Questa è la Dedica*:*
ognuno può scrivere quello che vuole,
anche nulla . . .

Introduzione

Questa è l'introduzione.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Rumore e Denoising

1.1 Imaging

Con il termine **medical imaging** ci si riferisce alla generica tecnica o processo attraveso il quale è possibile osservare un' area di un organismo non visibile dall'esterno; per analisi cliniche o una rappresentazione visiva delle funzioni di organi e tessuti.
Una **radiografia** è una tecnica di imaging che produce un radiogramma. Tale tecnica si basa sull' utilizzo di un fascio di **fotoni**, come raggi X o raggi Gamma, generati da una sorgente e catturati da un **recettore**.
Durante il passaggio per il corpo interposto tra la sorgente e il recettore, i raggi sono assorbiti o attenutati in maniera differente l'uno dall'altro creando un insieme di raggi di diversa intensità. Il recettore trasforma questo insieme di raggi di diversa concentrazione in un immagine "in negativo" del corpo, essendo catturati dal recettore solo i fotoni che non vengono assorbiti dall' oggetto.
L'assorbimento dovuto alle diverse composizioni elementali del corpo gioca un ruolo fondamentale nell' imaging medico. Il processo di assorbimento è proporzionale al **numero atomico** della materia attraversata e questo permette di distinguere la materia grigia dalla materia bianca nell'immagine generata. Per esempio tra il sangue e il muscolo non è osservabile un contrasto

elevato, essendo la densità degli elementi che li compongono molto simile. Per generare un contrasto tra i due è necessario iniettare un liquido di contrasto composto da elementi con un alto numero atomico durante l'esposizione ai raggi.
I recettori localizzati in maniera opposta alla sorgente dei raggi, ricevono i fasci di fotoni e generano un segnale elettrico. Questo segnale elettrico deve essere convertito in un segnale digitale da un convertitore analogico/digitale e successivamente in un immagine. [3].

1.2 Rumore

Ci sono molti elementi che possono influenzare la qualità dell'immagine costruita [3, 12], i fattori principali sono :

- Blurring: l' immagine potrebbe risultare sfocata per diverse ragioni, i motivi principali sono:
  1. Uso improprio della strumentazione.
  2. Movimento dell paziente dovuto al respiro o al battito del cuore.
  3. Fluttuazioni dei valori dei pixel dell' immagine in una regione di materiale uniforme.
  4. Gli algoritmi usati per filtrare i raggi sfocano l'immagine.
- Field of View (FOV): La dimensione della regione anatomica inclusa nella scansione viene detta field of view. Se la dimensione della regione è troppo grande o troppo piccola, la qualità dell' immagine potrebbe risultare degradata.
- Artefatti: Gli artefatti sono distorsioni o errori nell' immagine digitale che non dipendono dall' oggetto scansionato. Gli artefatti più comuni sono:
  1. Beam Hardening: Beam Hardening avviene quando l'energia media di un raggio passante per il paziente aumenta. Per evitare che

ciò avvenga alcune soluzioni possono essere messe in pratica, aumentare il kvp(peak kilovoltage), decrementare la grandezza delle slice(in caso di tomografia computerizzata), pre-filtrare i raggi, evitare regioni ad alto assorbimento, soluzioni algorimiche.

2. Artefatti metallici: Materiali metallici possono causare soprattutto nelle CT i cosiddetti streaking artifacts (fasci di luce), dovuti dal blocco delle informazioni di proiezione. Per ridurre questo fenomeno è necessario rimuovere il materiale metallico se possibile.
3. Movimento del paziente: Il movimenti volontari o involontari del paziente possono produrre streaking artifacts. Si può ridurre il tempo di scansione e migliorare il posizionamento del paziente.
4. Software e Hardware: Artefatti derivanti da implementazioni o configurazioni scorrette di algotimi di costruzione dell'immagine digitale o degradazione dei componenti elettronici e meccanici.

- Rumore Visivo: Il rumore visivo è una variazione casuale della luminosità dei singoli pixel dell' immagine. Questo tipo di rumore deriva di solito da errori nell' acquisizione, trasmissione del segnale ed errori computazionali matematici. Questo tipo di rumore ha una maggiore influenza su soggetti a basso contrasto.

La dose di radiazioni a cui è sottoposto il paziente è uno dei fattori più importanti nella generazione di rumore. Per la sicurezza del paziente è necessario minimizzare il tempo di esposizione alle radiazioni, ma la conseguenza principale nel ridurre la dose di radiazioni è un aumento del rumore dell'immagine facendo diminuire la qualità della diagnosi.
Ci sono due approcci principali per cercare di ridurre il rumore visivo senza sacrificare la qualità dell'immagine:

1. Facendo uno studio ed una ottimizzazione della dose di radiazioni le immagini a bassa dose di radiazioni possono essere migliorate.
2. La qualità dell' immagine può essere migliorata sviluppando algoritmi per la rimozione del rumore nelle immagini. Questi algoritmi possono

essere successivamente utilizzati per ridurre la dose di radiazioni. Il processo di rimozione di rumore da immagini è conosciuto con il termine di **image denoising**.

Le radiografie sono caratterizzate da una sensitibilità ai contrasti che permettono di distinguere tra i diversi tessuti molli del corpo umano. Questa caratteristica è particolarmente influenzata dal rumore che nuoce alla visualizzazione delle strutture a basso contrasto.
Prima di considerare tecniche di image denoising é necessario comprendere quali sono le sorgenti principali di rumore visivo:

Rumore randomico e statistico: Può essere generato dalla ricezione di un numero finito di quanti (pacchetti di energia) di raggi da parte del recettore. Durante il processo non c'è nessuna forza che distribuisce i fotoni in maniera uniforme sulla superfice, un area del recettore potrebbe percepire più fotoni di un altra. La causa principale di questo rumore è la cosiddetta radiazione secondaria [12], prodotta dall'interazione dei fotoni emessi dal generatore con il corpo in scansione, disperdendo ulteriormente in maniera casuale le particelle. Il risultato del rumore viene percepito come una fluttuazione nella densità dell' immagine, che varia in maniera randomica [3]. Questo tipo di rumore è anche chiamato **rumore quantico**.

Rumore Elettrico: Il processo di un circuito elettrico che riceve segnali analogici può generare un disturbo del segnale. La sorgente più comune di rumore è il rumore termico intrinseco di ogni elemento dissipativo (es. resistori) che si trovi ad una temperatura diversa dallo zero assoluto [1].

Errori di arrotondamento: I segnali analogici sono convertiti in segnali digitali usando algoritmi di quantizzazione del segnale. Per ricostruire l'immagine il segnale ora discreto viene elaborato da un calcolatore. La rappresentazione in memoria del segnale deve essere necessariamente finita, essendo limitata dal numero di bit usati dal sistema numerico implementato dal computer per il calcolo. La computazione matematica non è dunque possibile senza errori di arrotondamento.

Generalmente il rumore nella ricostruzione di immagini mediche viene introdotto per due motivi. Il primo é un rumore variabile dovuto al rumore elettrico e da errori di arrotondamento e può essere modellato con un semplice rumore additivo. Il secondo è il rumore dovuto dalle fluttuazioni di fotoni percepiti nelle diverse zone del recettore, più complesso da modellizzare.
Il rumore può essere calcolato statisticamente utilizzando la deviazione standard dell'intensità dei pixel in una regione fisica unifome dell' immagine. L'analisi statistica dei valori dei pixel produce una distribuzione con forma a campana. La variazione standard, cioè la misura della variazione dei pixel nella zona di interesse, rappresenta il livello di rumore nell'immagine. Un valore alto della deviazione standard implica un alto livello di rumore. Questa distribuzione statistica mostra mostra che il valore medio del rumore è uguale alla varianza del rumore e può essere analizzato che il rapporto segnale-rumore è proporzionale alla radice quadrata della dose di raggi emessi. Dunque la quantit'à relativa di rumore quantico percepita dai recettori diminuisce all'aumentare della dose di raggi emessi. Il problema di identficare la distribuzione esatta del rumore deriva dal fatto che tutti i passi intermedi nella generazione dell'immagine introducono una correlazione statistica con i valori contenenti rumore catturati dal recettore. A causa di queste dipendenze la distribuzione esatta del rumore nell' immagine finale risulta sconosciuta. Sperimentalmente si è potuto osservare come il rumore quantico sia modellizzabile con una distribuzione di Poisson mentre il rumore termico con una distribuzione di Gauss [12, 1, 3].

1.3 Denoising

1.3.1 Approccio Classico

Con image denoising si intente il problema di rimuovere rumori e distorsioni da una immagine. Generalmente si considera l'immagine rumorosa come una somma tra l'immagine originale e una componente di rumore, questo comporta che l'obbiettivo delle tecniche di denoising sia quello di ridurre

la componente rumorosa cercando di mantenere nell'immagine pulita le caratteristiche dell' immagine originale.
Quando si tratta di denoising applicato a imaging medico l'obbiettivo del denoising è quello di ridurre il rumore mantendendo i dettagli clinici in modo tale da mantenere l'immagine utile per una corretta diagnosi. Nel tempo sono stati sviluppati filtri di lisciamento (smoothing) e di sharpening. I primi in caso di alti livelli di rumore tendono a sfocare l'immagine e a non preservare i bordi dell'oggetto, mentre i filtri di sharpening non eccellono nel mantenere piccoli dettagli dell' immagine [4]. In caso di denoising per analisi di immagini mediche questo presenta un grande problema in quando mantenere più dettagli possibili dell'immagine originale è di fondamentale importanza.
I problemi principali nel denoising di immagini mediche sono:

- Le regioni piatte (zone aventi una variazione minima dei valori dei pixel) dell'immagine devono rimanere zone piatte.
- I contorni degli oggetti devono rimanere preservati.
- I dettagli di struttura dell'oggetto non devono essere persi.
- Il contrasto dell'immagine deve essere preservato.
- Non devono formarsi nuovo artefatti.

Generalmente ci sono due approcci per il denoising di immagini : spatial domain filtering e transform domain filtering [10].

Filtri Spaziali

I filtri spaziali agiscono applicando un filtro direttamente all' immagine con rumore, e possono essere definiti in filtri lineari e non lineari.

Filtri Lineari

I filtri lineari come il filtro medio (Mean filter) o il filtro di Wiener tendono a sfocare i bordi dell' oggetto, distruggere linee e dettagli nell'immagine, e

generalmente non performano bene nel caso di rumore dipendente dal segnale come il rumore poissoniano. L'idea del filtro medio è molto semplice, consiste nel sostituire ogni valore di ogni pixel con il valore medio dei pixel vicini, incluso se stesso. Il filto di Wiener invece richiede informazioni sullo spettro del rumore e dell'immagine originale.

Filtri Non Lineari

I filtri non lineari invece rimuovono il rumore senza cercare di identificarlo. Per cercare di risolvere i problemi dei filtri lineari una serie di filtri non lineari sono stati sviluppati: filtro mediano, filtro mediano pesato e filtro mediano rilassato.

Transform Domain Filtering

A differenza dei metodi spaziali, i metodi di trasformazione prima ottengono una trasformazione dell' immagine rumorosa e successivamente applicano la procedura di denoising sulla trasformazione. Si dividono in fitri Data adaptive e Non-Data adaptive.

1.3.2 Deep Learning

I metodi di **Machine Learning** si dividono in supervisionati, semi-supervisionati e non supervisionati. I metodi supervisionati utilizzano delle informazioni associate all' input (labels) per imparare dei parametri e trainare il modello di denoising. Per esempio dato il seguente modello

$$y = x + \mu$$

dove x, y e μ rappresentano rispettivamente l' immagine originale, l' immagine rumorosa e un rumore additivo Gaussiano (AWGN) con deviazione standard σ. Il modello di denoising utilizza le coppie $\{x_k, y_k\}_{k=1}^{N}$ per imparare i parametri della funzione f che cerca di rimuovere il rumore, dove x_k e y_k sono la k-esima immagine originale e la k-esima immagine con rumore; N

é il numero di immagini totali. Il processo può essere descritto come

$$x_k = f(y_k, \theta, m)$$

dove θ sono i parametri e m la quantità di rumore.
I metodi non supervisionati utilizzano gli esempi di training per cercare patterns nel data al posto di fare label matching come nel caso supervisionato. Nel caso dei metodi semi-supervisionati invece si cerca di imparare una distribuzione del data in modo da poter assegnare labels ai dati che non ne hanno.
Le **Reti Neurali** sono alla base dei metodi di machine learning, che a loro volta sono alla base dei metodi di deep learning. La maggior parte delle reti neurali sono composte da neuroni, input X, funzioni di attivazione ϕ, pesi $W = [W^0, W^1, ..., W^{n-1}]$ e bias $b = [b^0, b^1, b^n - 1]$. A differenza dei modelli lineari come la regressione lineare e la regressione logistica, che possono modellizzare solo funzioni lineari senza comprendere l'interazioni tra due qualsiasi variabili di input, le reti neurali non applicano un modello lineare direttamente ad $\mathbf{x}$. Viene utilizzata invece una trasformazione $\phi(\boldsymbol{x})$, dove ϕ è una trasformazione non lineare.

$$y = f(\boldsymbol{X}; \boldsymbol{b}, \boldsymbol{W}) = \phi(\boldsymbol{W}^T \boldsymbol{X} + \boldsymbol{b}) \tag{1.1}$$

Se la rete neurale è composta da più "livelli" di profondità, dati da funzioni di attivazione e pesi diversi, la rete viene definita con il termine **multilayer perceptrons** (MLPs) oppure **Deep feedforward networks** o **feedforward neural networks**.
L' obbiettivo delle reti feedforward è quello di approssimare una funzione f^*. Per esempio nel caso di un classificatore , $y = f^*(\mathbf{x})$ mappa un input $\mathbf{x}$ ad una categoria y. Una rete feedforward definisce un mapping $\boldsymbol{y} = f(\boldsymbol{x}; \boldsymbol{\theta})$ ed impara il valore dei parametri $\boldsymbol{\theta}$ che permettono a f di meglio approssimare f^*. Queste reti sono chiamate **feedforward** perchè le informazioni fluiscono dall' input $\mathbf{x}$, attraverso le funzioni intermedie che compongono f, per arrivare all' output $\mathbf{y}$. Non ci sono connessioni **feedback** nelle quali l'output del

modelo viene restituito al modello stesso.
Le feedforward neural networks vengono chiamate **networks** perchè tipicamente sono rappresentate dalla composizione di funzioni diverse. Per esempio potremmo avere tre funzioni f^1, f^2, f^3 composte tra di loro per formare $f(\boldsymbol{x}) = f^3(f^2(f^1(\boldsymbol{x})))$. In questo caso, f^1 è chiamato il primo layer, f^2 il secondo layer e cosi via. La lunghezza della catena definisce la **depth** del modello, da questa terminologia deriva il termine "deep learning". L'ultimo layer della rete neurale viene detto **output layer**. Durante il training della rete neurale , l' obbiettivo è quello di far assomigliare $f(\boldsymbol{x})$ a $f^*(\boldsymbol{x})$. Il data di training è composto da dei punti **x**, ognuno associato ad una label $y \approx f^*(x)$. Il data di training specifica quale deve essere il comportamento dell' output layer per ogni punto **x**, cioè produrre valori vicini a y. L' algoritmo di training deve decidere come variare i parametri degli altri layer in modo tale da ottenere la migliore approssimazione di f^*, per fare questo viene utilizzata una **loss function** che descrive quanto il valore in output della rete su input x sia "distante" secondo una qualche metrica dalla label associata. Siccome l'output dei layer intermedi non mostrano l'output desiderato, questi layers vengono chiamati **hidden layers**.

In fine questo tipo di reti vengono definite *neurali* perchè sono ispirate alle neuroscienze. Ogni hidden layer della rete prende tipicamente in input un vettore di valori. La grandezza di questo vettore determina la **width** della rete. Ogni elemento di questo vettore può essere interpretato come un segnale per un neurone. Invece di pensare ad un hidden layer come una funzione che mappa un vettore in un altro, possiamo considerare ogni layer come composti da un insieme di **units** che operano in parallelo. Ogni unit assomiglia ad un neurone, nel senso che riceve degli input da molte altre unit e computa la propria funzione di attivazione. Nonostante questa intuizione, non bisogna pensare alle reti neurali come modellizzazioni del cervello, ma piuttosto come macchine che approssimano funzioni in modo tale da raggiungere una generalizzazione statistica del problema dato un insieme di esempi.

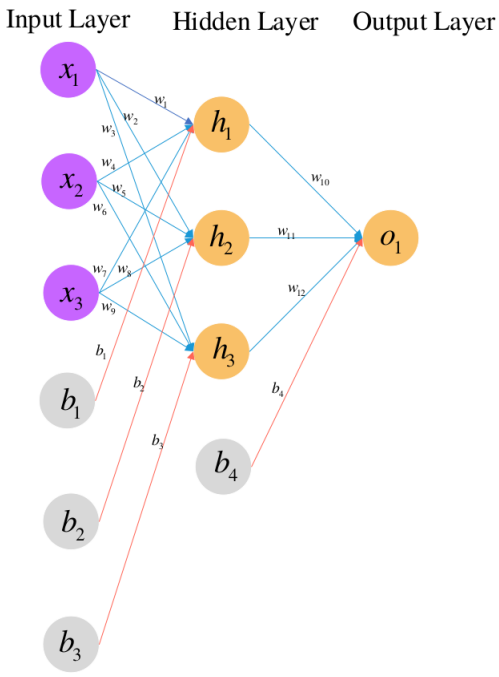

Figura 1.1: Rete neurale a due layer

Una rete neurale può essere espressa nel seguente modo :

$$f(X;W;b) = \phi^{n-1}(W^{n-1}\phi^{n-2}(W^{n-2}...\phi^{0}(W^{0}X + b^{0})...b^{n-2}) + b^{n-1}) \quad (1.2)$$

La rete neurale in figura 1.1 completamente connessa presenta due layer : un hidden layer e un layer di output (Il layer di input solitamente non viene considerato come layer). I parametri x_1, x_2, x_3 e o_1 rappresentano l'input e l'output della rete. $w_1, w_2, ..., w_{12}$ sono i pesi e b_1, b_2, b_3, b_4 è il bias. Per esempio, l'output del neurone h_1 si ottiene come segue :

$$f(z_{h1}) = f(w1x1 + w4x2 + w7x3 + b1) \quad (1.3)$$

$$o(h1) = f(z_{h1}) \quad (1.4)$$

Una volta calcolato l'output o_1 la rete usa un algoritmo di back propagation e una funzione di loss per imparare i parametri [7]. Nell' ambito dell'

elaborazioni delle immagini una tipologia di rete neurale molto utilizzata è la **Convolutional Neural Network** (CNN).
Le Convolutional Network sono delle reti neurali specializzate per processare del data con una topologia a griglia. Un esempio sono le serie temporali, che possono essere rappresentate come una griglia 1D prendendo osservazioni ad intervalli regolari, e immagini, rappresentabili come una griglia 2D di pixel. Il nome "convolutional network" indica che la rete utilizza una particolare operazione matematica lineare denominata **convoluzione**. Utilizziamo un esempio per spiegare l'operazione di convoluzione : supponiamo di voler tracciare la posizione di un astronave con un sensore laser. Il sensore restituisce un singolo output $x(t)$, la posizione dell' astronave al tempo t. Sia x che t sono valori reali. Supponiamo che il sensore restituisca output rumorosi. Per ottenere una stima meno rumorosa della posizione dell' astronave vogliamo utilizzare una media pesata tra le misurazioni dando più importanza alle misurazioni recenti. Possiamo utilizzare una funzione di pesatura $w(\alpha)$ dove α è il tempo della misurazione. Applicando questa media pesata ad ogni istante si ottiene una nuova funzione s che indica una stima regolarizzata della posizione.

$$s(t) = \int x(a)w(t-a)da \tag{1.5}$$

Questa operazione viene detta **convoluzione**. L' operazione viene tipicamente denotata con un asterisco.

$$s(t) = (x * w)(t)$$

Nel nostro esempio, w deve essere una funzione di densità valida, altrimenti l'output non sarebbe una media pesata. Ma questo non è necessario per altri esempi, in generale la convoluzione è devinita per ogni coppia di funzioni dove l'integrale 1.5 è definito; e l'operazione può essere usata per altri scopi oltre alla media pesata. Nella terminologia delle convoluzioni il primo termine, in questo caso la funzione x, prende il nome di **input**. Il secondo argomento viene detto **kernel**. L'output viene indicato con il termine **feature map**. Nelle applicazioni di machine learning l'input è solitamente discretizzato e rappresentato da un array multidimensionale, e il kernel

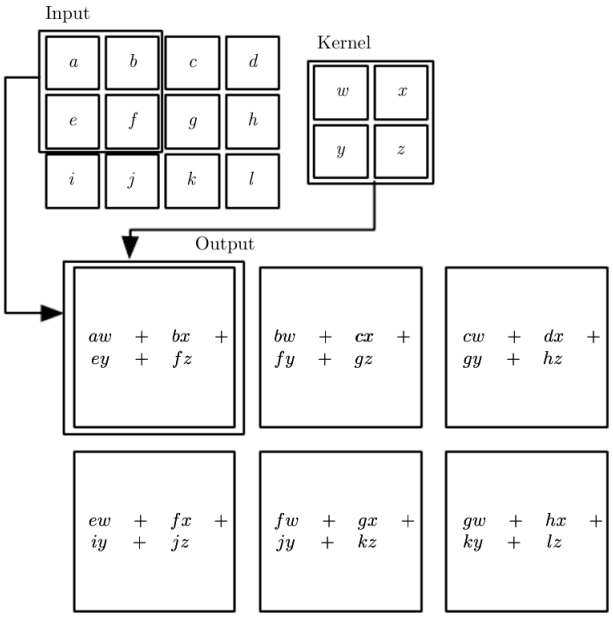

Figura 1.2: Un esempio di convoluzione 2D

è rappresentato da un array multidimensionale di parametri adattati dall' algoritmo di apprendimento. Nel caso di immagini vogliamo definire la convoluzione su più assi. Per esempio con un immagine due-dimensionale I come input, vogliamo utilizzare un kernel due-dimensionale K :

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n)$$

Le CNNs per come sono definite sono in grado di sfruttare le correlazioni spaziali dell' immagine [7, 14, 6], preservando bordi e dettagli.
Un avanzamento importante alle CNNs è stato dato dalla funzione di attivazione ReLU [15, 14], migliorando la velocità della discesa stocastica del gradiente (SGD) [7], un metodo molto utilizzato per il training dei parametri del kernel. Sperimentalmente è stato osservato che se la rete è molto profonda, durante il training si possono presentare fenomini di annullameto o esplosione del gradiente; e se la rete è molto ampia, si può presentare il fenomeno dell' overfitting [7]. Per risolvere questi problemi nel 2016 è stato

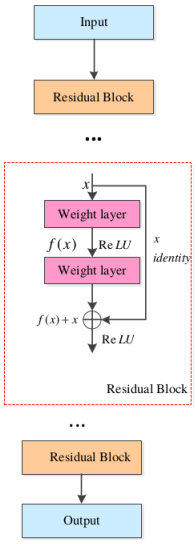

Figura 1.3: Un esempio di blocco residuo

proposto un particolare tipo di rete, ResNet [8]. ResNet introduce il concetto di *skip connection*: l'input di un layer è aggiunto all'output di un layer successivo. L' obbiettivo di una rete neurale è quello di modellizzare una funzione target $h(\mathbf{x})$. Aggiungendo l' input $\mathbf{x}$ all' output della rete, allora la rete è costretta ad imparare una funzione $f(\mathbf{x}) = h(\mathbf{x}) - \mathbf{x}$. Questa tecnica prende il nome di *residual learning*. Figura 1.3.

Quando si inizializza una rete neurale i suoi pesi sono vicini a zero e la rete restituisce in output valori vicini a zero. Aggiungendo una skip connection, la rete restituirà in output una copia del suo input; modellando inizialmente la funzione identità. Se la funzione target è abbastanza vicina alla funzione di identità (particolarmente vero per il problema del denoising), il residual learning velocizza l' apprendimento dei parametri in maniera considerevole.

Capitolo 2

Denoising Residual Network

2.1 Modellizzazione del Rumore

Lo scopo delle tecniche di denoising è quello di generare una immagine pulita X dato un input rumoroso Y, ottenuto secondo un modello di degradazione $Y = D(X)$. Per il rumore Gaussiano additivo (AWGN), il valore del pixel i-esimo è :

$$y_i = D(x_i) = x_i + n_i \tag{2.1}$$

Dove $n_i \sim \mathcal{N}(0, \sigma^2)$ è un rumore Gaussiano i.i.d con media zero e varianza σ^2. AWGN è usato per modellare rumore termico indipendente dal segnale elaborato e altri errori introdotti dall' elaborazione dell' immagine come descritto in precedenza. La degradazione dell'immagine dovuta al cosidetto rumore quantico viene invece modellata usando un rumore di Poisson dipendente dal segnale :

$$y_i = D(x_i) = p_i, \quad p_i \sim \mathcal{P}(x_i) \tag{2.2}$$

dove $\mathcal{P}(x_i)$ è una variablie aleatoria di Poisson con media x_i. Siccome la distribuzione di Poisson tende alla distribuzion di Gauss per valori abbastanza grandi di λ , $\mathcal{P}(\lambda) \approx \mathcal{N}(\lambda, \lambda)$, il rumore generato in sistemi che catturano immagini è meglio modellato come un rumore di Poisson con AWGN, chiamato **Poisson-Gaussian** :

$$y_i = D(x_i) = \alpha p_i + n_i, \quad \alpha > 0 \tag{2.3}$$

che è stato verificato da risultati sperimentali [5].

2.2 DN-Resnet

DN-Resnet consiste in una rete neurale composti da blocchi residui (ResBlocks), inseriti gradualmente nella rete durante il training. Questa strategia di training permette alla rete di convergere più velocemente. In aggiunta al convenzionale **errore quadratico medio** (MSE) si utilizza anche un funzione di loss **edge-aware** e una **perceptual loss** basata su Resnet-50 [8]. Aggiungendo in cascata 5 Resblocks la rete raggiunge performance da stato dell' arte su tutti e tre i tipi di rumore, Gaussian, Poisson e Poisson-Gaussian. Rispetto ad altre reti neurali proposte per il problema del denoising, DN-Resnet ha un tempo di training molto minore e può essere eseguita in tempo reale.

Dato un dataset di training $\{X_i, Y_i\}, i = 1, \ldots, N$ con N esempi, l'obbiettivo è quello di imparare un modello S che predice immagini pulite $\hat{X}_i = S(Y_i)$.
L' elemento base di DN-Resnet è un ResBlock semplificato, come mostrato in figura 2.1(a). A differenza del ResBlock standard sono stati rimossi i layer di batch normalization [9] e il layer di ReLU dopo la somma residua, perchè rimuovere questi layer nelle reti residue basate su feature-map non peggiora le performance della rete [11].

DN-Resnet viene costruita concatenando i ResBlock in figura 2.1(b). Per velocizzare il training, viene utilizzato il metodo del **cascade training** [13], che separa il training in stage trainati uno alla volta. Il training di DN-Resnet inizia come un modello di CNN a tre layer. Il primo layer consiste in 64 9×9 filtri. Il secondo layer consiste in 32 5×5. L'ultimo layer è composto da un layer 5×5. Tutte le convoluzioni hanno stride uno, e tutti i pesi sono inizializzati da una distribuzione Gaussiana con $\sigma = 0.001$. Dopo

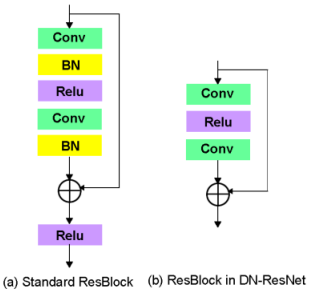

Figura 2.1: (a) Standard ResBlock, (b) ResBlock in DN-Resnet

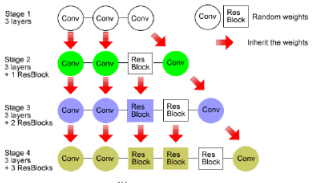

Figura 2.2: Cascade training di DN-Resnet. I cerchi denotano i layer convoluzionari standard, mentri i quadrati i ResBlock.

il training dei tre layer si procede con il cascade training come mostrato in figura 2.2. Quando il training dello stage corrente è finito si passa allo stage successivo, rendendo più profonda la rete. In ogni stage viene aggiunto un ResBlock. Quindi la rete inizia con 3 layers, e si sviluppa in 5 layers, poi 7 layers, etc. Ogni layer convoluzionario nel ResBlock consiste in 32 3×3 filtri, i nuovi layers vengono inseriti prima dell' ultimo layer 5×5. I pesi dei layers gia esistenti vengono ereditati dallo stage precedente, mentre i pesi del nuovo blocco vengono inizializzati randomicamente (Gaussian con $\sigma = 0.001$). Aggiungendo 5 ResBlock, la DN-Resnet risultante avrà $5 \times 2 + 3 = 13$ layers convoluzionari.

2.3 Funzione di loss Edge-aware

Molte reti di denoising puntano a minimizzare l' Errore Quadratico Medio (MSE) sul data di training.

$$\frac{1}{N}\sum_{i=1}^{N}\left\|X_i - \hat{X}_i\right\|^2 \tag{2.4}$$

In questa rete viene utilizzata una funzione di loss dove ai pixel che rappresentano dei bordi dell' oggetto vengono assegnati pesi piu grandi rispetto agli altri pixel.

$$\text{edge-loss} = w \times \frac{1}{N}\sum_{i=1}^{N}\left\|X_i M_i - \hat{X}_i M_i\right\|^2 \tag{2.5}$$

In Eq. 2.5, 2.4, X_i è l' i-esima immagine pulita, $\hat{X}_i$ è i-esima immagine a cui è stato applicato il denoising, M è una edge map, N è il numero di immagini e w è una costante per controllare il trade-off tra i pixel negli edge e nel resto dell' immagine.

Ci sono due vantaggi nell' applicare questa funzione di loss edge-aware. Il primo è che una delle sfide principali nel denoising consiste nella difficolta di riconoscere i bordi dell' oggetto, soprattuto in caso di rumore elevato, mentre aggiungere un vincolo nella funzione di loss risulta molto semplice. Il secondo è che dal punto di vista della qualità dell' immagine, aumentare la precisione del denoising sui bordi dell' oggetto ne aumenta la definizione.

M può essere costruito in due modi, tramite la magnitudine del gradiente ottenuto dai filtri di sobel o tramite una maschera binaria ottenuta impostando una soglia di valore del pixel da superare. I risultati migliori si ottengono tramite i **filtri di sobel**.

L' operatore di sobel calcola un valore approssimato del valore in ogni punto del gradiente della funzione che rappresenta la luminosità dell' immagine. L' operatore trova la direzione lungo il quale si ha il massimo incremento dal chiaro allo scuro, e la velocità con cui avviene il cambiamento lungo questa direzione. Il risultato finale fornisce una misura di quanto bruscamente o gradualmente l' immagine cambia in quel punto, e quindi della probabilità che

quella parte di immagine rappresenti un contorno. In termini matematici, il gradiente di una funzione in due variabili è in ciascun punto dell' immagine un vettore bi-dimensionale le cui componenti sono le derivate del valore della luminosità in direzione orizzontale e verticale. In ciascun punto dell' immagine questo vettore gradiente punta nella direzione di massimo aumento della luminosità, e la lunghezza del vettore corrisponde alla rapidità con cui la luminosità cambia spostandosi in quella direzione. Nelle zone dell' immagine in cui la luminosità è costante l' operatore di Sobel ha valore zero, mentre nei punti posti sui bordi è un vettore orientato attraveso il contorno, che punta nella direzione in cui si passa da valori di scuro a valori di chiaro.

L' operatore applica una operazione di convoluzione utilizzando due kernel 3×3 all' immagine originale per calcolare i valori approssimati delle derivate lungo le due direzioni, orizzonatale e verticale. Chiamando $\mathbf{A}$ l' immagine sorgente, e $\mathbf{G}_x$, $\mathbf{G}_y$ la componente x ed y del gradiente, l' operazione è descritta da :

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \text{e} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A} \tag{2.6}$$

dove $*$ indica l' operazione di convoluzione bi-dimensionale. In ogni punto dell' immagine, le approssimazioni del gradiente possono essere combinate per ottenere la magnitudine del gradiente :

$$\mathbf{G} = \sqrt{\mathbf{G}_x^2 + \mathbf{G}_y{}^2} \tag{2.7}$$

2.4 Perceptual Loss

L' errore quadratico medio 2.4 calcola la distanza pixel per pixel del l'output del modello con l' immagine target. Questa differenza puntuale tende a lisciare i valori dei pixel e ad aumentare la sfocatura dell' immagine. Per ovviare a questi problemi viene introdotta una **multi scales perceptual**

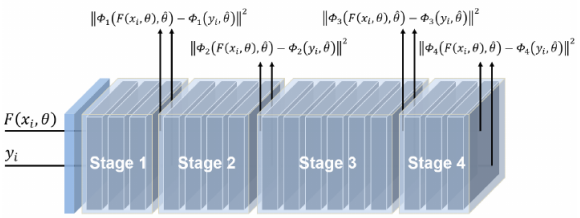

Figura 2.3: multi-scale perceptual loss basata su Resnet-50, composto da 4 stage principali [8]

loss, mostrata nella seguente formula :

$$L_{multi-p} = \frac{1}{NS} \sum_{i=1}^{N} \sum_{s=1}^{S} \left\| \phi_s \left(F\left(x_i, \theta\right), \hat{\theta} \right) - \phi_s \left(y_i, \hat{\theta} \right) \right\|^2 \tag{2.8}$$

In questa formula, x_i è l'input, y_i è il target e N è il numero di immagini. F rappresenta la rete di denoising con parametri θ. ϕ rappresenta un modello con dei pesi fissati pre-trainati $\hat{\theta}$, usato per calcolare la perceptual loss. S è il numero di stage usati nel modello pre-trainato.

Per costruire la perceptual loss è utilizzato Resnet-50 come estrattore delle feature map, in particolare sono stati rimossi i layer di pooling e i layer completamente connessi alla fine della rete, mantenendo solo i layer convoluzionari. Inizialmente viene caricato il modello con i pesi trainati sul dataset ImageNet [2], i pesi vengono "freezati" e resi non trainabili. Durante il calcolo del valore della perceptual loss, sia l'output denoised che l' immagine target vengono passati all' estrattore per fare forward propagation. Vengono scelte le feature map dei quattro stage di ResNet, il quale ognuno dimezza lo spazio dell' immagine, rappresentando caratteristiche spaziali in forme diverse. A questo punto viene utilizzato MSE per misurare la similitudine tra queste feature map. La multi-scale perceptual loss è ottenuta facendo la media tra queste similitudini.

2.5 Compound Loss

Combinando le funzioni di loss descritte in precedenza otteniamo la funzione 2.12, che ha l'obbiettivo di ottenere una similitudine tra i valori dei pixel, mantenendo i dettagli strutturali dell' oggetto e aumentandone la definizione dei contorni.

$$L_{mse} = \frac{1}{N}\sum_{i=1}^{N}\left\|X_i - \hat{X}_i\right\|^2 \tag{2.9}$$

$$L_{edge} = \frac{1}{N}\sum_{i=1}^{N}\left\|X_i M_i - \hat{X}_i M_i\right\|^2 \tag{2.10}$$

$$L_{multi-p} = \frac{1}{NS}\sum_{i=1}^{N}\sum_{s=1}^{S}\left\|\phi_s\left(F\left(x_i,\theta\right),\hat{\theta}\right) - \phi_s\left(y_i,\hat{\theta}\right)\right\|^2 \tag{2.11}$$

$$L_{\text{compound}} = L_{mse} + w_p \cdot L_{\text{multi- } p} + w_e \cdot L_{edge} \tag{2.12}$$

Conclusioni

Queste sono le conclusioni.ssssssss

Bibliografia

[1] Ajay Kumar Boyat e Brijendra Kumar Joshi. "A Review Paper : Noise Models in Digital Image Processing". en. In: *Signal & Image Processing : An International Journal* 6.2 (apr. 2015), pp. 63–75. ISSN: 22293922, 0976710X. DOI: 10.5121/sipij.2015.6206. URL: http://www.aircconline.com/sipij/V6N2/6215sipij06.pdf (visitato il 16/06/2021).

[2] Jia Deng et al. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. ISSN: 1063-6919. Giu. 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[3] Manoj Diwakar e Manoj Kumar. "A review on CT image noise and its denoising". en. In: *Biomedical Signal Processing and Control* 42 (apr. 2018), pp. 73–88. ISSN: 17468094. DOI: 10.1016/j.bspc.2018.01.010. URL: https://linkinghub.elsevier.com/retrieve/pii/S1746809418300107 (visitato il 14/06/2021).

[4] Linwei Fan et al. "Brief review of image denoising techniques". en. In: *Visual Computing for Industry, Biomedicine, and Art* 2.1 (dic. 2019), p. 7. ISSN: 2524-4442. DOI: 10.1186/s42492-019-0016-7. URL: https://vciba.springeropen.com/articles/10.1186/s42492-019-0016-7 (visitato il 17/06/2021).

[5] Alessandro Foi et al. "Practical Poissonian-Gaussian Noise Modeling and Fitting for Single-Image Raw-Data". In: *IEEE Transactions on Image Processing* 17.10 (ott. 2008). Conference Name: IEEE Tran-

sactions on Image Processing, pp. 1737–1754. ISSN: 1941-0042. DOI: 10.1109/TIP.2008.2001399.

[6] Lovedeep Gondara. "Medical image denoising using convolutional denoising autoencoders". en. In: *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)* (dic. 2016). arXiv: 1608.04667, pp. 241–246. DOI: 10.1109/ICDMW.2016.0041. URL: http://arxiv.org/abs/1608.04667 (visitato il 17/06/2021).

[7] Ian Goodfellow, Yoshua Bengio e Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[8] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *arXiv:1512.03385 [cs]* (dic. 2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385 (visitato il 18/06/2021).

[9] Sergey Ioffe e Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *arXiv:1502.03167 [cs]* (mar. 2015). arXiv: 1502.03167. URL: http://arxiv.org/abs/1502.03167 (visitato il 23/06/2021).

[10] Paras Jain e Vipin Tyagi. "A survey of edge-preserving image denoising methods". en. In: *Information Systems Frontiers* 18.1 (feb. 2016), pp. 159–170. ISSN: 1572-9419. DOI: 10.1007/s10796-014-9527-0. URL: https://doi.org/10.1007/s10796-014-9527-0 (visitato il 17/06/2021).

[11] Bee Lim et al. "Enhanced Deep Residual Networks for Single Image Super-Resolution". In: *arXiv:1707.02921 [cs]* (lug. 2017). arXiv: 1707.02921. URL: http://arxiv.org/abs/1707.02921 (visitato il 23/06/2021).

[12] EN Manson et al. "Image Noise in Radiography and Tomography: Causes, Effects and Reduction Techniques". en. In: *Medical Imaging* (), p. 6.

[13] Haoyu Ren, Mostafa El-Khamy e Jungwon Lee. "CT-SRCNN: Cascade Trained and Trimmed Deep Convolutional Neural Networks for Image Super Resolution". In: *arXiv:1711.04048 [cs]* (nov. 2017). arXiv: 1711.04048. URL: `http://arxiv.org/abs/1711.04048` (visitato il 23/06/2021).

[14] Chunwei Tian et al. "Deep Learning on Image Denoising: An overview". en. In: *arXiv:1912.13171 [cs, eess]* (ago. 2020). arXiv: 1912.13171. URL: `http://arxiv.org/abs/1912.13171` (visitato il 17/06/2021).

[15] Bing Xu et al. "Empirical Evaluation of Rectified Activations in Convolutional Network". en. In: *arXiv:1505.00853 [cs, stat]* (nov. 2015). arXiv: 1505.00853. URL: `http://arxiv.org/abs/1505.00853` (visitato il 19/06/2021).

Ringraziamenti

Qui possiamo ringraziare il mondo intero!!!!!!!!!!
Ovviamente solo se uno vuole, non è obbligatorio.