 3D*. URL: <https://discourse.slicer.org/t/outer-surface-segmentation-of-breast-mri-image/33817>.
- [3] *Breast MRI Machine*. URL: <https://www.dukehealth.org/blog/fast-breast-mri-available-women-average-breast-cancer-risk>.
- [4] Marco Cantone. *Experimental Framework for 3D MRI Segmentation*. Unpublished internal software, University of Cassino. 2025.
- [5] M. Jorge Cardoso, Wenqi Li, Tom Brown et al. *MONAI: An open-source framework for deep learning in healthcare*. arXiv preprint arXiv:2211.02701. 2022. URL: <https://arxiv.org/abs/2211.02701>.
- [6] Jin Chen et al. *TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation*. arXiv preprint arXiv:2102.04306. 2021. URL: <https://arxiv.org/abs/2102.04306>.
- [7] Kenneth Clark, Brian Vendt, Kirk Smith et al. *The Duke-Breast-Cancer-MRI Dataset*. The Cancer Imaging Archive (TCIA). <https://doi.org/10.7937/TCIA.2019.4VAFYFM9>. 2020.
- [8] Christopher O. Lew et al. “A publicly available deep learning model and dataset for segmentation of breast, fibroglandular tissue, and vessels in breast MRI”. In: *Scientific Reports* 14.1 (2024), p. 5383. DOI: 10.1038/s41598-024-54048-2. URL: <https://doi.org/10.1038/s41598-024-54048-2>.
- [9] Ozan Oktay et al. “Attention U-Net: Learning Where to Look for the Pancreas”. In: *arXiv preprint arXiv:1804.03999*. 2018.
- [10] *PyTorch Logo*. URL: <https://pytorch.org/>.