

Image-To-Image Translation Overview

Gaetano Di Grazia*

Giuseppe Naso*

Università degli studi di Palermo



Figure 1: Credit to Pixabay. A representation of a brain.

Abstract

La traduzione immagine-immagine è una classe di problemi di visione e di grafica in cui l’obiettivo è apprendere la mappatura tra un’immagine in ingresso e un’immagine in uscita utilizzando insiemi di allenamento formati da coppie di immagini; per essere più precisi, essi sono paragonati a vettori immagine di n elementi che possiamo indicare, genericamente, X e Y . Troviamo utile, per i meno esperti, fornire una carrellata di concetti che stanno alla base di questa tecnologia e, infine, una breve panoramica sullo stato dell’arte.

1. Introduction

Prima ancora di imbatterci nella disamina delle GAN, nonché dell’image-to-image translation, è utile fare qualche riferimento ai concetti che stanno alla base di tale tecnologia.

1.1. Intelligenza artificiale

Il primo concetto che possa venirci in mente è proprio quello della scienza, della disciplina, che studia e sviluppa questo genere di tecnologie; diamo dunque la seguente

Definition 1.1 *L’intelligenza artificiale è una disciplina appartenente all’informatica che studia i fondamenti teorici, le metodologie e le tecniche che consentono la progettazione di sistemi hardware e sistemi di programmi software capaci di fornire all’elaboratore elettronico prestazioni che, a un osservatore comune, sembrerebbero essere di pertinenza esclusiva dell’intelligenza umana.*

L’intelligenza artificiale fa abbondantemente uso delle reti neurali che vengono utilizzate per risolvere problemi ingegneristici legati a diversi ambiti tecnologici come l’informatica, l’elettronica o altre discipline; più precisamente

Definition 1.2 *Una rete neurale (in inglese neural network) è un modello matematico composto da neuroni artificiali di ispirazione alle reti neurali biologiche (quella*

umana o animale).

Le reti neurali fanno sì che i computer siano in grado di risolvere i problemi in modo indipendente e che migliorino le loro capacità.

A sua volta, facenti parte della branca dell'intelligenza artificiale troviamo i concetti di machine learning e deep learning.

Il Machine Learning è la tecnologia storicamente più antica e più semplice; questa utilizza un algoritmo che il sistema adatta, solo dopo aver ricevuto un feedback umano e presuppone l'esistenza di dati strutturati disponibili all'elaborazione.

In tal caso il sistema viene prima alimentato con dati strutturati e categorizzati e quindi "cenisce" come classificare i nuovi dati a seconda del tipo e poi, in base alla classificazione, il sistema esegue le attività programmate.

Ad esempio è in grado di riconoscere un cane o un gatto in una foto e di spostare i file nelle cartelle corrispondenti.

Dopo una fase iniziale di applicazione, l'algoritmo è ottimizzato dal feedback umano, che indica al sistema le classificazioni errate e le categorizzazioni corrette.

Nel caso del Deep Learning, invece, i dati strutturati non sono necessari, infatti il sistema funziona nelle reti neurali multistrato, che combinano diversi algoritmi e sono modellate sul cervello umano; ciò consente al sistema di elaborare anche dati non strutturati.

Ciò che è importante da notare è che nel Deep Learning, è il sistema stesso ad identificare nei dati le caratteristiche distinctive adeguate, senza la necessità di una categorizzazione dall'esterno; l'addestramento da parte di uno sviluppatore non è necessario, è il sistema stesso a controllare se le classificazioni cambiano a causa di un nuovo input o se ne vanno introdotte di nuove.

2. Un pizzico di matematica

2.1. Norma

La norma di un vettore $\vec{u} = (u_1, u_2, \dots, u_n) \in \mathbf{R}^n$ è un'applicazione che ad un vettore associa un numero reale

$$\|\cdot\| : \mathbf{R}^n \rightarrow \mathbf{R} \mid f : \mathbf{R}^n \rightarrow \mathbf{R}^+ \cup \{0\}$$

così definito:

$$\|\vec{u}\| = \sqrt{u_1^2 + u_2^2 + \dots + u_n^2}$$

altro non è che la radice quadrata della somma dei quadrati delle componenti del vettore.

2.2. Probabilità condizionata. Formula di Bayes. Teorema delle probabilità composte

Siano ΔN_{E_1} e $\Delta N_{E_1 \cap E_2}$ il numero di volte che, su un totale di N ripetizioni dell'esperimento casuale, si manifes-

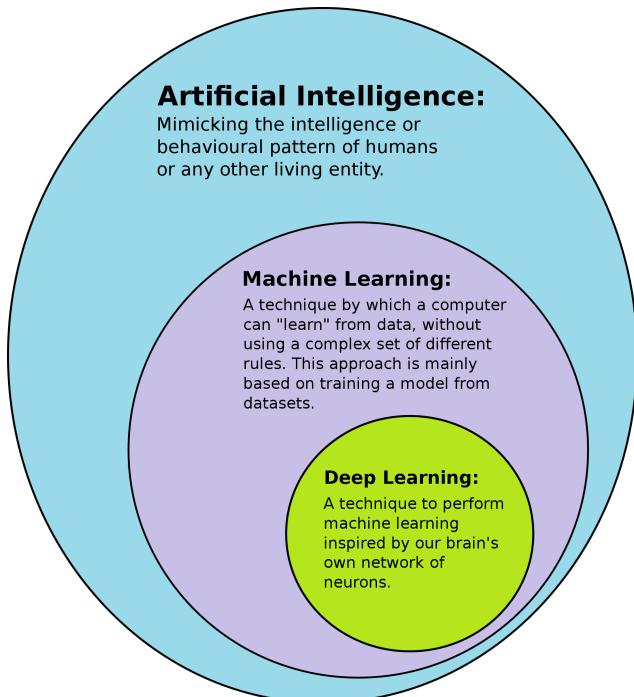


Figure 2: Hierarchy of AI.

Intelligenza artificiale: qualsiasi tecnica che permette ai computer di imitare il comportamento umano; Machine learning: capacità di apprendere senza essere esplicitamente programmato; Deep learning: estrarre modelli dai dati utilizzando una rete neurale.

tano gli eventi E_1 e $E_1 \cap E_2$, allora dalla relazione

$$\frac{\Delta N_{E_1 \cap E_2}}{N} = \frac{\Delta N_{E_1 \cap E_2}}{\Delta N_{E_1}} \cdot \frac{\Delta N_{E_1}}{N}$$

si ottiene, passando al limite per $N \rightarrow \infty$

$$Pr\{E_1 \cap E_2\} = Pr\{E_2|E_1\} Pr\{E_1\}$$

dove $Pr\{E_2|E_1\}$ è:

$$Pr\{E_2|E_1\} = \lim_{N \rightarrow \infty} \frac{\Delta N_{E_1 \cap E_2}}{\Delta N_{E_1}}$$

Osservando che $\frac{\Delta N_{E_1 \cap E_2}}{\Delta N_{E_1}}$ rappresenta il rapporto fra il numero di volte in cui, in un totale di N ripetizioni dell'esperimento casuale, si verifica l'evento E_1 , la $Pr\{E_2|E_1\}$ può essere interpretata come la probabilità che si verifichi l'evento E_2 sotto l'ipotesi che E_1 sia soddisfatto.

Definition 2.1 (Probabilità condizionata) Dicesi probabilità dell'evento E_2 condizionata da E_1 la quantità

$$Pr\{E_1|E_2\}$$

In modo analogo può scriversi:

$$Pr\{E_1 \cap E_2\} = Pr\{E_1|E_2\} Pr\{E_2\}$$

dove $Pr\{E_1|E_2\}$ denota la probabilità che si manifesti E_1 atteso che E_2 sia verificato.

2.3. Formula di Bayes

Dal confronto fra le $Pr\{E_1 \cap E_2\} = Pr\{E_2|E_1\}Pr\{E_1\}$ e $Pr\{E_2|E_1\} = \lim_{N \rightarrow \infty} \frac{\Delta N_{E_1 \cap E_2}}{\Delta N_{E_1}}$ si ottiene la

Formula 2.2 (di Bayes)

$$Pr\{E_1|E_2\} = Pr\{E_2|E_1\} \frac{Pr\{E_1\}}{Pr\{E_2\}}$$

Essa stabilisce la relazione tra le probabilità condizionate e quelle non condizionate degli eventi E_1 e E_2 .

Definition 2.3 (Statisticamente indipendenti) Due variabili E_1 ed E_2 si dicono statisticamente indipendenti se risulta:

$$Pr\{E_1|E_2\} = Pr\{E_1\} \text{ o } Pr\{E_1|E_2\} = Pr\{E_2\}$$

cioè la probabilità con cui si manifesta E_1 (E_2) è indipendente dalla circostanza che E_2 (E_1) sia verificato.

Dunque possiamo scrivere

$$Pr\{E_1 \cap E_2\} = Pr\{E_1\} \cdot Pr\{E_2\}$$

A parole, la probabilità dell'intersezione di due eventi statisticamente indipendenti si riduce semplicemente al prodotto delle probabilità associate ai singoli eventi.

2.4. Correlazione

La correlazione è una relazione tra due variabili tale che a ciascun valore della prima corrisponda un valore della seconda a patto che si segua una certa regolarità; è comune, nella statistica, calcolare gli *indici di correlazione* che misurano la *forza* di questa relazione, di seguito una delle formule più usate, (ne esistono anche altre come r_s di Spearman o il τ di Kendall), detta coefficiente di correlazione lineare di Pearson

$$r = \frac{\sum(x_t - \hat{x})(y_t - \hat{y})}{\sqrt{\sum(x_t - \hat{x})^2(y_t - \hat{y})^2}}$$

In particolare si dice lineare la relazione il cui grafico su assi cartesiani si avvicina alla forma di una retta, non lineare quando ha un andamento curvilineo.

È importante notare che intanto $r \in [-1, 1]$ e poi che tale coefficiente misura la forza della relazione **lineare** tra le due variabili ma non quella non lineare; infatti, ad esempio, potrebbe accadere di poter misurare una correlazione lineare, magari anche positiva, tra due variabili ma che se ne misuri una ancora più forte ma *non lineare*.

2.5. Autocorrelazione

Proprio come la correlazione misura l'entità di una relazione lineare tra due variabili, l'autocorrelazione misura la relazione lineare tra i valori ritardati di una serie temporale.

In generale, possiamo dunque misurare l'indice di autocorrelazione mediante la formula

$$r_k = \frac{\sum_{t=k+1}^T (y_t - \hat{y})(y_{t-k} - \hat{y})}{\sum_{t=1}^T (y_t - \hat{y})^2}$$

dove T è la lunghezza della serie storica considerata.

Dunque esaminando questo risultato possiamo dire che:

- quando i dati presentano un trend, allora le autocorrelazioni per piccoli ritardi tendono ad essere ampie e positive. Questo è chiaro in quanto, salvo casi eccezionali, osservazioni vicine nel tempo hanno anche dimensioni pressoché coerenti;
- quando i dati sono stagionali allora le autocorrelazioni saranno maggiori per i ritardi stagionali e si presenteranno a multipli della frequenza stagionale;
- quando, invece, i dati mostrano sia un trend che una stagionalità allora è possibile osservare una combinazione dei due effetti sopra descritti.

3. Computer vision

3.1. Pattern recognition

Il campo del pattern recognition riguarda la scoperta automatica di regolarità nei dati attraverso l'uso di algoritmi informatici e l'uso di queste regolarità per intraprendere azioni come la classificazione dei dati in diverse categorie.

Chiaramente i dati devono essere preprocessati e convertiti in una forma che un computer può capire per fare questo possiamo usare tre diversi metodi: classificazione, regressione, o algoritmi di clustering a seconda delle informazioni disponibili sul problema per ottenere risultati validi, in particolare

- in **classification**, l'algoritmo assegna etichette ai dati in base alle caratteristiche predefinite e questo è un esempio di apprendimento supervisionato;
- **clustering** l'algoritmo divide i dati in un certo numero di cluster basati sulla somiglianza delle caratteristiche, in questo caso viene adottato l'apprendimento non supervisionato;
- **regression** gli algoritmi cercano di trovare una relazione tra le variabili e prevedere le variabili dipendenti sconosciute sulla base di dati noti, anche questo algoritmo è basato sull'apprendimento supervisionato.

3.2. Feature

Nell'apprendimento automatico e nel riconoscimento dei modelli, una caratteristica è una proprietà individuale misurabile o una caratteristica di un fenomeno; scegliere caratteristiche informative, discriminanti e indipendenti è un elemento cruciale per algoritmi efficaci nel riconoscimento dei modelli. Le caratteristiche sono di solito numeriche, ma caratteristiche strutturali come stringhe e grafici sono usate nel riconoscimento sintattico dei modelli. In sintesi le caratteristiche sono variabili indipendenti individuali che agiscono come input nel vostro sistema e i modelli di predizione usano le caratteristiche per fare previsioni, a volte queste sono anche chiamate attributi, e il numero di caratteristiche è chiamato dimensioni.

3.3. Classification

La classificazione è un processo di categorizzazione di un dato insieme di dati in classi attraverso l'apprendimento supervisionato, può essere eseguito sia su dati strutturati che non strutturati. Il processo inizia con la previsione della classe di punti dati dati e le classi sono spesso indicate come target, etichette o categorie. La modellazione predittiva di classificazione è il compito di approssimare la funzione di mappatura da variabili di input a variabili di output discrete. L'obiettivo principale è quello di identificare in quale classe/categoria cadranno i nuovi dati. Le etichette di classe sono spesso valori stringa, ad esempio "spam", "non spam", e devono essere mappate in valori numerici prima di essere fornite a un algoritmo per la modellazione. Questo è spesso indicato come codifica dell'etichetta, dove un unico numero intero è assegnato ad ogni etichetta di classe, ad esempio "spam" = 0, "no spam" = 1.

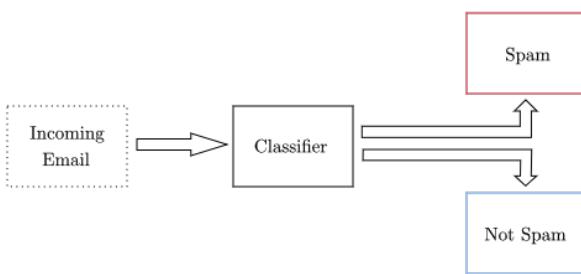


Figure 3: An example of classification of an email.

Possiamo usare diversi modi di classificazione come: Regressione logistica, Naive Bayes, Stochastic Gradient Descent, K-Nearest Neighbors, Decision Tree, Random Forest, Artificial Neural Network, Support Vector Machine.

Alla fine, le etichette, o classi, sono l'output finale, infatti quando gli scienziati dei dati parlano di dati etichettati, intendono gruppi di campioni che sono stati etichettati con una o più etichette.

In breve, la caratteristica è l'input; l'etichetta è l'output.

Questo si applica sia ai problemi di classificazione che di regressione. Una caratteristica è una colonna dei dati nel nostro set di input. Per esempio, se stiamo cercando di predire il tipo di animale domestico che qualcuno sceglierà, le nostre caratteristiche di input potrebbero includere l'età, la regione di provenienza, il reddito familiare, ecc. Pertanto, l'etichetta sarà la scelta finale come cane, pesce, iguana, roccia, ecc.

3.4. Image segmentation

Ci sono un paio di compiti che la computer vision si ripromette di risolvere, in particolare

- image classification, la classificazione delle immagini vecchio stile perché interessati alla sola classe degli elementi in esse contenuti;
- object detection, l'identificazione degli oggetti in un'immagine con l'aggiunta di rettangoli di selezione (bounding box) che ne individuano la posizione
- object segmentation, la distinzione degli oggetti dallo sfondo tramite l'identificazione dei loro bordi

Inoltre, la object segmentation si suddivide ulteriormente in Semantic Segmentation e Instance Segmentation.

In particolare, quella che andremo a definire, e che poi ritroviamo nelle GAN, il processo di Image segmentation.

Definition 3.1 *Nell'elaborazione digitale delle immagini e nella computer vision, la segmentazione delle immagini è il processo di suddivisione di un'immagine digitale in più segmenti (insiemi di pixel, noti anche come oggetti dell'immagine). L'obiettivo della segmentazione è di semplificare e/o cambiare la rappresentazione di un'immagine in qualcosa che sia più significativo e più facile da analizzare.*

Dunque nella image segmentation si avrà la classificazione di ogni pixel in una specifica classe, o label.

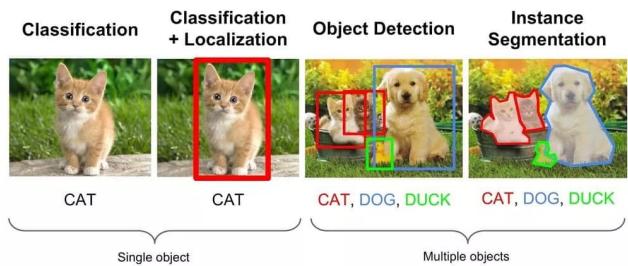


Figure 4: Sample of instance segmentation

All'interno dell'image segmentation possiamo trovare altri due ambiti, ovvero: la semantic segmentation e la instance segmentation, ne diamo dunque due definizioni poco formali

Definition 3.2 *La segmentazione semantica è un approccio che individua, per ogni pixel, la classe di appartenenza dell'oggetto.*

Definition 3.3 *La segmentazione dell'istanza è un approccio che identifica, per ogni pixel, un'istanza appartenente all'oggetto. Rileva ogni oggetto di interesse distinto nell'immagine.*

Riferendoci, dunque, ancora all'immagine sopra, ciò che è notiamo è che nel caso descritto come object detection abbiamo due gatti ed entrambi vengono descritti con la label "gatto", nel caso accanto a destra, invece, gli animali verranno etichettati come: gatto 1, gatto 2, cane e papera.

4. Deep learning

Il deep learning nasce come conseguenza di una necessità, infatti i metodi tradizionali hanno a che fare con problemi multivariati ad alta dimensione, relazioni non lineari e set di dati incompleti; queste sono situazioni che si verificano comunemente nei dati del mondo reale e sono molto più facilmente risolvibili utilizzando reti neurali profonde. Esistono anche degli svantaggi però, infatti le reti neurali sono progettate come scatole nere che possono essere di difficile interpretabilità statistica.

La modellazione sequence-to-sequence per le serie temporali è stata abbastanza popolare in passato e continua sempre di più ad esserlo oggi giorno; questi modelli si riferiscono specificamente a reti neurali in cui l'input è una sequenza (ad esempio, una serie temporale) e l'output è analogamente una sequenza.

Per consentire la manipolazione della serie, il modello deve:

- tenere traccia dello storico delle dipendenze della sequenza;
- ricordare l'ordinamento dell'indice temporale;
- consentire l'ottimizzazione dei parametri sulla durata della sequenza.

Esistono vari tipi di architetture di rete neurale in base ai diversi tasks da eseguire, le principali sono: Reti Neurali Convoluzionali (CNN) e Reti Neurali Ricorrenti (RNN).

4.1. CNN

Le CNN sono utilizzate principalmente nella visione artificiale, dove gli input sono spesso matrici con voci corrispondenti ai valori dei pixel.

Queste sono comunemente utilizzate grazie alla relativa semplicità nella progettazione delle funzionalità; infatti non necessitano, per esempio, di imparare da una serie di osservazioni ritardate nel tempo.

To this day, vision systems based on convolutional neural networks are among the best performing systems. Le CNN sono ottime per apprendere rappresentazioni di input di grandi dimensioni.

Convolutional nets were inspired by the visual system's structure, and in particular by the models of it proposed by Hubel and Wiesel (1962). The first computational models based on these local connectivities between neurons and on hierarchically organized transformations of the image are found in Fukushima's Neocognitron (Fukushima, 1980).

4.2. RNN

Una rete neurale ricorrente (RNN) è una classe di reti neurali artificiali in cui le connessioni tra i nodi formano un grafico diretto lungo una sequenza temporale. Questo gli permette di esibire un comportamento dinamico temporale. Derivate dalle reti neurali feedforward, le RNN possono utilizzare il loro stato interno (memoria) per elaborare sequenze di input di lunghezza variabile. Le reti neurali ricorrenti sono utilizzate principalmente nell'elaborazione del linguaggio naturale, l'esempio più classico è la traduzione.

Le RNN, visto che lavorano su sequenze di lunghezza arbitraria, superano le limitazioni imposte da altre strutture, quali le Convolutional Neural Network (CNN) che impongono input di lunghezza fissa. Inoltre, le RNN consentono ai layer di avere stati nascosti quando vengono inseriti in un livello successivo, in modo che i valori storici possano giocare un ruolo nella previsione. Condividono molti aspetti positivi con le CNN quando si tratta di sequenze, con il plus di applicare una ricorrenza in ogni fase temporale durante l'elaborazione delle sequenze.

4.3. Some others concepts

4.3.1 Parameter and Hyper-Parameter

I parametri sono variabili di configurazione che possono essere pensate come interne al modello in quanto possono essere stimate dai dati di addestramento e gli algoritmi possono ottimizzare i parametri. Mentre, un discorso più complesso avviene per gli iperparametri, infatti, questi non possono essere stimati a partire dai dati di addestramento; in altre parole gli iperparametri di un modello sono, in qualche modo, esterni al modello e sono impostati a seconda di una combinazione di alcune euristiche e dell'esperienza e conoscenza del dominio del data scientist.

4.3.2 Underfitting and Overfitting

Una considerazione importante nell'apprendimento automatico è quanto bene l'approssimazione della funzione obiettivo che è stata addestrata utilizzando i dati di allenamento.

mento, si generalizza ai nuovi dati. La generalizzazione funziona meglio se il segnale o il campione che viene usato come dati di allenamento ha un alto rapporto segnale/rumore, in base alla qualità della generalizzazione possiamo avere due tipi di situazioni: underfitting o overfitting. L'underfitting si riferisce a un modello che non può né modellare i dati di allenamento né generalizzare a nuovi dati.

Chiaramente, un modello di apprendimento automatico underfitting non è un modello adatto e sarà ovvio in quanto avrà: scarse prestazioni sui dati di allenamento e quindi non otterremo buone previsioni. Altrimenti un modello overfitting è un modello che si adatta troppo bene ai dati di allenamento e c'è una scarsa generalizzazione dei nuovi dati.

4.3.3 Weight and bias

I pesi e le distorsioni (comunemente chiamati w e b) sono i parametri apprendibili di un modello di apprendimento automatico. Il peso è il parametro all'interno di una rete neurale che trasforma i dati di input all'interno degli strati nascosti della rete. Una rete neurale è una serie di nodi, o neuroni. All'interno di ogni nodo c'è un insieme di input, un peso e un valore di bias. Quando un input entra nel nodo, viene moltiplicato per un valore di peso e l'output risultante viene osservato o passato allo strato successivo della rete neurale. Spesso i pesi di una rete neurale sono contenuti negli strati nascosti della rete.

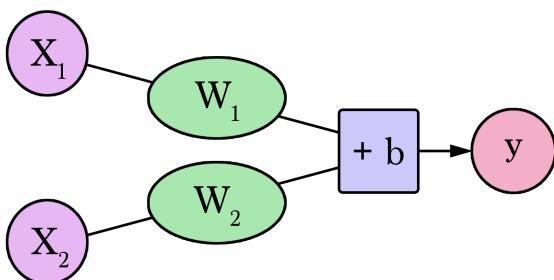


Figure 5: Source of image [Source](#)

Quindi, i pesi controllano il segnale (o la forza della connessione) tra due neuroni. In altre parole, un peso decide quanta influenza avrà l'input sull'output.

I bias, che sono costanti, sono un input aggiuntivo nello strato successivo che avrà sempre il valore di 1. Le unità bias non sono influenzate dallo strato precedente (non hanno connessioni in entrata) ma hanno connessioni in uscita con i loro pesi. L'unità bias garantisce che anche quando tutti gli ingressi sono zero ci sarà comunque un'attivazione nel neurone.

4.4. Training

Durante l'addestramento per l'apprendimento automatico, si passa un algoritmo con dati di addestramento. L'algoritmo di apprendimento trova dei modelli nei dati di addestramento in modo che i parametri di input corrispondano all'obiettivo. L'output del processo di formazione è un modello di apprendimento automatico che si può poi utilizzare per fare previsioni. Questo processo è anche chiamato "apprendimento".

4.4.1 Training, validation and tests

In genere, per eseguire l'apprendimento supervisionato, avete bisogno di due tipi di set di dati:

- In un set di dati, il nostro cosiddetto "gold standard", si hanno i dati di input insieme all'output corretto/atteso; questo set di dati è di solito debitamente preparato da esseri umani o raccogliendo alcuni dati in modo semi-automatico. Ma dovete avere l'output previsto per ogni riga di dati qui, perché ne avete bisogno per l'apprendimento supervisionato.
- I dati a cui applichiamo il nostro modello. In molti casi, questi sono i dati a cui siete interessati all'output del nostro modello, e quindi non avete ancora un output "atteso" qui.

Quindi, mentre si esegue l'apprendimento automatico, si fa quanto segue:

- Fase di addestramento: presentate i dati del vostro "gold standard" e addestrate il vostro modello, accoppiando l'input con l'output previsto.
- Fase di convalida/prova: per stimare quanto bene il vostro modello sia stato addestrato (che dipende dalla dimensione dei vostri dati, il valore che vorreste prevedere, l'input, ecc.) e per stimare le proprietà del modello (errore medio per i predittori numerici, errori di classificazione per i classificatori, recall e precisione per i modelli IR, ecc.)
- Fase di applicazione: ora, si applica il modello appena sviluppato ai dati del mondo reale e si ottengono i risultati. Dato che di solito non avete alcun valore di riferimento in questo tipo di dati, potete solo speculare sulla qualità dell'output del vostro modello usando i risultati della vostra fase di validazione.

Si noti che la fase di convalida è spesso divisa in due parti diverse:

- Nella prima parte, si guardano semplicemente i modelli e si seleziona l'approccio più performante usando i dati di validazione (=validation);

- Poi si stima la precisione dell'approccio selezionato (=test).

Spesso l'ultima fase, l'applicazione, non viene considerata come una vera e propria fase perché è immediatamente successiva alla fase di Test e quindi viene spesso considerata allo stesso modo, quindi vengono menzionate solo tre fasi: Addestramento, Convalida e Test.

5. GAN e CycleGAN

L'architettura proposta dai ricercatori dell'articolo scientifico oggetto di questo caso studio è la Cycle-GAN. Questa si basa sulle GAN e costituisce un approccio innovativo nell'addestramento di modelli per la traduzione immagine-a-immagine. L'innovazione

si ha perchè rispetto ai metodi precedentemente progettati la traduzione avviene in assenza di collezioni di immagini appaiate. Abbiamo due funzioni di mappatura G ed F . La funzione G prova a tradurre il dominio X in un dominio Y di output, costituito da dati che vengono analizzati dal discriminatore D_y per verificare se questi sono reali o falsi secondo il dominio Y .

D'altra parte, la funzione F prova a tradurre il dominio Y in un dominio X di output, costituito da dati che vengono analizzati dal discriminatore D_x per verificare se questi sono indistinguibili dal dominio X .

Quindi questo processo di traduzione è di fatto quello che viene definito come cycle consistency. Ciò su cui si focalizzano i ricercatori è la cycle consistency loss, cioè la differenza tra le due immagini presenti nel dominio X (figura 10.b).

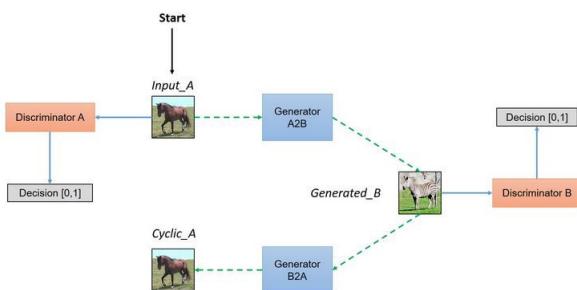


Figure 6: Schema dell'architettura CycleGAN

Quello nella figura sopra è lo schema di un'architettura CycleGAN.

In questo schema, il generatore A2B converte l'immagine di un cavallo in una zebra e il generatore B2A converte una zebra in un cavallo. Entrambi i generatori vengono addestrati insieme per verificare l'immagine del cavallo in ingresso e l'immagine generata della zebra. I due discriminatori determinano se le immagini del cavallo e della zebra siano reali o false. In particolare, Generatore e

Discriminatore sono gli elementi principali di una GAN, e questi sono realizzati come percetroni multistrato (modelli di rete neurale artificiale); l'obiettivo del generatore è quello di sintetizzare immagini che assomigliano a immagini reali, mentre l'obiettivo del discriminatore è quello di distinguere le immagini reali da quelle sintetizzate

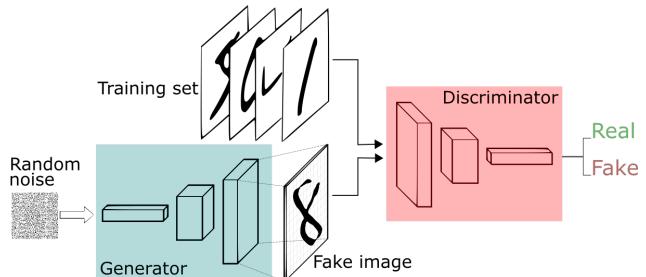


Figure 7: GAN architecture

In sintesi, l'idea è di addestrare contemporaneamente due reti neurali che lavorano insieme (Generatore e Discriminatore).

5.1. Generative vs Discriminative models

In discriminative models, to predict the label y from the training example x , you must evaluate:

$$f(x) = \arg \max_y p(y|x)$$

che si limita a scegliere quale sia la classe più probabile y considerando x . È come se cercassimo di modellare il confine decisionale tra le classi. Questo comportamento è molto chiaro nelle reti neurali, dove i pesi calcolati possono essere visti come una curva di forma complessa che isola gli elementi di una classe nello spazio.

Ora, usando la regola di Bayes, sostituiamo il $p(y|x)$ nell'equazione con

$$\frac{p(x|y)p(y)}{p(x)}$$

Poiché siamo interessati solo al $\arg \max$, possiamo eliminare il denominatore, che sarà lo stesso per ogni y . Quindi, si rimane con

$$f(x) = \arg \max_y p(y|x)p(y)$$

che è l'equazione che si usa nei modelli generativi.

Mentre nel primo caso si aveva la distribuzione di probabilità condizionata $p(y|x)$, che modellava il confine tra le classi, nel secondo si aveva la distribuzione di probabilità congiunta $p(x,y)$, poiché $p(x|y)p(y) = p(x,y)$, che modella esplicitamente la distribuzione effettiva di ogni classe.

Con la funzione di distribuzione di probabilità congiunta, data una y , è possibile calcolare ("generare") la rispettiva x . Per questo motivo, sono chiamati modelli "generativi". Per

fare un esempio, supponiamo di avere due classi di animali, elefante ($y = 1$) e cane ($y = 0$). E x è il vettore di caratteristiche degli animali.

Dato un set di allenamento, un algoritmo come la regressione logistica o l'algoritmo perceptron (fondamentalmente) cerca di trovare una linea retta - cioè un confine di decisione - che separa gli elefanti e i cani. Poi, per classificare un nuovo animale come un elefante o un cane, controlla da quale parte del confine decisionale cade, e fa la sua previsione di conseguenza. Chiamiamo questi algoritmi di apprendimento discriminatorio.

Ecco un approccio diverso. Prima, guardando gli elefanti, possiamo costruire un modello di come sono fatti gli elefanti. Poi, guardando i cani, possiamo costruire un modello separato di come sono fatti i cani. Infine, per classificare un nuovo animale, possiamo confrontare il nuovo animale con il modello di elefante, e confrontarlo con il modello di cane, per vedere se il nuovo animale assomiglia più agli elefanti o più ai cani che avevamo visto nel set di allenamento. Chiamiamo questi algoritmi di apprendimento generativo.

5.2. Architettura GAN: Generatore

Il Generatore $G(Z)$, dove Z è generalmente un vettore campionato casualmente in una distribuzione semplice (es. Gaussiana), produce immagini in modo da addestrare la funzione $D(Y)$ a prendere la forma giusta (valori bassi per le immagini reali, valori più alti per tutto il resto). Durante l'allenamento a D viene mostrata un'immagine reale, e regola il suo parametro per ridurre la sua uscita. Quindi a D

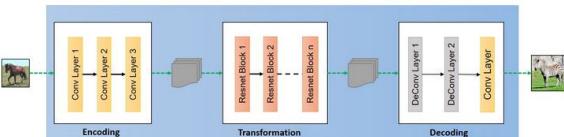


Figure 8: Schema del Generatore

viene mostrata un'immagine prodotta da G e regola i suoi parametri per ingrandire il suo output $D(G(Z))$ (seguendo il gradiente di qualche funzione oggettiva predefinita).

5.2.1 Encoder

Il passo iniziale è separare i punti salienti da un'immagine, e ciò viene fatto da una rete di convoluzione. Come input, una rete di convoluzione prende un'immagine, la dimensione della finestra del filtro che muoviamo sull'immagine di input per estrarre le caratteristiche e la dimensione dello Stride per scegliere la quantità di spostamento della finestra del filtro dopo ogni progressione. Ogni strato di convoluzione richiede l'estrazione di punti salienti di livello dinamicamente più elevato.

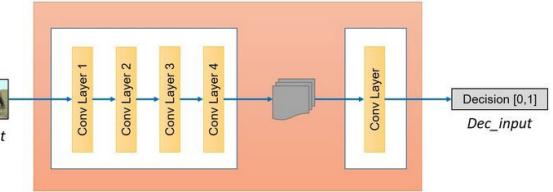


Figure 9: Discriminator schema

5.2.2 Transformer

Si possono vedere questi strati come l'unione di diverse caratteristiche vicine delle immagini e in seguito, a seconda di queste caratteristiche, come scegliere come cambiare la codifica degli elementi di un'immagine a partire da uno e poi al successivo. Per questo, abbiamo utilizzato da 6 a 9 strati di blocchi resnet come segue: Il blocco Resnet è uno strato di rete neurale che comprende due strati di convoluzione dove un accumulo di informazioni viene aggiunto alla resa. Questo viene fatto per garantire che le proprietà del contributo degli strati passati siano accessibili per gli strati successivi anche con l'obiettivo che la loro resa non divaghi molto dalle informazioni uniche, in ogni caso, le qualità delle immagini uniche non saranno tenute nella resa e i risultati saranno inaspettati. Uno dei punti essenziali per l'assegnazione è quello di tenere il tratto delle informazioni uniche come la dimensione e lo stato dell'articolo, quindi i sistemi di avanzi sono una misura straordinaria per questi tipi di cambiamenti. Il blocco Resnet può essere delineato nell'immagine di accompagnamento.

5.2.3 Decoder

La fase di decodifica è l'inverso della fase di codifica; si rielaborano le caratteristiche di basso livello dal vettore di elementi. Questa fase termina applicando un livello di deconvoluzione (o convoluzione di trasposizione).

5.3. Architettura GAN: Discriminatore

Il Discriminatore $D(Y)$ prende un input (es. un'immagine) ed emette uno scalare che indica se l'immagine Y sembra "naturale" o meno. In un caso di addestramento in contraddittorio, $D(Y)$ può sembrare una sorta di funzione energetica che assume un valore basso (ad esempio vicino a 0) quando Y è un campione reale (ad esempio un'immagine da un database) e un valore positivo quando non lo è (ad es. se si tratta di un'immagine rumorosa o dall'aspetto strano).

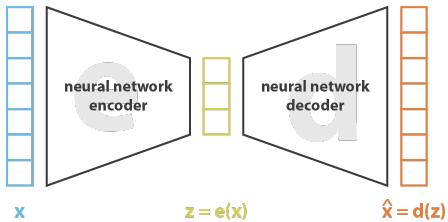
5.4. Concetto di "avversario"

Come già riportato, il Generatore G e il Discriminatore D sono due reti neurali. $G(Z)$ si allenerà a produrre immagini in modo da indurre D a pensare che siano reali. Lo fa ottenendo il gradiente di D rispetto a Y per ogni campione

che produce. In altre parole, sta cercando di ridurre al minimo l'output di D mentre D sta cercando di massimizzarlo. Da qui il nome di addestramento avversario.

5.5. Autoencoders and Variational autoencoders

Un autocodificatore accetta l'input, lo comprime e poi ricrea l'input originale. Questa è una tecnica non supervisionata perché tutto ciò di cui avete bisogno sono i dati originali, senza alcuna etichetta di risultati noti e corretti. I due usi principali di un autocodificatore sono di comprimere i dati in due (o tre) dimensioni in modo che possano essere graficati, e di comprimere e decomprimere immagini o documenti, il che rimuove il rumore nei dati. L'idea generale degli autocodificatori è piuttosto semplice e consiste nell'impostare un codificatore e un decodificatore come reti neurali e imparare il miglior schema di codifica-decodifica utilizzando un processo di ottimizzazione iterativo. Così, ad ogni iterazione alimentiamo l'architettura dell'autoencoder (il codificatore seguito dal decodificatore) con alcuni dati, confrontiamo l'uscita codificata-decodificata con i dati iniziali e retropropaghiamo l'errore attraverso l'architettura per aggiornare i pesi delle reti.



$$\text{loss} = \|x - \hat{x}\|^2 = \|x - d(z)\|^2 = \|x - d(e(x))\|^2$$

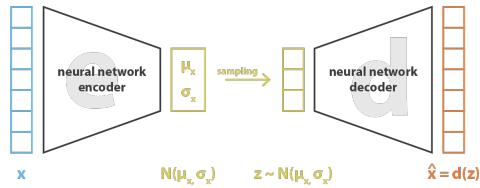
Figure 10: Illustration of an autoencoder with its loss function, [Source: Joseph Rocca, Towards datascience](#))

Un autocodificatore variazionale presuppone che i dati sorgente abbiano un qualche tipo di distribuzione di probabilità sottostante (come la gaussiana) e poi cerca di trovare i parametri della distribuzione. Implementare un autocodificatore variazionale è molto più impegnativo che implementare un autoencoder. L'uso principale di un autoencoder variazionale è quello di generare nuovi dati che sono correlati ai dati originali. Ora, è difficile dire esattamente a cosa servano i dati aggiuntivi. Possiamo vedere uno schema nella figura qui sotto

Un autocodificatore variazionale è un sistema generativo e serve uno scopo simile a quello di una rete generativa avversaria (anche se le GAN funzionano in modo diverso).

5.6. Cycle Consistency

In questo metodo viene introdotto anche il concetto di cycle consistency, e ciò significa che se convertiamo



$$\text{loss} = \|x - \hat{x}\|^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)] = \|x - d(z)\|^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)]$$

Figure 11: In variational autoencoders, the loss function is composed of a reconstruction term (that makes the encoding-decoding scheme efficient) and a regularisation term (that makes the latent space regular),, [Source: Joseph Rocca, Towards datascience](#))



Figure 12: Face images generated with a Variational Autoencoder (source: Wojciech Mormul on [Github](#))

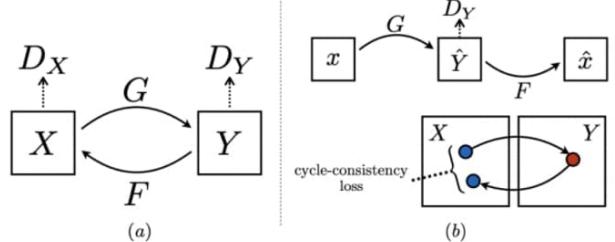


Figure 13: Cycle consistency and cycle consistency loss

l'immagine di una zebra in un'immagine di un cavallo e poi torniamo indietro per riottenere l'immagine della zebra, dovremmo recuperare la stessa immagine di input. Analogamente, data una frase tradotta dall'inglese al francese, se questa viene tradotta nuovamente in inglese, dovrebbe portare alla frase in inglese originaria. In entrambi gli esempi, il processo inverso dovrebbe anche valere.

5.7. Generazione tramite photoshop

In assenza di strumenti che ci permettessero di riprodurre facilmente l'architettura CycleGAN, per renderci conto delle potenzialità dei filtri applicabili ad un'immagine per poterne ottenere un'altra, abbiamo applicato un effetto spennellata ad un'immagine

ottenendo un'altra con un effetto simile ai quadri del pittore Monet. Si parla in questo caso di "trasferimento di



Figure 14: Un Monet originale

stile". I software di Adobe hanno adottato l'AI e il machine learning come è possibile leggere qui[22].

Presentiamo dunque l'immagine originale e quella a cui



Figure 15: Immagine di un paesaggio presa da Pixabay applichiamo il filtro. Naturalmente, ciò che ci aspettiamo



Figure 16: Applicazione di un filtro su Photoshop da un discriminatore è il fatto che questa si "accorga" che l'immagine da noi generata è un falso o, comunque, che dia un risultato abbastanza alto da poter dire che questa immagine non sia un Monet abbastanza preciso; infatti, sebbene, Photoshop possa usare le tecniche di machine learning, non sono presenti esplicitamente le opzioni "applica lo stile Monet", "applica lo stile Cezanne" o simili.

6. Applications

6.1. Collection style transfers, object transfiguration and season transfer

In modo informale possiamo dare le seguenti definizioni

Definition 6.1 Il trasferimento di stile della collezione è una tecnica di ottimizzazione usata per prendere due immagini - un'immagine di contenuto e un'immagine di riferimento dello stile (come un'opera d'arte di un famoso pittore) - e fonderle insieme in modo che l'immagine di uscita assomigli all'immagine di contenuto, ma "dipinta" nello stile dell'immagine di riferimento dello stile.



Figure 17: An example of collection style transfer

Definition 6.2 La trasfigurazione degli oggetti è il processo per trasferire il colore e la struttura di un oggetto all'altro.

Quindi, date un'immagine A e un'immagine B, la trasfigurazione dell'oggetto sostituisce l'oggetto nell'immagine A con un altro oggetto dell'immagine B. Ultimo ma non

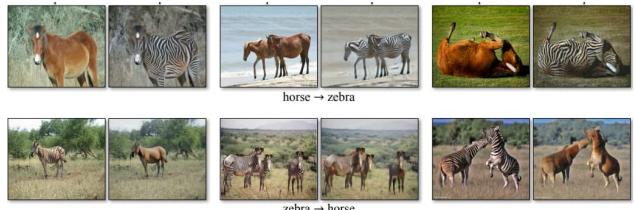


Figure 18: An example of collection object transfiguration meno importante abbiamo i trasferimenti di stagione, "banalmente" quello che cerchiamo di fare in questo caso è solo la variazione della stagione rappresentata in una foto con una stagione scelta da noi.



Figure 19: An example of collection season transfer

6.2. Image-To-Image Translations

La traduzione da immagine a immagine è una classe di problemi di visione e grafica in cui l'obiettivo è imparare

la mappatura tra un’immagine in ingresso e un’immagine in uscita. Può essere applicato a una vasta gamma di applicazioni, come il trasferimento di stile di collezione, la trasfigurazione di oggetti, il trasferimento di stagione e il miglioramento delle foto.

6.2.1 Paired vs Unpaired

Per la traduzione, nonché l’addestramento, del modello, i ricercatori si possono avvalere di due tipi di dati: appaiati e non appaiati.

Cosa significa traduzione con dati non appaiati?

I dati di addestramento accoppiati (a sinistra) sono costituiti da esempi di addestramento $\{x_i, y_i\}_{i=1}^N$, dove esiste la corrispondenza tra x_i, y_i . Consideriamo invece dati di addestramento non appaiati (a destra), costituiti da un insieme sorgente $\{x_i\}_{i=1}^N, (x_i \in X)$ e un insieme obiettivo $\{y_j\}_{j=1}^N, (y_j \in Y)$, senza informazioni fornite su quale x_i corrisponde a quale y_j .

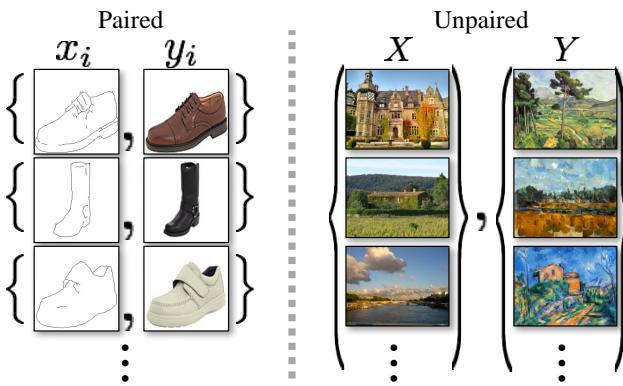


Figure 20: Paired and unpaired sets

Le architetture GAN condizionali sono state ampiamente applicate in compiti di traduzione da immagine a immagine, quando è disponibile una quantità sufficiente di esempi di training accoppiati. In particolare, è stato proposto un traduttore generale da immagine a immagine ”pix2pix” per varie applicazioni come mappe semantiche a immagini reali, bordi a immagini di scarpe, e immagini da grigio a colore e, questo lavoro, è stato successivamente esteso per produrre output ad alta risoluzione.

6.2.2 Paired: PIX2PIX

PIX2PIX utilizza una rete generativa condizionale avversaria (cGAN) per imparare una mappatura da un’immagine di input a un’immagine di output. Un esempio di dataset potrebbe essere che l’immagine di input è una foto in bianco e nero e l’immagine di destinazione è la versione a colori della foto. Il generatore in questo caso sta cercando di imparare come colorare un’immagine in bianco e nero. Il discriminatore guarda i tentativi di colorazione del generatore

e cerca di imparare a distinguere la differenza tra le colorazioni che il generatore fornisce e la vera immagine di destinazione colorata fornita nel dataset.

6.2.3 Unpaired: CycleGAN

Mentre PIX2PIX può produrre risultati davvero magici, la sfida sta nei dati di allenamento. I due spazi di immagine tra cui si voleva imparare a tradurre dovevano essere preformati in una singola immagine X/Y che contenesse entrambe le immagini strettamente correlate. Questo potrebbe essere dispendioso in termini di tempo, irrealizzabile o addirittura impossibile in base ai due tipi di immagine tra i quali si stava cercando di tradurre (per esempio, se non si aveva una corrispondenza uno-a-uno tra i due profili di immagine). È qui che entra in gioco il CycleGAN. L’idea chiave dietro le CycleGAN è che possono basarsi sulla potenza dell’architettura PIX2PIX, ma permettono di puntare il modello a due collezioni discrete e non accoppiate di immagini. Per esempio, una collezione di immagini, il Gruppo X, sarebbe piena di foto di spiagge soleggiate mentre il Gruppo Y sarebbe una collezione di foto di spiagge nuvolose. Il modello CycleGAN può imparare a tradurre le immagini tra queste due estetiche senza la necessità di unire le corrispondenze strettamente correlate in una singola immagine di allenamento X/Y.

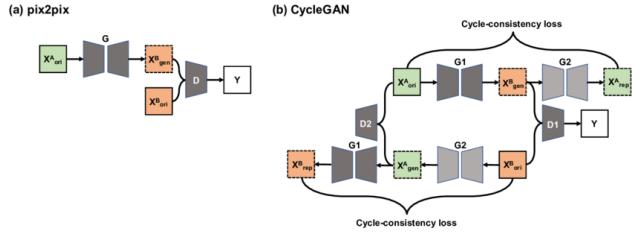


Figure 21: Architecture comparison

6.3. Other Reasons Why Unsupervised Learning is Crucial

Una delle affermazioni di questo articolo è che un potente apprendimento non supervisionato o semi-supervisionato è una componente cruciale nella costruzione di algoritmi di apprendimento di successo per architetture profonde finalizzate all’IA. Noi trattiamo brevemente gli argomenti a favore di questa ipotesi:

- compiti futuri sconosciuti: se un agente di apprendimento non sa quali compiti di apprendimento futuri dovrà affrontare in futuro, ma sa che il compito sarà definito rispetto a un mondo (cioè variabili casuali variabili casuali) che può osservare ora, sembrerebbe molto razionale raccogliere quante più informazioni possibile su questo mondo in modo da imparare cosa lo fa funzionare.

- Una volta appresa una buona rappresentazione di alto livello, altri compiti di apprendimento (per esempio, apprendimento supervisionato o di rinforzo) potrebbero essere molto più facili. Sappiamo per esempio che le macchine a kernel possono essere molto potenti se si usa un kernel appropriato, cioè uno spazio delle caratteristiche appropriato. Allo stesso modo, conosciamo potenti algoritmi di apprendimento per rinforzo che hanno garanzie nel caso in cui le azioni sono essenzialmente ottenute attraverso la combinazione lineare di caratteristiche appropriate. Non sappiamo quale sia la rappresentazione rappresentazione appropriata, ma si sarebbe rassicurati se essa catturasse i fattori salienti di variazione nei dati di input e li distingue.
- Apprendimento non supervisionato a livelli: questo è stato argomentato nella sezione 5.3. Gran parte dell'apprendimento potrebbe essere fatto utilizzando le informazioni disponibili localmente in uno strato o sottolivello dell'architettura, evitando così i problemi ipotizzati con gradienti supervisionati che si propagano attraverso lunghe catene con grandi elementi elementi.
- Collegato ai due punti precedenti è l'idea che l'apprendimento non supervisionato potrebbe mettere i parametri di una macchina di apprendimento supervisionato o di rinforzo in una regione dalla quale la discesa del gradiente (ottimizzazione locale) produrrebbe buone soluzioni. Questo è stato verificato empiricamente nei casi studiati in Bengio et al. (2007).
- Meno incline all'overfitting: è stato sostenuto (Hinton, 2006) che l'apprendimento non supervisionato è meno incline all'overfitting rispetto all'apprendimento supervisionato. L'intuizione è la seguente. Quando si fa la classificazione discriminante Quando si fa la classificazione discriminante, si ha solo bisogno di imparare una funzione le cui variazioni sono importanti vicino al confine della decisione. Un sottoinsieme molto piccolo delle variazioni di input potrebbe essere rilevante per scoprire la classificazione corretta. D'altra parte, l'apprendimento non supervisionato cerca di catturare tutte le variazioni dell'input. Perciò esso richiede molta più capacità, o equivalentemente, è meno incline all'overfitting a parità di capacità e di numero di esempi di formazione. L'apprendimento non supervisionato può essere usato per inizializzare o regolarizzare nel contesto dei sistemi di apprendimento supervisionato.
- I vincoli extra imposti all'ottimizzazione richiedendo al modello di catturare non solo la dipendenza tra input e target ma anche le regolarità statistiche della distribuzione degli input potrebbero essere utili per

evitare alcuni minimi locali (quelli che non corrispondono a una buona modellazione della distribuzione di input).

6.4. Face generation: StyleGAN

StyleGAN è un'architettura generativa alternativa per reti generative avversarie, mutuata dalla letteratura sul trasferimento di stile. La nuova architettura porta ad una separazione automatica e non supervisionata degli attributi di alto livello (ad esempio, la posa e l'identità quando addestrati su volti umani) e la variazione stocastica nelle immagini generate (ad esempio, lentiggi, capelli), e consente un controllo intuitivo e specifico per la scala della sintesi.

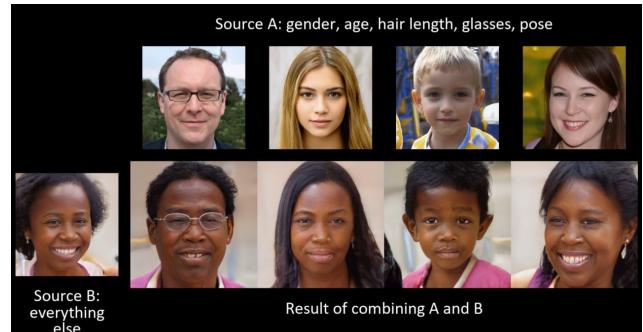


Figure 22: This is how faces are combined to generate new faces

Il nuovo generatore migliora lo stato dell'arte in termini di metriche tradizionali di qualità della distribuzione, porta a proprietà di interpolazione dimostrabilmente migliori, e inoltre distingue meglio i fattori latenti di variazione.



Figure 23: Generated faces

7. Other methods

Accenniamo brevemente le principali architetture contro cui è stata messa a confronto la CycleGAN.

7.1. BiGAN

Una BiGAN, o GAN bidirezionale, è un tipo di GAN in cui il generatore non solo mappa i campioni latenti sui

dati generati, ma ha anche una mappatura inversa dai dati alla rappresentazione latente. La motivazione è quella di creare un tipo di GAN in grado di apprendere rappresentazioni ricche per noi in applicazioni come l'apprendimento non supervisionato.

7.2. ALI

The adversarially learned inference (ALI) model is a deep directed generative model which jointly learns a generation network and an inference network using an adversarial process. This model constitutes a novel approach to integrating efficient inference with the generative adversarial networks (GAN) framework.

What makes ALI unique is that unlike other approaches to learning inference in deep directed generative models (like variational autoencoders (VAEs)), the objective function involves no explicit reconstruction loop.

7.3. CoGAN

Una CoGAN, o GAN accoppiata, è costituita da una copia di GAN: GAN1 e GAN2. Ciascuna ha un modello generativo per sintetizzare immagini realistiche in un dominio e un modello discriminativo per classificare se un'immagine è reale o sintetizzata. Una CoGAN addestrata può essere utilizzata per sintetizzare coppie di immagini/coppie di immagini corrispondenti che condividono la stessa astrazione di alto livello ma con realizzazioni di basso livello diverse.

8. Misurazioni

8.1. AMT: Amazon Mechanical Turk (AMT)

Amazon Mechanical Turk (MTurk) è un servizio internet di crowdsourcing che permette ai programmati informatici di coordinare l'uso di intelligenze umane per eseguire compiti che i computer, ad oggi, non sono in grado di fare, ed è una delle suite di Amazon Web Services. AMT è stato utilizzato per il task mappa→foto aerea per valutare il realismo dei risultati. Ai partecipanti è stata mostrata una sequenza di coppie di immagini, una foto o una mappa reale e una falsa (generata dall'algoritmo o da un riferimento) e gli è stato chiesto di fare click sull'immagine che pensavano fosse reale.

8.2. FCN

L'FCN score è stato utilizzato per valutare i task etichette di paesaggi urbani→foto, e valuta quanto le foto generate siano interpretabili secondo un algoritmo di segmentazione semantica.

Una FCN (Fully Convolutional Network), in particolare, predice una mappa di etichette per una foto generata, che può essere confrontata con le etichette reali di input, utilizzando le metriche standard di segmentazione semantica. Quindi, se generiamo una foto da una mappa di

etichette di "auto sulla strada", allora abbiamo successo se la FCN applicata alla foto generata rileva "auto sulla strada".

References

- [1] Jason Brownlee. *4 Types of Classification Tasks in Machine Learning*. en-US. Apr. 2020. URL: <https://machinelearningmastery.com/types-of-classification-in-machine-learning/> (visited on 06/02/2021).
- [2] Jason Brownlee. *Overfitting and Underfitting With Machine Learning Algorithms*. en-US. Mar. 2016. URL: <https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/> (visited on 06/02/2021).
- [3] *Classification In Machine Learning — Classification Algorithms*. en-US. Dec. 2019. URL: <https://www.edureka.co/blog/classification-in-machine-learning/> (visited on 06/02/2021).
- [4] *English: How deep learning is a subset of machine learning and how machine learning is a subset of artificial intelligence (AI)*. May 2020. URL: <https://commons.wikimedia.org/wiki/File:AI-ML-DL.svg> (visited on 06/01/2021).
- [5] *Español: Diagrama simplificado de predicción de un nuevo correo electrónico, mediante un clasificador, en categorías spam ó no-spam*. Oct. 2020. URL: https://commons.wikimedia.org/wiki/File:Spam_VS_Not_Spam_Email_Classification_Diagram.png (visited on 06/02/2021).
- [6] *Feature (machine learning)*. en. Page Version ID: 1023358590. May 2021. URL: [https://en.wikipedia.org/w/index.php?title=Feature_\(machine_learning\)&oldid=1023358590](https://en.wikipedia.org/w/index.php?title=Feature_(machine_learning)&oldid=1023358590) (visited on 06/02/2021).
- [7] *Image segmentation*. en. Page Version ID: 1026516450. June 2021. URL: https://en.wikipedia.org/w/index.php?title=Image_segmentation&oldid=1026516450 (visited on 06/02/2021).
- [8] *Image Segmentation — Segmentazione semantica e delle istanze*. it-IT. Nov. 2020. URL: <https://andreaprovino.it/image-segmentation-segmentazione-semantica-e-delle-istanze/> (visited on 06/02/2021).
- [9] Phillip Isola et al. “Image-to-Image Translation with Conditional Adversarial Networks”. In: *arXiv:1611.07004 [cs]* (Nov. 2018). arXiv: 1611.07004. URL: <http://arxiv.org/abs/1611.07004> (visited on 06/03/2021).
- [10] jamesdmccaffrey. *The Difference Between an Autoencoder and a Variational Autoencoder*. en. July 2018. URL: <https://jamesdmccaffrey.wordpress.com/2018/07/03/the-difference-between-an-autoencoder-and-a-variational-autoencoder/> (visited on 06/03/2021).
- [11] joydeep bhattacharjee joydeep. *Some Key Machine Learning Definitions*. en. Nov. 2017. URL: <https://medium.com/technology-nineleaps/some-key-machine-learning-definitions-b524eb6cb48> (visited on 06/02/2021).
- [12] joydeep bhattacharjee joydeep. *Some Key Machine Learning Definitions*. en. Nov. 2017. URL: <https://medium.com/technology-nineleaps/some-key-machine-learning-definitions-b524eb6cb48> (visited on 06/02/2021).
- [13] Tero Karras, Samuli Laine, and Timo Aila. “A Style-Based Generator Architecture for Generative Adversarial Networks”. In: *arXiv:1812.04948 [cs, stat]* (Mar. 2019). arXiv: 1812.04948. URL: <http://arxiv.org/abs/1812.04948> (visited on 05/31/2021).
- [14] *Lecture 1 - Welcome — Stanford CS229: Machine Learning (Autumn 2018)*. it-IT. URL: https://www.youtube.com/watch?v=jGwO_UgTS7I (visited on 06/03/2021).
- [15] *machine learning - What is the difference between a generative and a discriminative algorithm?* URL: <https://stackoverflow.com/questions/879432/what-is-the-difference-between-a-generative-and-a-discriminative-algorithm> (visited on 06/03/2021).
- [16] *machine learning - What is the difference between test set and validation set?* URL: <https://stats.stackexchange.com/questions/19048/what-is-the-difference-between-test-set-and-validation-set> (visited on 06/02/2021).
- [17] *Machine Learning: Pattern Recognition*. en. URL: <https://serokell.io/blog/pattern-recognition> (visited on 06/02/2021).
- [18] Denis Malimonov. *tg-bomze/Style-Transfer-Collection*. original-date: 2020-07-06T07:19:42Z. May 2021. URL: <https://github.com/tg-bomze/Style-Transfer-Collection> (visited on 06/03/2021).

- [19] Wojciech Mo. *wojciechmo/vae*. original-date: 2018-01-01T16:49:04Z. Feb. 2021. URL: <https://github.com/wojciechmo/vae> (visited on 06/03/2021).
- [20] *Papers with Code - ALI Explained*. en. URL: <https://paperswithcode.com/method/ali> (visited on 06/03/2021).
- [21] *Pattern recognition*. en. Page Version ID: 1001993739. Jan. 2021. URL: https://en.wikipedia.org/w/index.php?title=Pattern_recognition&oldid=1001993739 (visited on 06/02/2021).
- [22] *Photoshop: Now the world's most advanced AI application for creatives*. URL: <https://blog.adobe.com/en/publish/2020/10/20/photoshop-the-worlds-most-advanced-ai-application-for-creatives.html> (visited on 06/03/2021).
- [23] Waseem Rawat and Zenghui Wang. “Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review”. en. In: *Neural Computation* 29.9 (Sept. 2017), pp. 2352–2449. ISSN: 0899-7667, 1530-888X. DOI: [10.1162/neco_a_00990](https://doi.org/10.1162/neco_a_00990). URL: <https://direct.mit.edu/neco/article/29/9/2352-2449/8292> (visited on 06/02/2021).
- [24] *Recurrent neural network*. en. Page Version ID: 1025306829. May 2021. URL: https://en.wikipedia.org/w/index.php?title=Recurrent_neural_network&oldid=1025306829 (visited on 06/02/2021).
- [25] Tarang Shah. *About Train, Validation and Test Sets in Machine Learning*. en. July 2020. URL: <https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7> (visited on 06/02/2021).
- [26] *Tutorial - What is a variational autoencoder?* URL: <https://jaan.io/what-is-variational-autoencoder-vae-tutorial/> (visited on 06/03/2021).
- [27] *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*. en. URL: <https://junyanz.github.io/CycleGAN/> (visited on 06/03/2021).
- [28] *Visione artificiale*. it. Page Version ID: 119542822. Mar. 2021. URL: https://it.wikipedia.org/w/index.php?title=Visione_artificiale&oldid=119542822 (visited on 06/02/2021).
- [29] *Weight (Artificial Neural Network)*. May 2019. URL: <https://deepai.org/machine-learning-glossary-and-terms/weight-artificial-neural-network> (visited on 06/02/2021).
- [30] *Weights and Biases*. URL: <https://docs.paperspace.com/machine-learning/wiki/weights-and-biases> (visited on 06/02/2021).
- [31] Richard Zhang, Phillip Isola, and Alexei A. Efros. “Colorful Image Colorization”. In: *arXiv:1603.08511 [cs]* (Oct. 2016). arXiv: 1603.08511. URL: <http://arxiv.org/abs/1603.08511> (visited on 05/31/2021).
- [32] Jun-Yan Zhu et al. “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”. In: *arXiv:1703.10593 [cs]* (Aug. 2020). arXiv: 1703.10593. URL: <http://arxiv.org/abs/1703.10593> (visited on 06/03/2021).