

# Documentazione della Pipeline Immagini

## xai\_img

July 3, 2025

### 1. Contesto

Lo scopo principale di questa pipeline è confrontare visivamente e quantitativamente le spiegazioni generate da due diversi metodi di interpretazione su immagini analizzate da modelli di machine learning. Nello specifico, considerata una determinata immagine, verranno identificate e quantificate le differenze significative (*Disagreement Problem*) nell'attribuzione di importanza a differenti regioni della stessa immagine, influenzando così la decisione finale del modello.

In pratica, dati i seguenti:

- una rete neurale pre-addestrata (ResNet-18);
- una singola immagine ( in questo caso la foto di un gatto);
- due metodi di interpretazione (Integrated Gradients e Saliency);

la pipeline produrrà due mappe di salienza, che rappresentano visivamente le zone dell'immagine considerate determinanti per la decisione finale. Successivamente, queste mappe verranno confrontate utilizzando metriche numeriche per stabilire se le corrispettive spiegazioni divergono, e, nel caso, quantificare le differenze.

Questo confronto evidenzia come l'utilizzo di metodi di valutazione differenti potrebbero portare sia a considerare significative le regioni dell'immagine simili per posizione, oppure identificare regioni completamente diverse, generando così confusione o difficoltà nell'interpretazione delle decisioni da parte del modello.

### 2. Tecnologie e librerie

È stato utilizzato Python 3.10 in ambiente **anaconda** (**xai**), con:

- **PyTorch** e **torchvision** per il caricamento della rete pre-addestrata;
- **Captum** per i metodi di interpretazione;
- **scikit-image** per la segmentazione in super-pixel;
- **NumPy** per la gestione delle mappe di salienza;
- **Matplotlib** per la visualizzazione grafica dei risultati.

### 3. Metodi di interpretazione

I metodi di interpretazione sono tecniche che permettono di comprendere le motivazioni per le quali un modello di intelligenza artificiale ha preso una certa decisione. In questo caso, si utilizzano due metodi:

### 3.1 Integrated Gradients (IG)

Integrated Gradients è un metodo che cerca di spiegare l'importanza di ciascun pixel di un'immagine. Funziona così:

- si parte da un'immagine completamente nera ( per indicare l'assenza di informazioni);
- si calcola gradualmente quanto ogni pixel dell'immagine reale cambia il risultato del modello mentre "si accende" (vale a dire mentre passa da nero al suo valore reale);
- si sommano tutti questi piccoli cambiamenti per ogni pixel.

Il risultato è una mappa in cui ogni pixel ha un valore che indica quanto ha influenzato la decisione.

Più è alto il numero di passi (**n\_steps**), più precisa è la stima, ma il calcolo richiede più tempo.

### 3.2 Saliency (gradiente puro)

Saliency è un metodo molto più semplice. Calcola direttamente quanto la predizione del modello cambierebbe se modificassimo leggermente ogni singolo pixel. Più il cambiamento è grande, più quel pixel è importante.

Differenze principali rispetto a IG:

- **Veloce**: richiede un solo calcolo;
- **Rumoroso**: spesso mette in evidenza molti pixel sparsi, anche se non sono realmente importanti;
- **Non richiede baseline**: non parte da un'immagine nera.

## 4. Flusso di lavoro

La pipeline è suddivisa in due fasi principali: creazione delle mappe e confronto tra di esse.

### 4.1 Creazione delle mappe (make\_maps.py)

Questa fase genera due mappe di salienza, una derivata da Integrated Gradients e una da Saliency, si suddividono in:

1. **Caricamento modello**: si utilizza ResNet-18 pre-addestrata su ImageNet.
2. **Caricamento immagine**: l'immagine (in questo caso `cat.jpg`) viene ridimensionata a 224x224, convertita in tensore e normalizzata secondo lo standard del modello.
3. **Integrated Gradients**:
  - si definisce una baseline (immagine vuota);
  - si calcola l'importanza di ogni pixel seguendo il metodo IG;
  - si somma sui canali, cioè una dimensione dell'immagine che rappresenta un colore specifico, come rosso, verde o blu; in questo caso, l'immagine ha 3 canali per i colori RGB, infine si salva la mappa come `saliency_A.npy`.
4. **Saliency (gradiente puro)**:
  - si calcola direttamente il gradiente rispetto all'immagine;

- si somma sui canali e si salva la mappa come `saliency_B.npy`.

Alla fine, il file stampa:

`saliency_A.npy` e `saliency_B.npy` salvati.

Questi due file contengono le mappe da confrontare nella fase successiva.

## 4.2 Confronto delle mappe (`compare_saliency.py`)

Questa fase serve a confrontare quantitativamente le due mappe generate, per capire se evidenziano le stesse regioni dell'immagine oppure no.

1. **Caricamento delle mappe e immagine:** le due mappe salvate vengono caricate. L'immagine originale viene ridimensionata per avere la stessa dimensione delle mappe.
2. **Segmentazione in super-pixel (SLIC):** L'immagine viene divisa in 200 "super-pixel" tramite l'algoritmo SLIC, cioè piccoli blocchi omogenei di pixel simili. Ogni blocco avrà un'etichetta diversa.
3. **Creazione dei vettori di salienza:** per ciascun super-pixel si calcola la media dei valori di salienza. Si ottengono così due vettori (uno per ogni mappa), che rappresentano l'importanza media di ciascun blocco.
4. **Metriche di confronto:** Si calcolano quattro misure che quantificano il disaccordo tra i due vettori:
  - **FeatureDisagreement:** misura quanta sovrapposizione c'è tra i blocchi più importanti nelle due mappe.
  - **SignDisagreement:** come il Feature Disagreement, ma tiene conto anche del segno.
  - **Euclidean:** distanza globale tra i due vettori (con segno).
  - **Euclidean-abs:** distanza globale tra i due vettori, ignorando il segno.

Il parametro `K_FRAC = 0.05` nel codice indica che si considera il top 5% dei super-pixel più importanti.

5. **Visualizzazione:** Viene creato un overlay che mostra l'immagine originale affiancata dalle due mappe colorate. I colori indicano l'importanza: rosso = alta, blu = bassa.

Il programma stampa a video i quattro valori numerici delle metriche e mostra graficamente le differenze tra le due spiegazioni.

## 5. Risultati

Eseguendo:

```
python make_maps.py
python compare_saliency.py
```

si ottiene:

```
FeatureDisagreement = 1.000,
SignDisagreement = 1.000,
Euclidean = 1.417,
Euclidean-abs = 0.856.
```

Questi valori indicano disaccordo massimo sulle top-5% di super-pixel e grande distanza globale, confermando che le due mappe (IG e Saliency) evidenziano aree significativamente diverse.

## 5.1 Interpretazione quantitativa

- **FeatureDisagreement = 1.000**: fra i super-pixel “top-k” scelti da saliency A e quelli scelti da saliency B, non c'è alcuna sovrapposizione (100% di disaccordo).
- **SignDisagreement = 1.000**: anche considerando il segno dell'attribuzione (positivo vs negativo), non c'è alcuna corrispondenza nelle stesse regioni.
- **Euclidean  $\approx 1.417$** : misura “globale” di distanza fra le due distribuzioni di saliency (vettori normalizzati): quindi le due mappe sono molto diverse.
- **Euclidean-abs  $\approx 0.856$** : stessa distanza ma ignorando il segno: resta elevata, il che conferma che anche le intensità sono molto differenti.

## 6. Le tre immagini



Figure 1: Da sinistra: immagine originale, heat-map IG (saliency A), heat-map Saliency (saliency B).

### 6.1 Immagine originale

La foto originale (`cat.jpg`), ridimensionata a  $224 \times 224$  pixel per coincidere con le mappe.

### 6.2 Saliency A

Overlay della mappa generata con Integrated Gradients. I colori (scala dal verde al rosso, mappati con `plt.cm.jet`) mostrano le aree che IG ritiene importanti: più una zona è “rossastra”, più pesa sulla decisione del modello. Qui emerge una colorazione diffusa, con pochissime aree calde concentrate.

### 6.3 Saliency B

Overlay della mappa generata con il metodo Saliency (gradiente puro). Qui il colore prevalente è bluastro, con pochi punti verde-chiaro che indicano dove il gradiente è massimo.

## 7. Discussione sul Disagreement Problem

Nessuna sovrapposizione nelle aree top-k ( $\text{FeatureDisagreement} = 1$ )  $\rightarrow$  IG e gradiente puro indicano regioni completamente diverse come “più importanti”.

Anche le intensità complessive non sono vicine ( $\text{Euclidean} = 1.417$ )  $\rightarrow$  l'intero pattern di saliency è distante.

Visivamente, IG ha evidenziato una zona ampia e sfumata (al centro del manto), mentre Saliency ha punti sparsi in tutta l'immagine.

Due explainers, applicati allo stesso modello e alla stessa immagine, raccontano storie molto diverse su quali pixel abbiano spinto la predizione.

## Glossario

- **Salienza (saliency map)**: immagine che mostra quali parti dell'immagine originale hanno contato di più nella decisione del modello.
- **Gradiente**: misura di quanto cambierebbe il risultato del modello se modificassi appena il valore di un pixel (quanto “sensibile” è il modello a quel pixel).
- **Integrated Gradients**: tecnica che valuta l'importanza di ogni pixel sommando il contributo dei gradienti mentre l'immagine passa da vuota a reale.
- **Super-pixel**: gruppi di pixel simili tra loro, trattati come blocchi per semplificare il confronto tra mappe.
- **Heatmap**: rappresentazione visiva con colori, dove ad esempio il rosso mostra le zone importanti e il blu quelle meno importanti.