UNIVERSITÀ
degli STUDI
di CATANIA

Inpainting di immagini tramite reti autoencoder

Nome: **Bruno** Cognome: **Montalto** Matricola: **1000016231**

Anno accademico 2022/2023

Indice

1 Introduzione
2 Cosa sono gli autoencoder
3 Analisi dei dati
    3.1 Adattamento dei dataset
4 Progettazione del modello
    4.1 Architettura dell’encoder
    4.2 Architettura del decoder
5 Addestramento
    5.1 Risultati dell’addestramento
6 Conclusioni

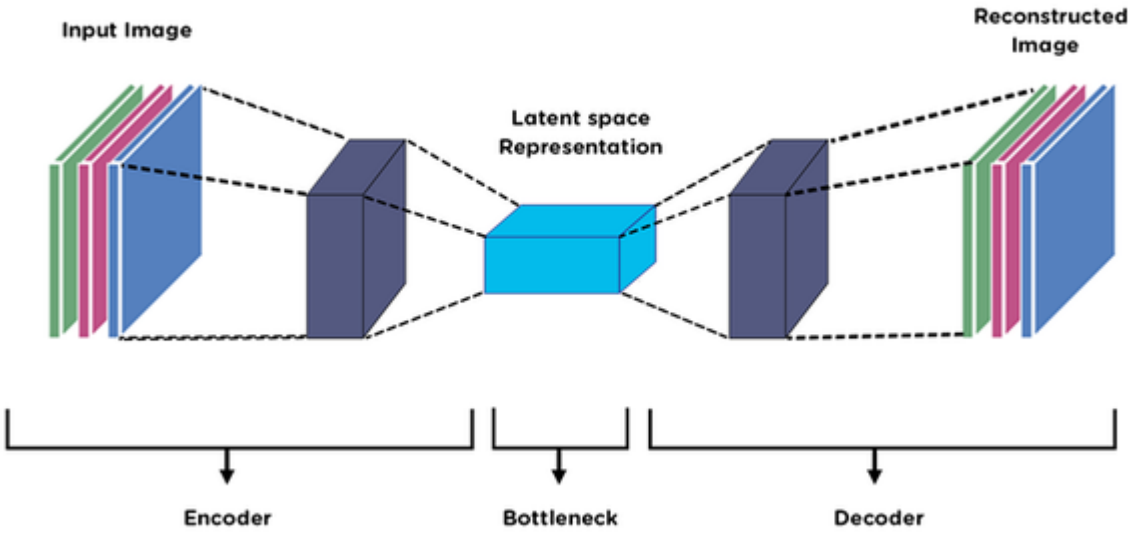
1 Introduzione

Nel vasto panorama della computer vision, una delle sfide più preminenti e in costante domanda riguarda la capacità di ripristinare immagini in presenza di perdita parziale di informazione. L'arte dell'inpainting, originariamente concepita per il restauro di opere d'arte ben prima dell'era digitale, si è evoluta con l'avvento della digitalizzazione delle immagini, divenendo ora una rilevante area di interesse all'interno del campo della computer vision.

Una delle tecniche chiave adottate per affrontare questa sfida è quella dell'autoencoder, un modello di machine learning il cui addestramento è condotto in modalità non supervisionata. Questa caratteristica conferisce agli autoencoder un notevole vantaggio, soprattutto considerando che la vasta mole di immagini disponibili in rete spesso non è adeguatamente etichettata.

Nel prosieguo di questa relazione, esamineremo l'implementazione di un modello di inpainting basato sugli autoencoder, discuteremo delle metodologie di addestramento e valutazione, e presenteremo i risultati ottenuti attraverso l'applicazione su due diversi dataset di immagini.

2 Cosa sono gli autoencoder



Gli autoencoder sono una categoria di modelli di deep learning utilizzati nell'ambito dell'apprendimento automatico e della riduzione delle dimensioni dei dati. Sono particolarmente noti per la loro capacità di apprendere rappresentazioni compatte ed efficienti dei dati di input, riducendo la loro dimensionalità. Gli autoencoder sono ampiamente utilizzati in diverse applicazioni di computer vision, tra cui l'inpainting.

La struttura di un autoencoder è costituita da due componenti principali:

- **Encoder**: Questa parte del modello accetta l'input e lo mappa in uno spazio di rappresentazione più compatto, noto come spazio latente. L'encoder è composto da una serie di strati neurali che riducono gradualmente la dimensionalità dei dati di input.

- **Decoder**: Questa parte del modello prende la rappresentazione compatta nello spazio latente generata dall'encoder e cerca di ricostruire l'input originale da questa rappresentazione. La struttura del decoder è spesso simmetrica rispetto a quella dell'encoder. Ciò significa che il numero di strati e il numero di neuroni in ciascuno strato sono organizzati in modo simile, ma in ordine inverso.

3 Analisi dei dati

Sono state create due versioni del modello, allenate su due differenti dataset di immagini:

- **FGVC-Aircraft** (Fine-Grained Visual Classification of Aircraft), che comprende 10.200 immagini a colori di velivoli di diverse categorie, principalmente aeroplani.
- **CelebA**, che comprende oltre 200mila immagini a colori di celebrità.

Mentre il primo dataset si è rivelato poco difficile da modellare, il secondo è quello che ci ha dato molto più filo da torcere. Per questo motivo, ci concentreremo in particolare sui risultati ottenuti da questo, che si sono rivelati essere più soddisfacenti.

3.1 Adattamento dei dataset

È stata scelta la dimensione 128x128 delle immagini per entrambi i dataset; il ridimensionamento è stato effettuato in maniera opportuna in maniera tale che i soggetti nelle immagini rimanessero ben visibili e inquadrati correttamente.

È stata inoltre introdotta una nuova colonna, che include, per ogni immagine, una copia dove è stata rimossa una porzione rettangolare, la cui posizione e dimensione, scelte in maniera casuale entro certi limiti, vengono incrementate per ogni epoca di training.

Nell'accesso al dataset, le immagini restituite vengono capovolte orizzontalmente il 50% delle volte.

4 Progettazione del modello

L'architettura del codice presentato è un esempio di un modello di autoencoder convoluzionale. La struttura dell'encoder e del decoder è simmetrica. Inoltre, alcune tecniche di regolarizzazione sono presenti all'interno dell'architettura.

4.1 Architettura dell'encoder

Sono stati sviluppati diversi modelli. Il modello di base (baseline) presenta i seguenti layer:

- **conv1**: Un layer convoluzionale con 64 filtri, una dimensione di kernel di 7x7, uno stride di 2 e padding di 3. Lo stride pari a 2 permette di ridurre la dimensionalità dell'immagine, mentre il padding di 3 fa in modo che la riduzione sia esattamente a metà.
- **conv2**: Un secondo layer convoluzionale con 128 filtri, una dimensione di kernel di 5x5, uno stride di 2 e padding di 2.
- **conv3**: Un terzo layer convoluzionale con 256 filtri, una dimensione di kernel di 5x5, uno stride di 2 e padding di 2.
- **conv4**: Un quarto layer convoluzionale con 512 filtri, una dimensione di kernel di 3x3, uno stride di 2 e padding di 1.
- **linear1**, **linear2**, **linear3**: Tre layer fully connected che riducono gradualmente la dimensione dell'output fino a raggiungere la dimensione del vettore latente (LATENT_DIM).

4.2 Architettura del decoder

Il decoder riceve il vettore latente prodotto dall'encoder e lo utilizza per generare una ricostruzione dell'immagine originale. Di seguito i principali componenti del decoder:

- **linear1**, **linear2**, **linear3**: Tre layer fully connected che aumentano gradualmente la dimensione dell'input fino a raggiungere la dimensione richiesta per i layer convoluzionali successivi.
- **conv1**, **conv2**, **conv3**, **conv4**: Quattro layer convoluzionali trasposti che aumentano progressivamente la dimensione dell'immagine fino a raggiungere le dimensioni dell'immagine originale.
- **residual_conv1** e **residual_conv2**: Due layer convoluzionali aggiuntivi utilizzati per aggiungere un blocco residuale alla ricostruzione. Questo blocco aiuta a migliorare la qualità dell'immagine ricostruita.

La normalizzazione **BatchNorm2d**, l'attivazione **SiLU** e il **dropout** sono applicati per una migliore generalizzazione. Tutti i layer convolutivi sono stati privati del bias, in quanto l'operazione di Batch normalizzazione imparerebbe semplicemente a rimuoverlo.

5 Addestramento

Per quanto riguarda l'addestramento, la funzione di loss utilizzata è la **MSE** per tutti i modelli (è stato effettuato un allenamento anche utilizzando la funzione Huber loss, ma con risultati scadenti, comparabili a quelli ottenuti con un primissimo modello fully connected).

Per quanto riguarda l'ottimizzazione del modello durante il training, è stato scelto l'ottimizzatore **AdamW** con un learning rate di 1e-4 e un peso di decay (weight decay) di 1e-2, insieme allo scheduler **Cosine Annealing Warm Restarts**, per garantire una convergenza più veloce.

5.1 Risultati dell'addestramento

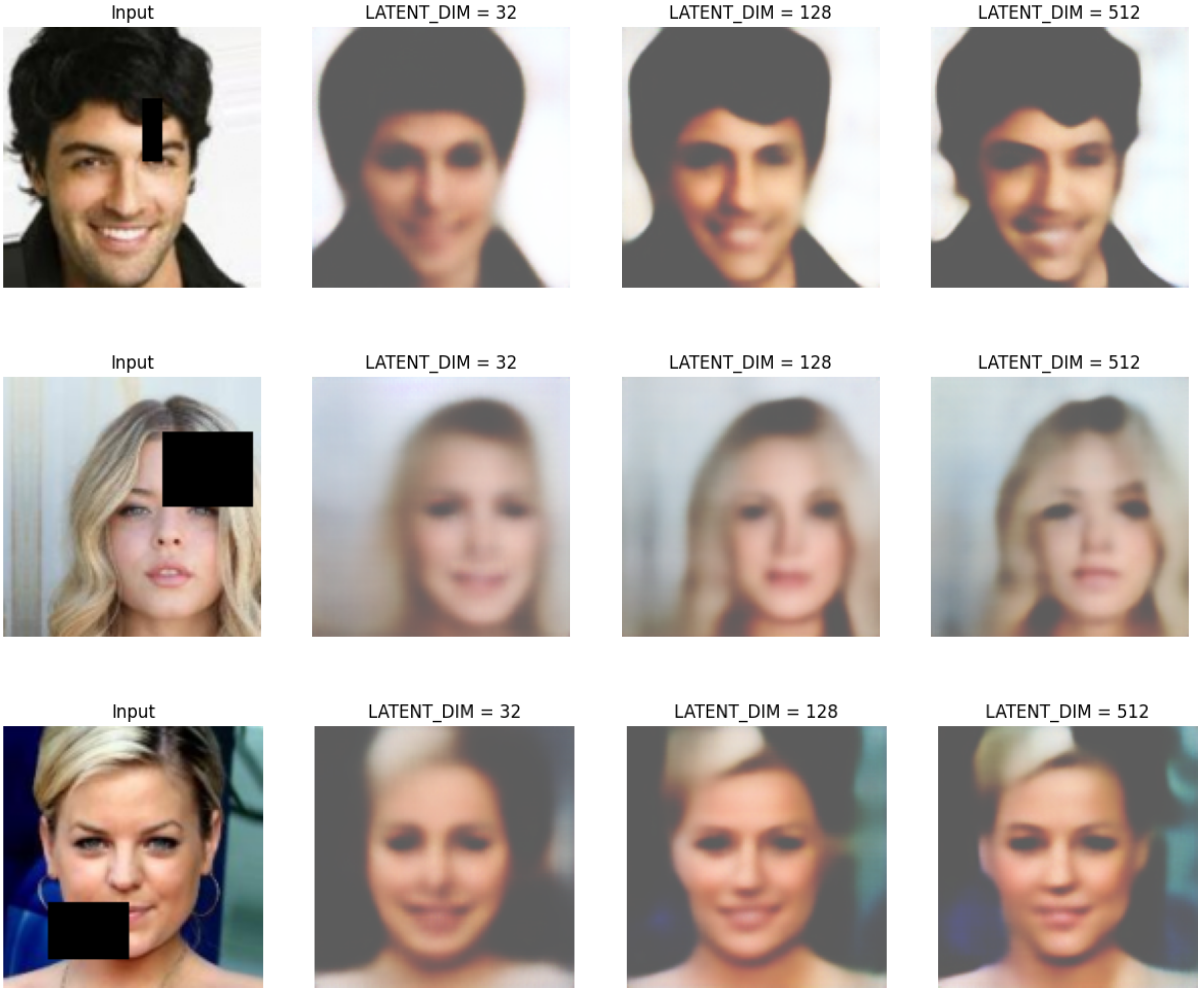
I vari training sono stati effettuati con diverse dimensioni per il vettore latente: **32**, **128**, **512**.

Una breve analisi visiva del risultato ottenuto sul primo dataset, rivela che il modello produce buoni output, con una qualità visiva nettamente superiore a quella di CelebA.

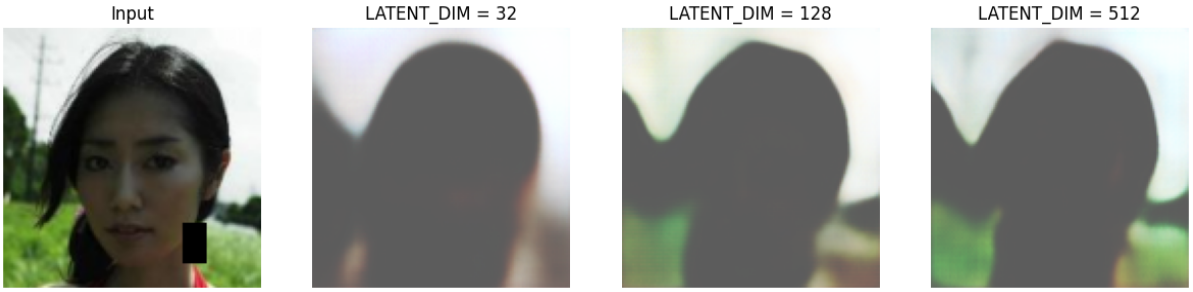
La tabella di seguito mostra i valori di loss ottenuti per le 3 versioni del modello sul dataset CelebA.

vettore latente	epoche	training loss	validation loss
32	8	1.77e+5	1.98e+5
128	10	1.71e+5	1.81e+5
512	8	1.67e+5	1.72e+5

All’analisi visiva il divario tra i vari vettori latenti è notevole, di seguito alcuni dei risultati più promettenti.



Dai risultati si è poi osservato che la rete sembra avere difficoltà con immagini che presentano toni scuri, come immagini ad alto contrasto, in condizione di bassa luce, e ampiamente nel caso di persone di pelle scura. Questo problema è noto nell'ambito della computer vision e delle reti convolutive, ed è dovuto, tra le altre cose, alla ridotta gamma dei valori di luminanza in immagini più scure.



Notare come l'aumentare della dimensione latente non ha migliorato la qualità del volto (in controluce), ma ha migliorato quella dello sfondo (ben illuminata).

Notare come, nonostante la buona illuminazione, le features facciali sono molto meno definite a tutte le dimensioni latenti. Ancora una volta, lo sfondo subisce un miglioramento in via dei toni a luminanza maggiore.

6 Conclusioni

I risultati ottenuti, per quanto ancora lontani dall'essere visivamente soddisfacenti, mostrano comunque che la rete si sta muovendo nella giusta direzione.
Bisogna comunque tenere conto che la rete utilizzata è relativamente piccola, e non fa uso di molte tecniche dell'arsenale degli autoencoder, in particolar modo non fa uso di nessuna tecnica di disentanglement del vettore latente o dropout nei layer lineari.

È altrettanto importante sottolineare che tutte le versioni non hanno raggiunto un plateau, quindi c'è ancora spazio per miglioramenti proseguendo l'allenamento per un numero maggiore di epoche.