# **TRASFORMAZIONE IMMAGINI SENSIBILI IN FUMETTI**

## *INDICE*

1. [*Introduzione*](#_In_cosa_consiste)
2. [*Recupero dei link*](#_Recupero_dei_link)
3. [*Setup pre-testing*](#_SETUP_PRE-TESTING)
4. [*Testing*](#_TESTING)
5. [*Target*](#_TARGET)
6. [*Results*](#_RESULTS)
7. [*Colab*](#_Colab)

## **INTRODUZIONE**

Progetto realizzato al fine di trasformare immagini dal contenuto sensibile (es. foto di incidenti, foto di operazioni chirurgiche, ecc.) in immagini fumettizzate tramite il metodo **l’Informative Drawings,** così da facilitarne la visione.

Il metodo «**Informative Drawings**» è un metodo che riesce a creare disegni, partendo da immagini statiche.

I metodi attuali non fanno altro che generare disegni a partire da un dataset, ma molte volte questo è causa di problemi e di limitazioni, come ad esempio il fatto di rimanere all’interno dello stesso dominio del dataset e quindi focalizzarsi su un singolo soggetto.

Informative Drawings utilizza l’addestramento di una rete neurale basata sull’**image-translation** e l’introduzione di una perdita geometrica (***geometric loss***), la quale riesce a generare disegni e linee, partendo dall’immagine principale.

## **RECUPERO DEI LINK**

Dopo aver parlato con il Professore e aver deciso che tipo di progetto portare a termine, Egli mi ha inviato alcuni link da dove scaricare i vari dataset e da dove prendere spunto per la portata al termine del progetto.

Una volta terminato il download del dataset di “GTA V” (<https://paperswithcode.com/task/image-to-image-translation>), da cui prenderemo le immagini per analizzare gli esempi, ho notato che purtroppo nei link inviati dal Professore relativi ai vari metodi di trasformazione, non era presente nessun metodo che permettesse la trasformazione di un’immagine normale in un’immagine fumetizzata (Images to Edges), ma solamente l’esatto contrario (Edges to Images), il quale ha portato ad uno stallo nel corretto proseguimento del progetto.

Poiché effettuare l’inversione del codice non era possibile, ho deciso di contattare il creatore del progetto in questione chiedendo se avesse un repository GitHub utile per il nostro obbiettivo.

Quest’ultimo mi ha inviato i seguenti link da cui prendere spunto:

1. <https://github.com/s9xie/hed>
2. <https://clipasso.github.io/clipasso/>
3. <https://carolineec.github.io/informative_drawings/>

Dopo varie letture e consultazioni ho deciso che sarebbe stato più efficiente procedere con il terzo link.

## **SETUP PRE-TESTING**

Dopo aver scaricato e salvato il dataset di GTA la prima cosa che ho fatto è stata copiare in locale il Repository, presente nel link scelto da me tra quelli inviati dal creatore diretto del GitHub, con il comando di gitBash «git clone <https://github.com/carolineec/informative-drawings.git>»

Successivamente creiamo un **ambiente Anaconda**, con le varie dipendenze e librerie necessarie per il corretto funzionamento di Informative Drawings, attraverso il caricamento di un file fornitoci dallo sviluppatore (*environment.yml*) e tramite il comando successivo eseguito sul prompt di comando di Anaconda:

«*conda env create -f environment.yml*»

Immagine che contiene testo

Descrizione generata automaticamente

Infine, attiviamo l’ambiente Anaconda appena creato con il seguente comando: *«conda activate drawings»*Immagine che contiene testo

Descrizione generata automaticamente

## **TESTING**

Seguendo il README.md all’interno del repository GitHub, è stato scaricato il file .zip contenente il **modello** **pre-addestrato** e successivamente estratto in una cartella chiamata ‘*’checkpoints*’’.

In seguito, all’interno della cartella «*examples*», è stata creata un’ulteriore cartella di nome ‘’**GTA**’’ contenente le nostre immagini di prova scaricate tramite i link distribuiti inizialmente dal professore.

Infine, è stato eseguito il comando principale del **testing**, caricando i pesi del modello scaricato al punto 1 e passando in input le nostre immagini di prova esplicitate al punto 2. Il comando è:

«*python test.py --name anime\_style --dataroot examples/GTA*»

Immagine che contiene testo

Descrizione generata automaticamente

* Nel comando, attraverso il parametro «*--name*», specifichiamo che le nostre immagini debbano essere processate attraverso il **motore grafico** «*anime\_style*» facente parte del modello pre-addestrato, poiché è il motore grafico più adatto alle nostre foto, mentre tramite parametro «--dataroot», viene specificato il **percorso** dalla quale verranno caricate le nostre immagini.
* Tutti i risultati, verranno salvati in una cartella di **default** denominata ‘*’results*’’. È possibile cambiare cartella di output, attraverso il parametro «--results\_dir».

## **TARGET**



## **RESULTS**

Immagine che contiene testo, esterni, cielo, strada

Descrizione generata automaticamenteImmagine che contiene esterni, cielo, bicicletta, albero

Descrizione generata automaticamenteImmagine che contiene testo, cielo, automobile, esterni

Descrizione generata automaticamente

Immagine che contiene testo, neve, esterni, disegnoatratteggio

Descrizione generata automaticamenteImmagine che contiene esterni, trasporto, disegnoatratteggio, bicicletta

Descrizione generata automaticamenteImmagine che contiene disegnoatratteggio

Descrizione generata automaticamente

## **COLAB**

Infine ho proceduto alla creazione di un Colab interattivo (<https://colab.research.google.com/drive/1ca70VZU44h64QIMmcmXDvlgFQ0owX0jz?usp=sharing>) ai fini di testing con possibilità di caricamento di immagini a piacere.

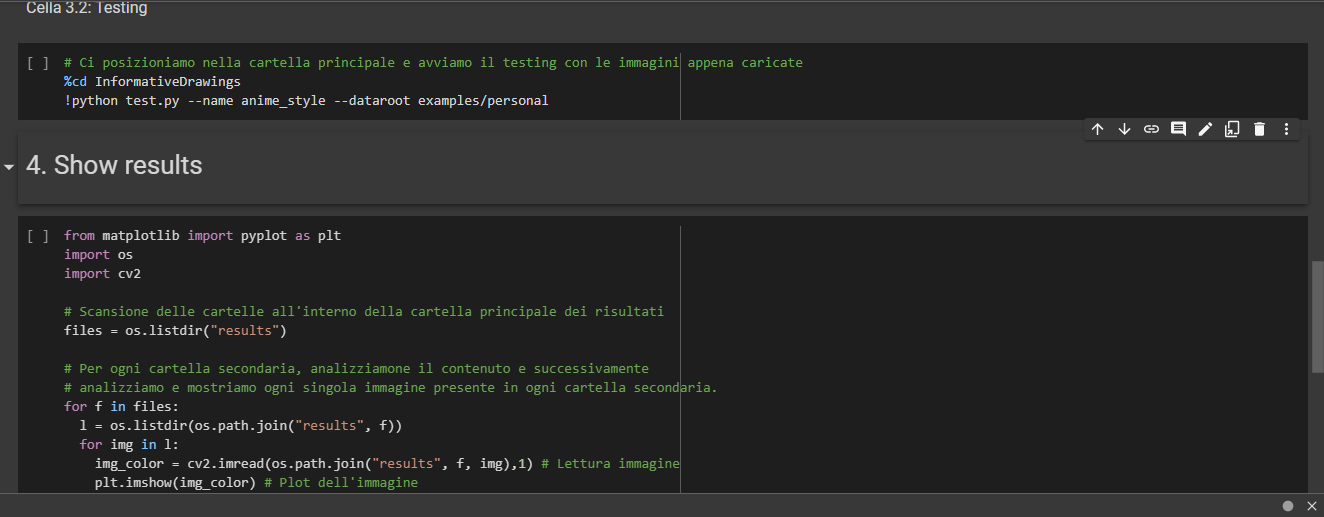
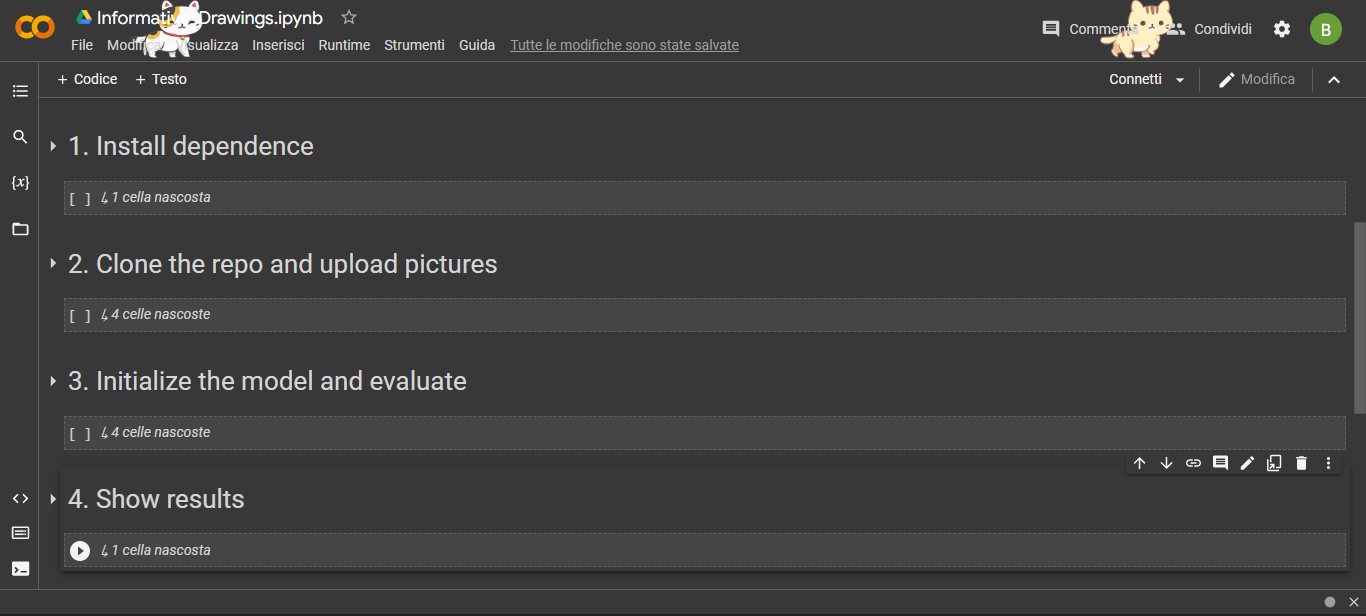


Immagine che contiene testo

Descrizione generata automaticamente