UNIVERSITÀ POLITECNICA DELLE MARCHE

FACOLTÀ DI INGEGNERIA

Corso di Laurea triennale in *Ingegneria Informatica e dell'Automazione*

Sviluppo di un modello GAN per la generazione di immagini e relativa segmentazione

Develop of a GAN model for image generation with related segmentations

Relatore:
Prof. Adriano Mancini

Laureando:
Massimiliano Biancucci

Anno Accademico 2022/2023

Prefazione

Il mio percorso nel campo dell'inteligenza artificiale è iniziato diversi anni fà, alle superiori per l'esattezza, dove sentii per la prima volta parlare di reti neurali, ad un corso pomeridiano voluto dal prof. Roberto Lulli il quale mi ha mostrato per primo questo affascinante campo di ricerca.

Ho svolto durante il mio percorso di studi diversi progetti incentrati su questa tematica, partendo da semplici reti neurali, e confrontandomi con progetti sempre più complessi fino ad arrivare ai modelli generativi basati sull'architettura GAN (Generative Adversarial Network), del quale in questa tesi proporò una variante.

Lo scopo di questa tesi è quello di investigare la fattibilità di una potenziale soluzione ad uno dei grandi problemi che affligge oggi le aziende che si occupano di addestrare modelli neurali per la segmentazione di immagini, ovvero la difficoltà nel reperire immagini annotate, le quali hanno elevatissimi costi di realizzazione.

Tale scelta è stata naturale, in quanto ho dovuto confrontarmi in prima persona con questo problema nell'ultimo anno, come sviluppatore presso l'azienda Cloe.ai. In questa esperienza ho gestito per quasi un anno la realizzazione di un complesso dataset per l'addestramento di un modello di segmentazione di difetti, tale dataset aveva dei requisiti molto alti, e al contempo le risorse per realizzarlo erano limitate, per tale ragione ho potuto comprendere affondo le problematiche legate a questo tipo di progetto, facendo i conti io stesso con le spese e i progressi ottenuti.

La realizzazione di un dataset su larga scala è un'operazione molto complessa, che richiede una elevata coordinazione tra annotatori, revisori, sviluppatori, e un'accurata documentazione che in base al problema può richiedere anche diversi mesi per poter essere redatta efficaciemente. Tutto cio mi ha fornito la motivazione per cercare una soluzione per accorciare questo lungo e tedioso processo e dunque attenuare gli ingenti costi che un'azienda deve sostenere per realizzare un dataset di questo tipo.

Indice

Capitolo 1

Introduzione

1.1 Motivaione

1.1.1 Modelli neurali per la segmentazione nel contorllo qualità

Oggi le reti neurali trovano un vasto impiego in moltissi campi, dall'industria alla medicina, fino alla vita di tutti i giorni. Il grande vantaggio che ci portano è la capacità di apprendere da un set di dati, e di generalizzare su di uno nuovo, permettendocci di risolvere problemi che altrimenti sarebbero matematicamente troppo complessi da risolvere con un algoritmo. Ci sono vari esempi in cui i modelli neurali raggiungono risultati superiori a quelli ottenuti dall'uomo, in determinati task, o almeno se non lo superano in termini di accuratezza, lo fanno in termini di velocità, scalabilità, costi e prestazioni.

Un task in cui le reti neurali eccellono è la segmentazione di immagini, ovvero la classificazione pixel per pixel di un'immagine, questo tipo di task è utilizzato ad esempio nel campo medico per la segmentazione di organi, tumori, o in campo industriale per la segmentazione di difetti, per la verifica automatica della qualità di un prodotto o di un semilavorato.

Nel caso specifico, per la segmentazione dei difetti l'utilizzo di questo tipo di modelli è molto diffuso, in quanto risolve un grave problema che affligge i reparti controllo qualità delle aziende, ovvero il calo della concentrazione al quale un operatore è soggetto dopo un certo numero di ore di lavoro. Infatti una persona per quanto allenata e preparata, dopo un certo numero di ore di lavoro, è soggetta a stanchezza e con essa la sua accuratezza nel riconoscere un difetto diminuisce, mentre un modello neurale adeguatamente addestrato, in condizioni ambientli stabili, come ad esempio una adeguata illuminazione, una videocamera ad alta risoluzione e un'adeguata distanza dal soggetto, sarà in grado di mantenere un'accuratezza costante, senza necessità di fermarsi per riposare. Questo si traduce in un risparmio di tempo e di denaro per l'azienda, In quanto il controllo manuale richiede più tempo ed è più soggetto ad errori, i quali spesso si trasformano in ritardi nella consegna dei prodotti, spese di trasposrto aggiuntive per il ritorno o la sostituzione del prodotto, o addirittura la perdita di un cliente.

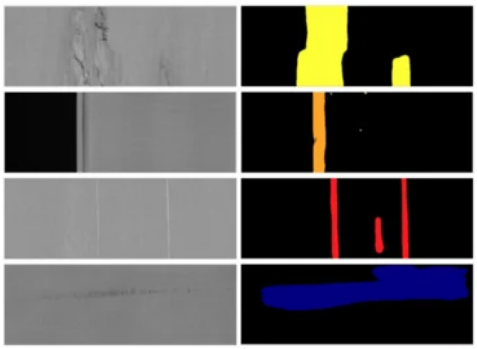

Figura 1.1: An example of segmented defects from the "Severstal steel defect dataset". credits: Neven Robby and Goedemé Toon, 2021, A Multi-Branch U-Net for Steel Surface Defect Type and Severity Segmentation. https://www.mdpi.com/2075-4701/11/6/870

1.1.2 Il dataset, requisiti e problematiche di realizzazione

La problematica di avere un modello con elevata accuratezza per task di segmentazione è relativa alla quantità e qualità dei dati necessari, i quali raramente sono disponibili opensource o per l'acquisto, rendendo necessaria la creazione di un dataset apposito. Molti task richiedono un grande quantità di dati per essere generalizzati correttamente, e ogni signolo esempio richiede molta concentrazione da parte dell'annotatore in quanto non sempre i difetti sono ben visibili.

In generale la creazione di un dataset è un'operazione molto complessa, non solo dal punto di vista dell'annotatore, ma in oltre da un punto di vista organizzativo e logistico. Infatti ci sono diversi step che si devono seguire:

- **Acquisizione delle immagini**: Le immagini devono essere acquisite in modo da avere una buona qualità, o almeno sufficiente ai fini dell'apprendimento. Se possibile in oltre dovrebbero avere una adeguata uniformità di condizioni (luce, distanza, ...) per garantire le migliori prestazioni da parte del modello, ovviamente solo se poi è possibile garantire le stesse condizioni anche nell'utilizzo finale del modello, altrimenti una grande varietà delle condizioni è preferibile.

- **Definizione delle classi**: Nel caso di dataset multiclasse, uno step molto importnate è quello di scegliere accuratamente le classi e definire in maniera univoca l'associazione tra una classe e una particolare tipologia di difetto. Questo passaggio potrebbe sembrare banale ma in realtà nasconde delle grandi insidie, infatti una classificazione non adeguata andrà a causare confusione nel modello, diminuendo la sua accuratezza e/o rendendo il lavoro più difficile per gli annotatori andando a rallentare il processo di annotazione o comunque a ridurne la qualità. Questo tipo di prolematiche purtroppo si manifestano chiaramente soltanto in uno stato avanzato del progetto, rendendo necessarie revisioni della documentazione, modifica di tutti gli esempi già annotati, con conseguente perdita di tempo e denaro.

- **Definizione della documentazione**: Questo passaggio è un'estensione del precedente, e consiste nella definizione di una documentazione Che specifichi senza ambiguità, ad un nuovo annotatore come riconoscere senza dubbio un difetto e classificarlo nella giusta

classe. Questa fase spesso non termina prima dell'inizio dell'annotazione, ma si protrae per tutta la durata del progetto, in quanto spesso nuovi casi non previsti si presentano durante l'annotazione, e la documentazione deve essere aggiornata in tempo reale.

- **Annotazione**: Questo è il passaggio più lungo e costoso, in quanto richiede una squadra di persone, che devono essere formate per lo specifico task, e che devono essere costantemente seguite per garantire la qualità del lavoro.
- **Revisione**: Assieme all'annotazione questo è un passaggio chiave, in quanto permette di verificare che l'annotazione sia stata fatta correttamente, e che non ci siano errori nell'annotazione. Spesso infatti gli annotatori aqcuisicono dei bias errati nei confronti di una certa classe, o di un certo tipo di difetto, che deve essere identificato e reso noto all'annotatore per correggerlo, ed evitare che questo errore si ripeta in futuro. Per evitare che ciò accada oltre al primo annotatore lo stesso esempio viene solitamente rivisto da 2 o 4 persone diverse. Si noti che gli errori degli annotatori che non vengono identificati verranno appresi dal modello finale come una corretta classificazione, ciò giustifica un tale dispendio di risorse in questa fase.

1.1.3 Approccio al problema

La creazione di un dataset come precedentemente illustrato è un processo complesso e dispendioso, che richiede molte risorse umane e finanziarie, dunque l'intento in questo progetto è quello di proporre un approccio alternativo che sia in grado di ridurre per quanto possibile la durata e il costo di questo lavoro. Partendo dal presupposto che almeno in parte il dataset deve essere realizzato manualmente, la proposta è quella di realizzare una certa quantità di campioni manualmente seguendo lo schema già visto, per poi addestrare un modello neurale per generare ulteriori esempi sintetici, raggiungendo un numero di esempi totali che permetta di addestrare un modello con buone prestazioni, ad un costo ridotto rispetto al caso in cui tutti i dati fossero stati realizzati manualmente.

Per la definizione della pipeline di generazione dei dati, si è partiti dal concetto di *generative adversarial network* (GAN), che è un approccio di machine learning che permette di generare dati sintetici utilizzando come base dati reali, tali dati sintetici possono essere utilizzati per addestrare un modello neurale. Tale tecnica ha trovato riscontri positivi in molte ricerche pubblicate in ambito di computer vision [2], in cui i modelli gan vengono utilizzati per espandere il numero di immagini presenti in un dataset e migliorare la generalizzazione di un modello di classificazione. Ovviamente gli aumenti di accuratezza, precisione e recall dipendono dal numero di esempi presenti nel dataset e dalla complessità del problema. Tale tecnica potrebbe essere considerata una versione più sofisticata di data augmentation, in quanto permette di generare dati sintetici molto più complessi e realistici di quelli che si possono ottenere con semplici trasformazioni geometriche o matematiche. Per generare dati utilizzabili per addestrare un modello di segmentazione però è necessario risolvere un'ulteriore problema, infatti un normale modello GAN, fedele alla sua definizione originale [5], è in grado di generare intere immagini, che possono essere utilizzate per addestrare un modello di classificazione, ma non sono utilizzabili per addestrare un modello di segmentazione, in quanto per l'addestramento di tale architettura

è necessario che gli oggetti di interesse abbiano una maschera che specifichi la loro posizione. Per risolvere questo problema ci sono 2 principali strade illustrate di seguito.

Generatore con architettura a solo decoder

Questo approccio prvede un'architettura a solo decoder, dunque con in ingresso, un vettore casuale di dimensione arbitraria, e in uscita un tensore che può contenere i canali rgb di un'imagine e altri n canali per la maschere che identificano le classi desiderate. Un'esempio di tale architettura a solo decoder potrebbe essere quella di DCGAN [9], illustrata qui sotto.

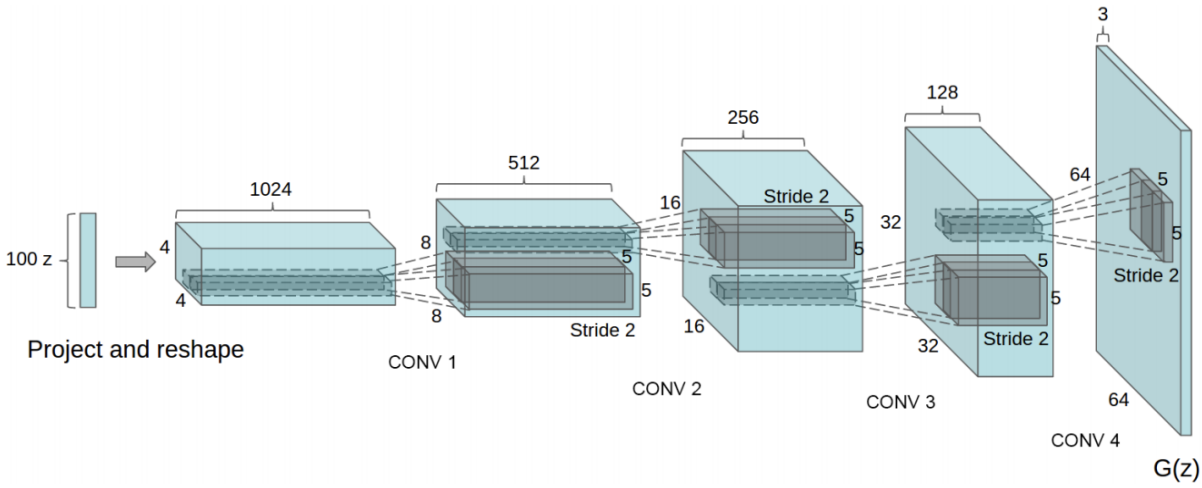

Figura 1.2: Architettura di DCGAN, un esempio di GAN convoluzionale, con architettura a solo decoder. credits: Imagine presa dall'articolo di Alec Radford, Luke Metz, Soumith Chintala, Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks

Questo approccio risulta più semplice da implementare, lasciando però al modello il compito di imparare a generare correttamente le immagini e delle maschere coerenti, compito non facile che probabilmente necessita di un elevato numero di esempi. Questa architettura dovrà imparare oltre alla struttura degli oggetti target, a posizionarli nell'immagine. Un'altra probblematica di questo approccio è il controllo, infatti l'unico modo di interagire con tale modello è modificando il valore del vettore z, il quale permette di spostarsi nello spazio latente, al quale il modello associa diverse caratteristiche dell'immagine di output in maniera altamente non lineare, rendendo un eventuale controllo dell'output del modelo molto difficile. La difficoltà di controllare il modello rende dunque difficoltoso o impossibile controllare, qualora fosse necessario, la posizione, l'intensità, la dimensione o la forma degli oggetti generati.

Generatore con architettura a encoder-decoder

OLD

Per quanto riguarda la segmentazione, l'approccio standard è ..

L'obbiettivo di questo progetto è proporre una soluzione a questo problema, o almeno mitigare la sua gravità. Per far ciò si propone di realizzare un architettura in grado di generare un oggetto in un'immagine presistente, specificando la maschera di quest'ultimo.

Tale architettura deve essere in grado di prendere in input un'immagine e una maschera, e generare l'oggetto all'interno della maschera, lasciando il più possibile invariata l'immagine al di fuori dell'area d'interesse. Il modello dovrebbe essere in grado di generare un oggetto all'interno dell'area obbiettivo seguendo la distribuzione morfologica degli oggetti presenti nel dataset di addestramento.

Tale tecnica è principalmente pensata per tutti quei casi in cui non è semplice o possibile realizzare sinteticamente un oggetto a partire da metodi deterministici, ad esempio per la segmentazione dei difetti, i quali non possono essere copiati e incollati in una nuova immagine senza che sia presente un evidente linea di separazione tra il difetto e il resto dell'immagine, e quindi non possono essere generati con una tecnica di tipo copy paste, senza introdurre artefatti, che potrebbero portare ad un peggioramento delle prestazioni del modello, in quanto questo associerebbe il difetto alla linea di separazione.

In questo progetto il dataset utilizzato è Severstal steel defect detection, il quale presenta campioni di acciaio più tosto simili, ma in generale questa tecnica potrebbe essere utilizzata anche per casi più complessi come il rilevmento di danni su veicoli o edifici, dove la morfologia del difetto è comune, ma la morfologia, il colore e la texture dell'oggetto su cui deve essere applicato sono differenti dal dataset di addestramento, in quest'ultimo scenario infatti non è possibile utilizzare una tecnica di tipo copy paste, in quanto il difetto non potrebbe essere applicato in modo coerente con l'immagine base.

Capitolo 2

Stato dell'arte

Capitolo 3

Strumenti e metodi

3.1 Il Dataset: Severstal steel defect detection

L'acciaio è uno dei materiali più comunemente utilizzati in tutto il mondo, e la sua produzione è in continua crescita. La sua versatilità e la sua resistenza lo rendono un materiale molto utilizzato in diversi settori, come l'edilizia, l'automotive, l'industria elettronica, l'industria aerospaziale, ecc. Per produzioni su larga scala di acciaio come di altri materiali o prodotti, è necessario che il materiale sia di qualità, e che non contenga difetti, ma è difficile per gli operatori umani rilevare difetti come graffi, crepe, ecc. con elevata affidabiltà, per tale ragione è necessario adottare sistemi automatizzati che siano in grado di rilevare difetti in modo affidabile e veloce. Severstal è una delle principali aziende produttrici di acciaio in Russia, e produce circa 10 milioni di tonnellate di acciaio all'anno, quest'ultima ha iniziato dunque ad utilizzare il machine learning per automatizzare il processo di rilevazione dei difetti, e ha messo a disposizione questo dataset nel 2019 per permettere a chiunque di partecipare alla challenge, e di testare le proprie idee, sperando di riuscire ad aumentare l'affidabiltà del proprio sistema di rilevazione difetti.

Questo dataset è diviso in 2 parti, training e test set, ma per il test set non sono stati rilasciati i ground truth, quindi non è possibile testare il modello finale sul test set originale e verranno dunque utilizzati solo i dati del training set, i quali verranno suddivisi in training e test set. Il training set originale è composto da 12568 immagini, che verrano divise in 50% per il nuovo training set e 50% per il nuovo test set, quindi un totale di 6284 immagini per il training set e 6284 immagini per il test set.

il training set verrà utilizzato per il training del generatore di difetti sintetici, mentre il test set verrà utilizzato per testare il modello di segmentazione finale addestrato con il dataset sintetico.

3.2 Lama:

3.3 Stylegan2:

Capitolo 4

Sviluppo del progetto

Capitolo 5

Risultati

Elenco delle figure

Bibliografia

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017.

[2] Maayan Frid-Adar, Idit Diamant, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification, 2018.

[3] Rinon Gal, Dana Cohen, Amit Bermano, and Daniel Cohen-Or. Swagan: A style-based wavelet-driven generative model, 2021.

[4] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D. Cubuk, Quoc V. Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation, 2021.

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[6] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium, 2018.

[7] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks, 2018.

[8] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge?, 2018.

[9] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016.

[10] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans, 2016.

[11] Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. Resolution-robust large mask inpainting with fourier convolutions, 2021.

[12] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric, 2018.