Painting Retrieval

Davide Pietrasanta - 844824
Giuseppe Magazzù - 829612
Gaetano Magazzù - 829685

Dataset - Painter By Numbers

Il dataset consiste in 103.250 immagini di quadri classificati in:

- 2319 artisti
- **42 generi**
- 136 stili pittorici

Ad ogni quadro sono associati titolo, data, fonte.

Partizionamento fornito train 77%, test 23%.

https://www.kaggle.com/c/painter-by-numbers/

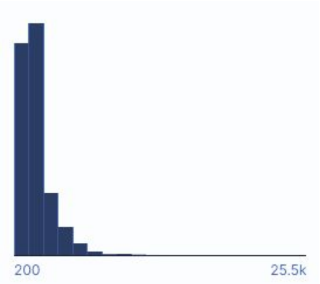
Dataset - Painter By Numbers

- Immagini con dimensioni variabili
- Alta varianza intra-classe e inter-classe
- 19 immagini corrotte

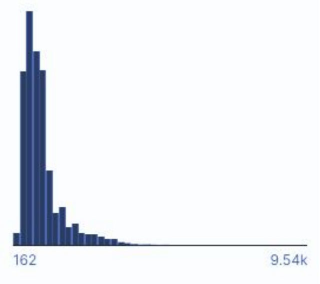
Genere:

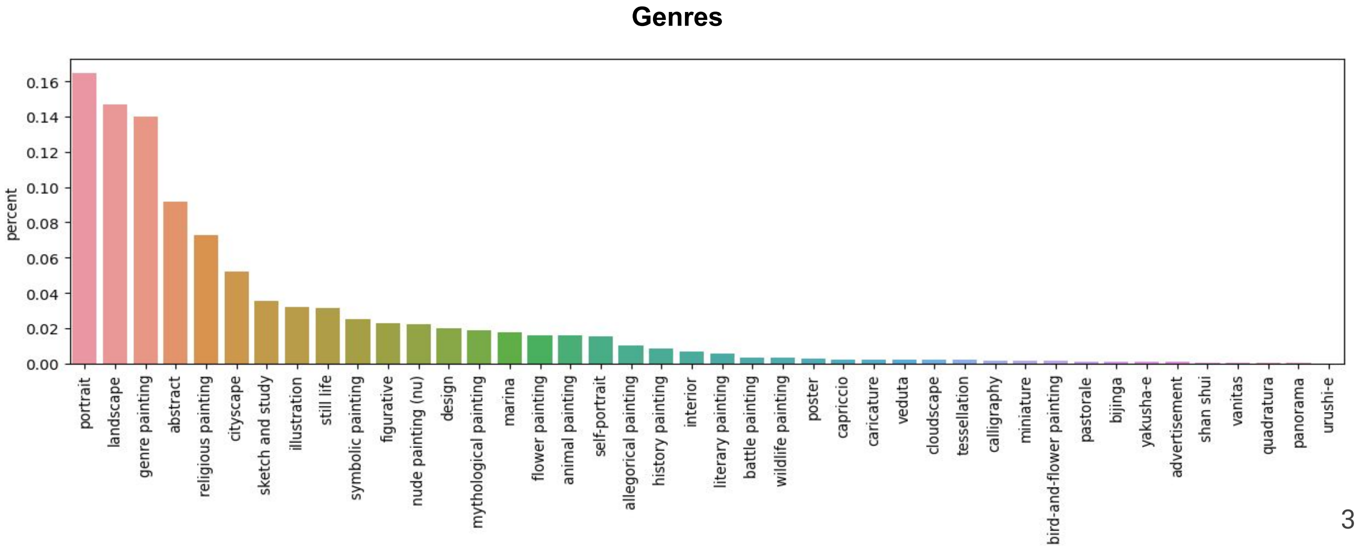
- Classe sbilanciata
- 1254 quadri non hanno genere
- 23 dei generi contengono meno dell'1% del trainset

widths



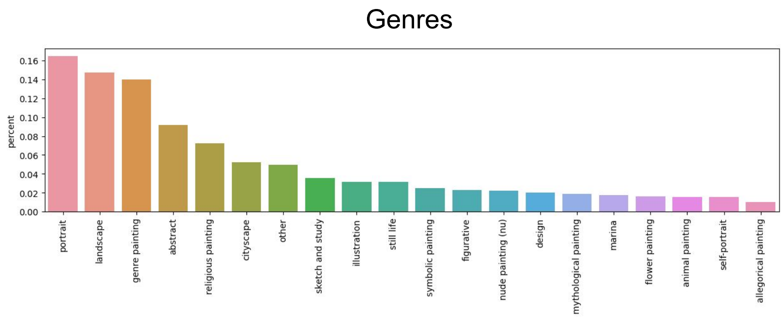
heights

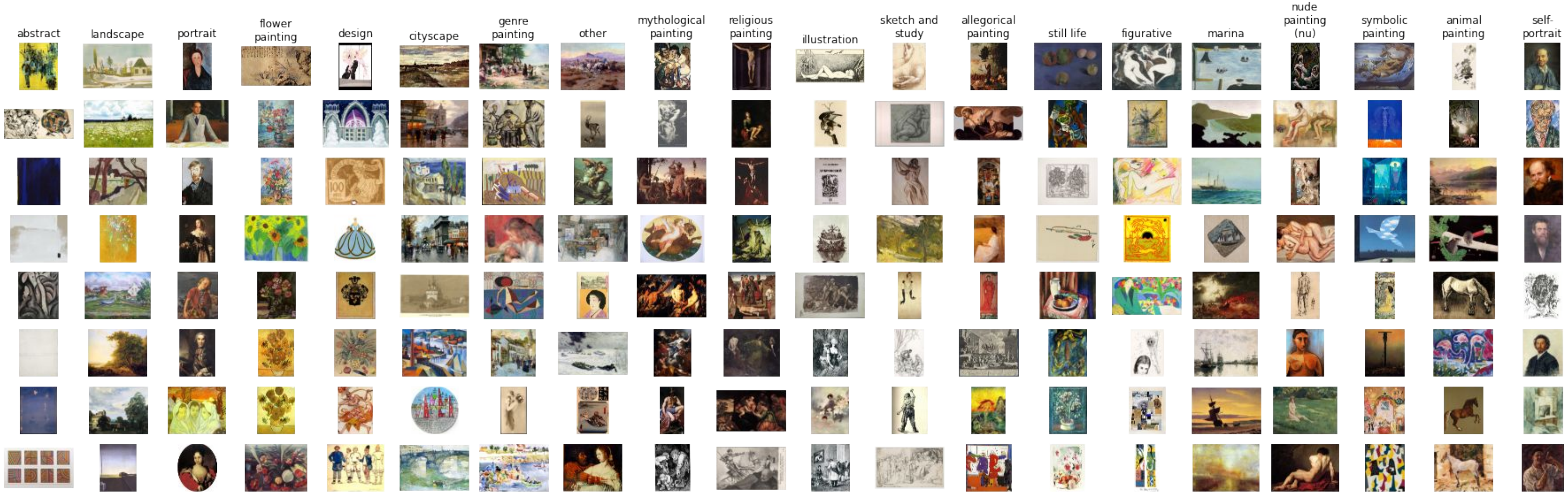




Dataset

- 19 generi + classe “other”
- 101.977 quadri





Processo

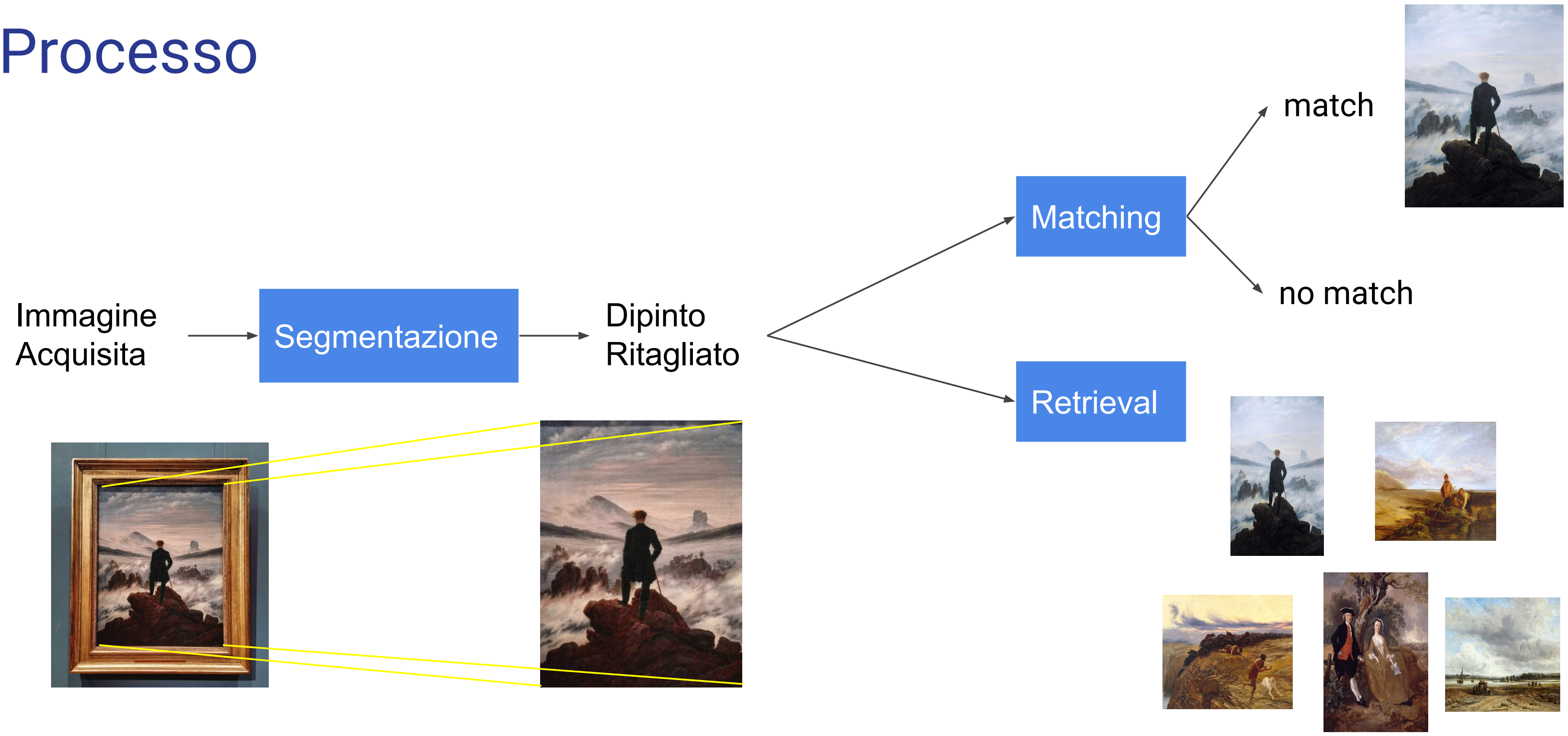


Immagine Acquisita
Segmentazione
Dipinto Ritagliato
Matching
Retrieval
match
no match

Segmentazione - Problema

Alta variabilità del contenuto.

I dipinti spesso contengono o sono all'interno di scene con persone o oggetti di uso comune.

Segmentazione - Problema

- Ombre
- Diverse Cornici (semplici, sottili, …)
- Diverse dimensione

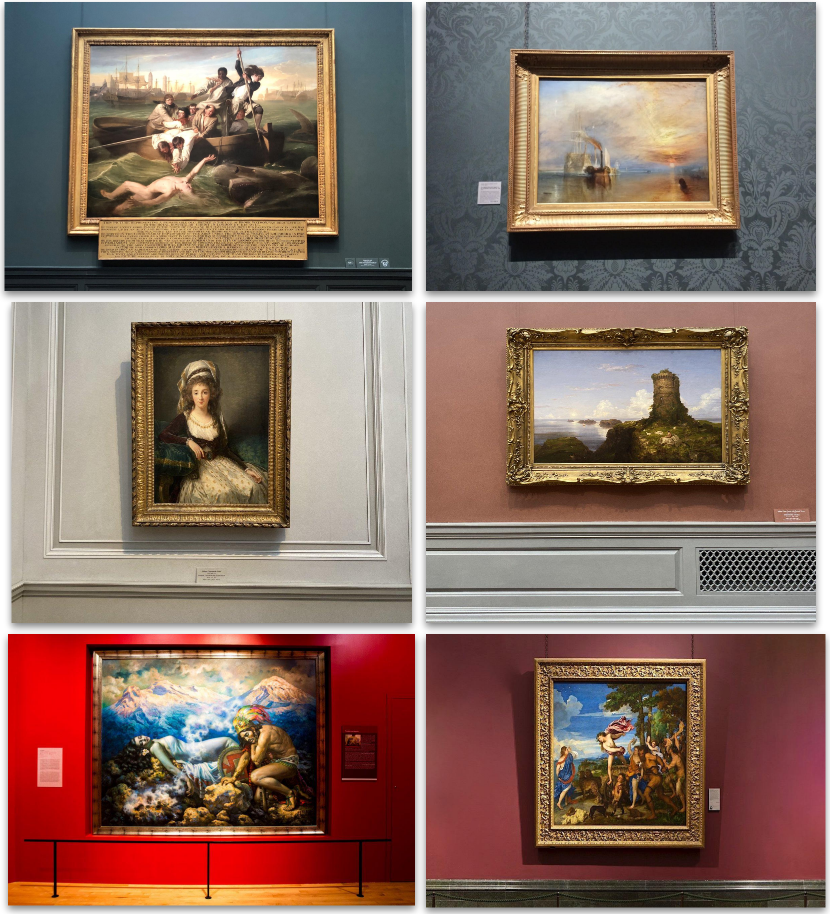
Segmentazione - Dataset

Raccolta:

- Immagini raccolte su internet con risoluzione
  max: 3264x2448, min: 471x567, avg: 1505x1638
- 300 immagini con relative maschere di segmentazioni
  (train: 250, test: 50)
- Alcune keywords (art gallery, museum, paintings, national
  museum, …)

Assunzioni:

- Un solo quadro per immagine sempre presente nell’immagine
- Scene di musei o sfondi abbastanza uniformi
- Risoluzione non troppo bassa e rumore (non eccessivo)
- Diverse condizioni di luce (non eccessive)
- Diverse scale e angolazioni (non eccessive)
- Diversi tipi di quadro principalmente rettangolari
- No occlusioni
- Oggetti o persone non troppo vicine al quadro
- Il quadro deve essere l’oggetto più grande della scena

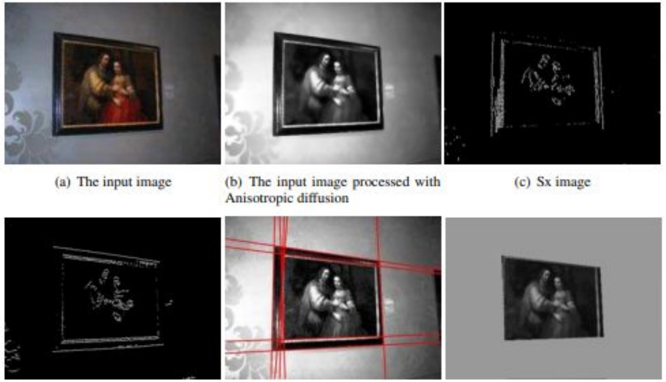


Segmentazione

Un primo possibile approccio come visto nel paper[1] è usare la trasformata di hough.

Contro:

- Nel nostro caso alcune cornici non presentano linee ben definite, difficile trovare le rette di hough.
- Il contenuto dei quadri è molto vario può contenere linee, il che impedisce di segmentare il dipinto come rettangolo più piccolo.
- Senza ulteriori assunzioni (sapere se il quadro è centrato o è l’oggetto più grande della scena, ...) diventa difficile segmentare il dipinto
- Le nostre scene presentano un background più vario



(a) The input image (b) The input image processed with Anisotropic diffusion (c) Sx image

(d) Sy image (e) Lines created using Hough transform (f) Final segmentation

1. HALADOVA, Zuzana. Segmentation and classification of fine art paintings. In: *14th Central European Seminar on Computer Graphics*. 2010. p. 59.

Segmentazione

Segmentazione semantica con u-net

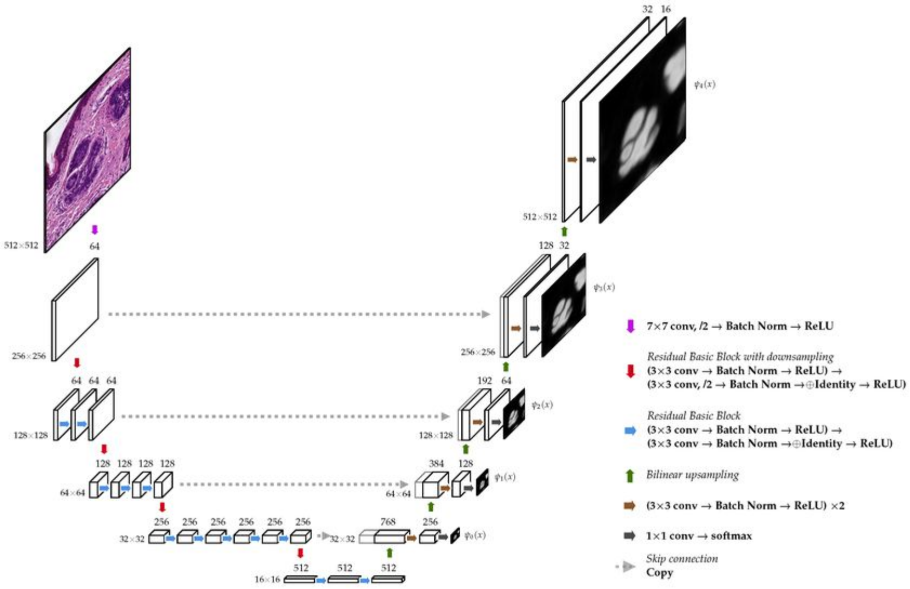
Fine-tuning su imagenet (Train 200, Validation 50, Test 50)

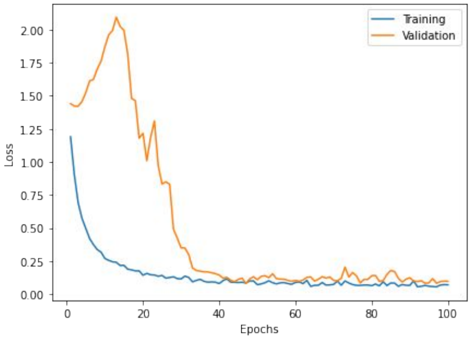
Info Rete:

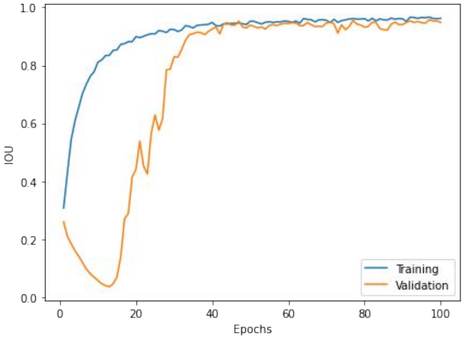
- input: (224, 224, 3),
- backbone: ResNet34
- ottimizzatore Adam (learning rate: 1e-4)
- batch size: 8
- epoche:100
- Binary Cross Entropy + Jaccard Loss

Data Augmentation:

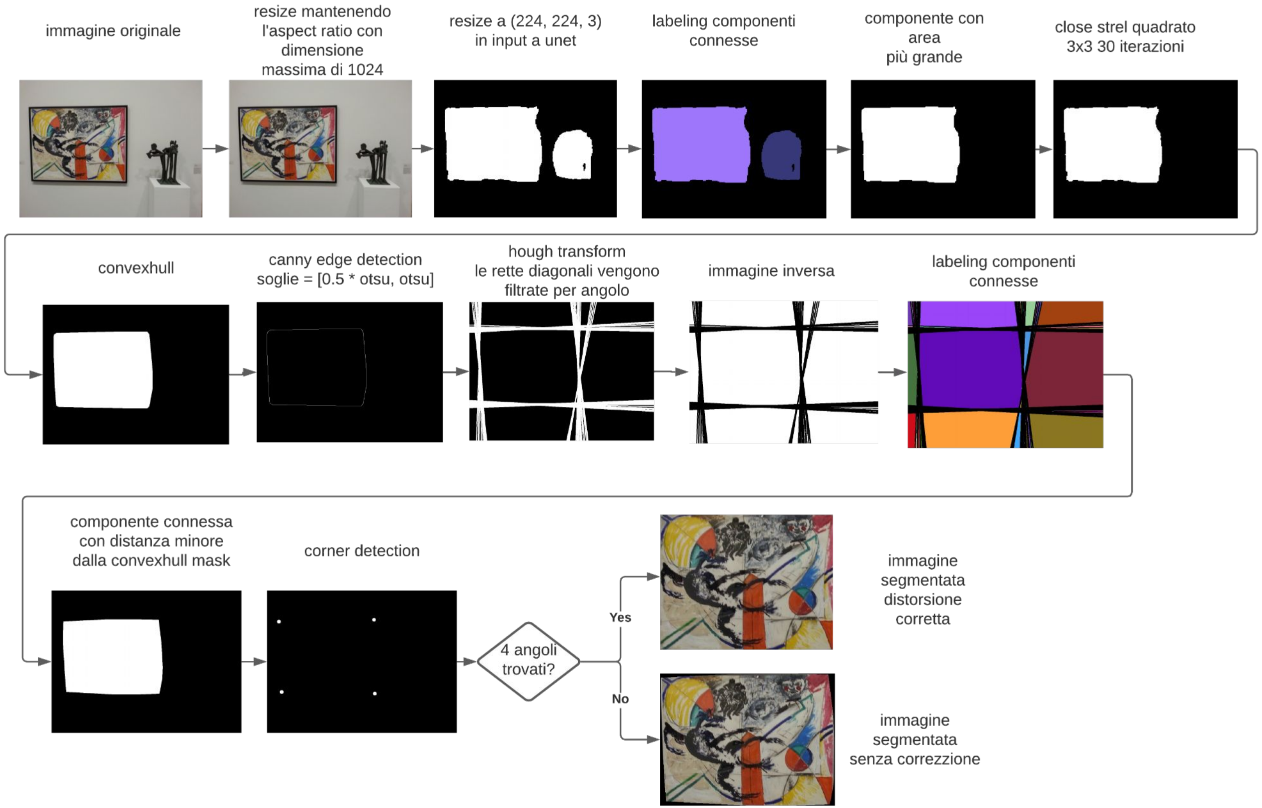
- Rotation Range: 5
- Shear Range: 10
- Zoom Range: [1, 1.2]
- Flip Verticale
- Flip Orizzontale







Pipeline segmentazione



Segmentazione - Risultati

iou_score (unet): 92,22%

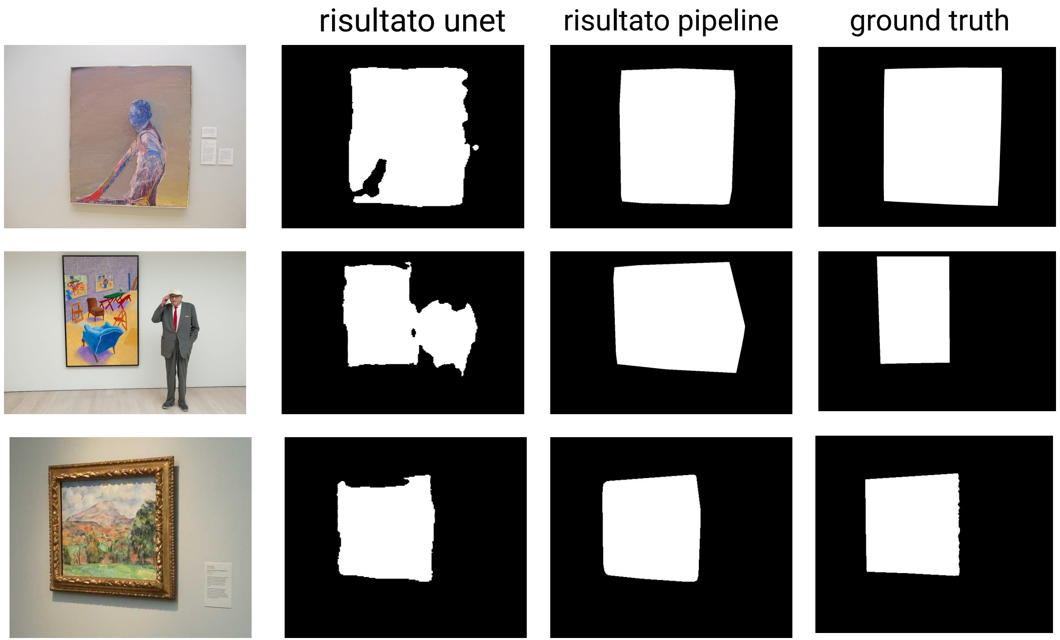
iou_score (pipeline): 91,60%

tempo (unet): 26.94ms(GPU) / 69.24ms(CPU)

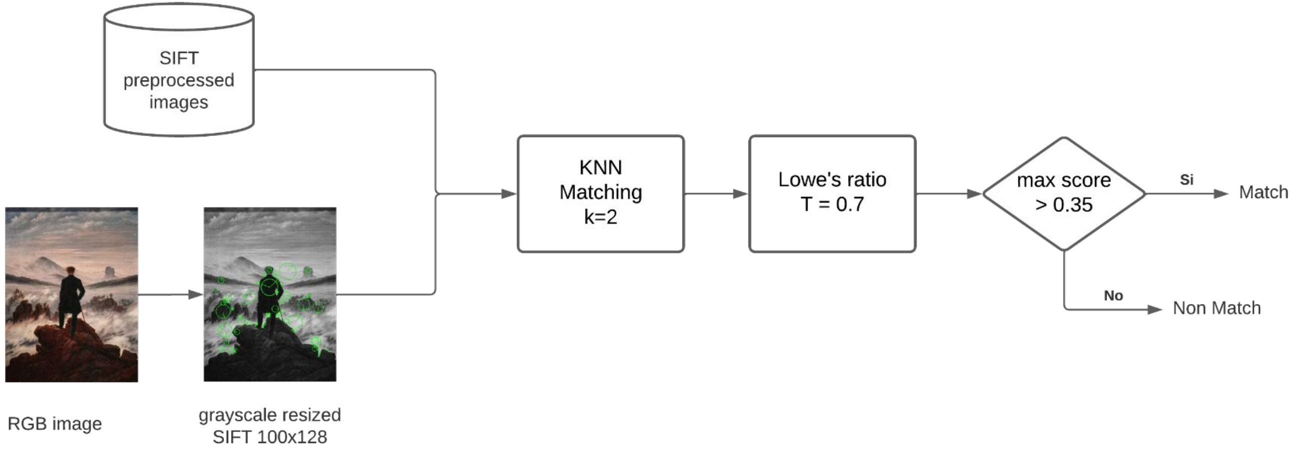
(stessa immagine stessa risoluzione media di 30 volte)

tempo (pipeline): 198.63ms (CPU)

(stessa immagine stessa risoluzione media di 30 volte CPU)



Matching



- image ridimensionata con dimensione massima di 512 mantenendo l'aspect ratio
- vettore con dimensione 100 x 128 (per motivi di risorse computazionali)
- score = |correct match| / |all match|
- soglie trovate empiricamente (slides extra)

Matching

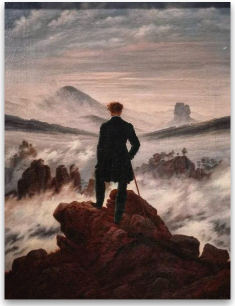
Immagine
nel dataset

score: 1.0
time: 8.51s

score: 1.0
time: 8.08s

Immagine
simile

score: 0.43
time: 8.45s
incorrect match



score: 0.56
time: 8.48s

Immagine
non presente
nel dataset

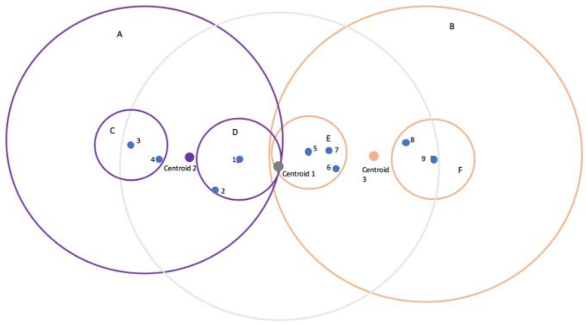
score: 0.26
time: 8.42s

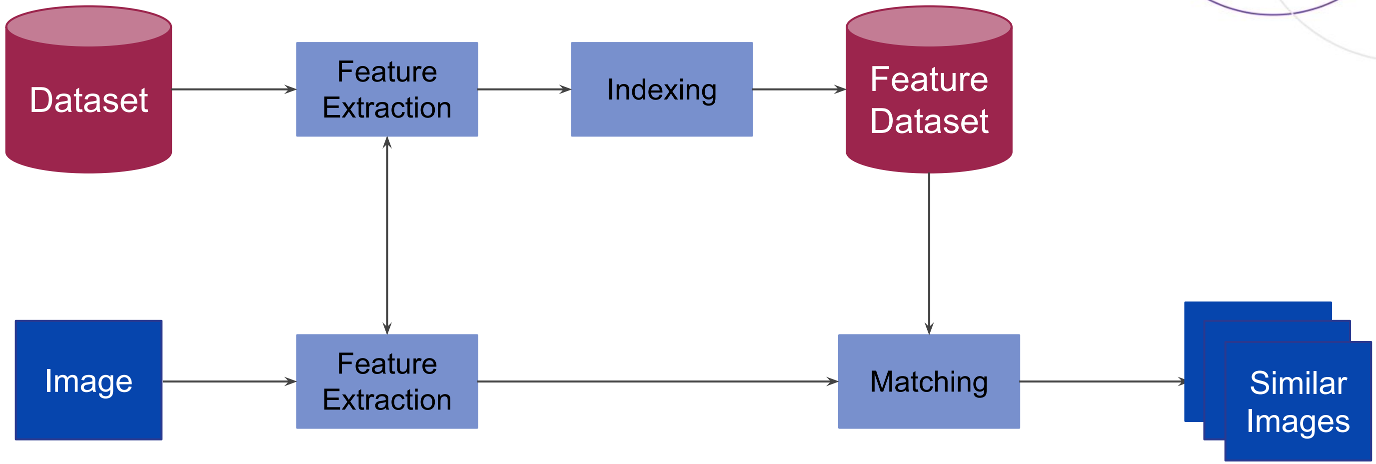
score: 0.2
time: 8.27s

Retrieval

- Handcrafted Features
- Deep Features
- Classificazione

Ball Tree





Retrieval - Handcrafted Features

Resize delle immagini a 512x512

- Istogramma RGB (Locale e Globale)
- Istogramma HSV (Locale e Globale)
- Local binary patterns (LBP)
- Histogram of oriented gradients (HOG)
- HOG + LBP + Istogramma RGB Locale (PCA 90%)

Retrieval - Deep Features

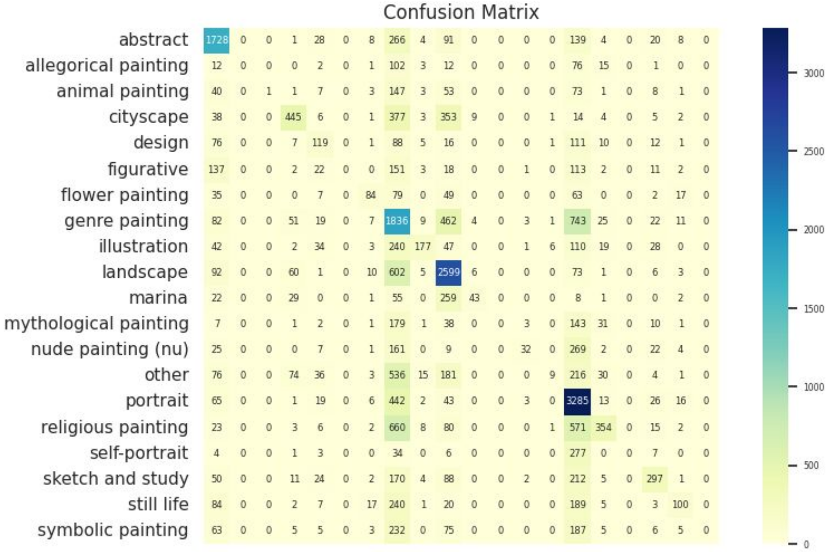
Estrazione delle features da **ResNet50**, dopo fine-tuning.

Abbiamo applicato un taglio all'ultimo layer e aggiunto un layer di pool e un layer fully connected con dropout.

Parametri finali, dopo hyper-parameter optimization, con Hyperband:

- **Neuroni** 1024
- **Dropout** 0.5
- **Epoche** 20

Abbiamo addestrato i nuovi layer arrivando ad una accuracy del 47.36%.



Retrieval - Similarity

La similarity viene calcolata in base alla **distanza** tra le features.

Minore la distanza, maggiore la similarity.

Abbiamo considerato le seguenti distanze:

- Euclidea
- Manhattan
- Chebyshev

Retrieval - Senza Classificazione

Questo metodo è basato solo sulla **similarity**. Vengono considerati rilevanti le k immagini più simili a quella in input.

Pro:

- Ranking e relevance sono fatti in un unico passaggio.

Contro:

- Il numero di elementi recuperati <u>non</u> dipende dall’input.

Retrieval - Con Classificazione

Questo metodo è basato sulla classificazione del genere dell'immagine in input. Vengono poi recuperate le immagini che, nel database, corrispondono allo stesso genere.

Viste le performance questo metodo viene usato solo se la **confidence** sulla predizione è superiore al 40%.

Pro:

- Il numero di elementi recuperati dipende dall'input.
- Se la classificazione è corretta precision e recall sono 1.

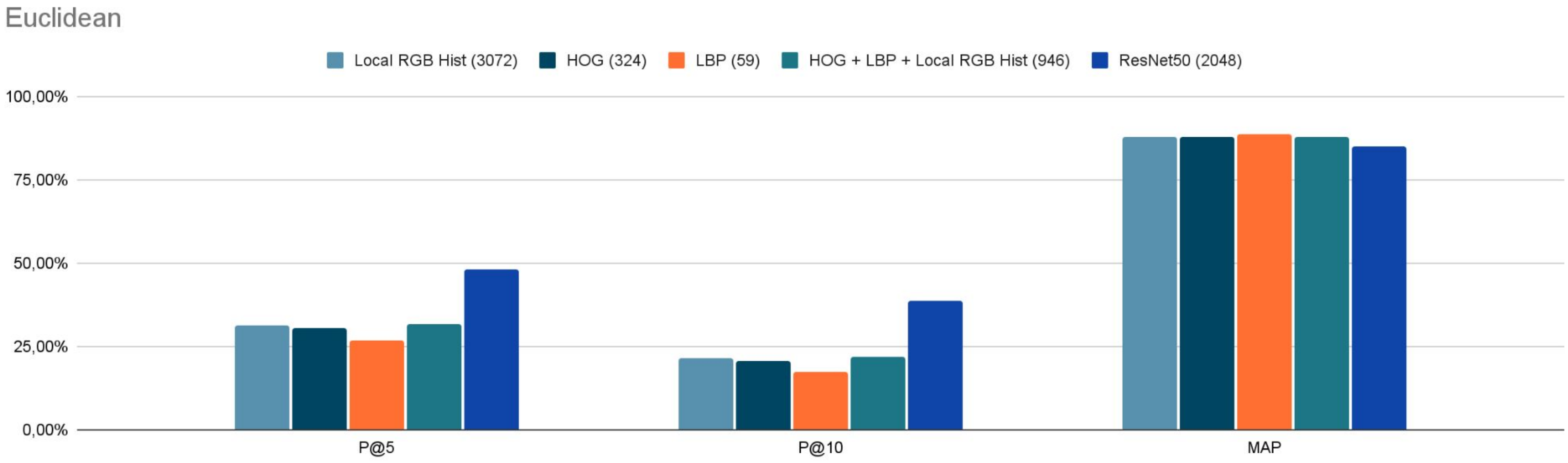
Contro:

- Se la classificazione è sbagliata precision e recall sono 0.

Retrieval - Valutazione

P@5, P@10 e MAP su 200 query per genere sul test set del dataset originale (23817)

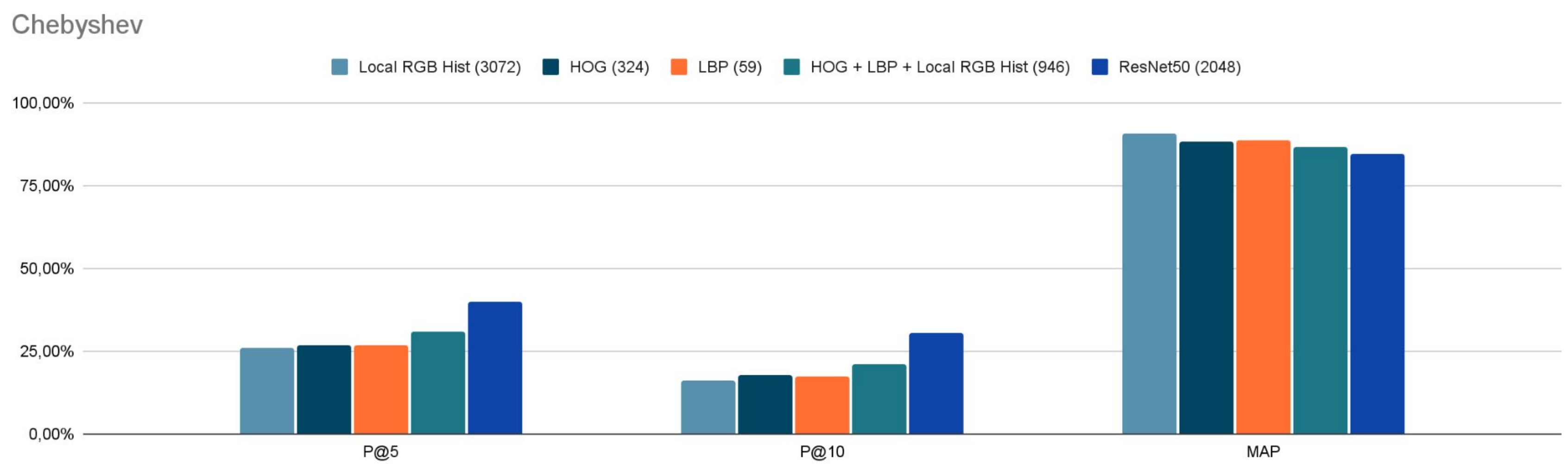
P@5 e P@10 più alte per ResNet50 e HOG + LBP + Local RGB Hist per tutte le similarità



Retrieval - Valutazione

P@5, P@10 simili all'Euclidea per LBP e HOG + LBP + Local RGB Hist

MAP simile per tutte le feature



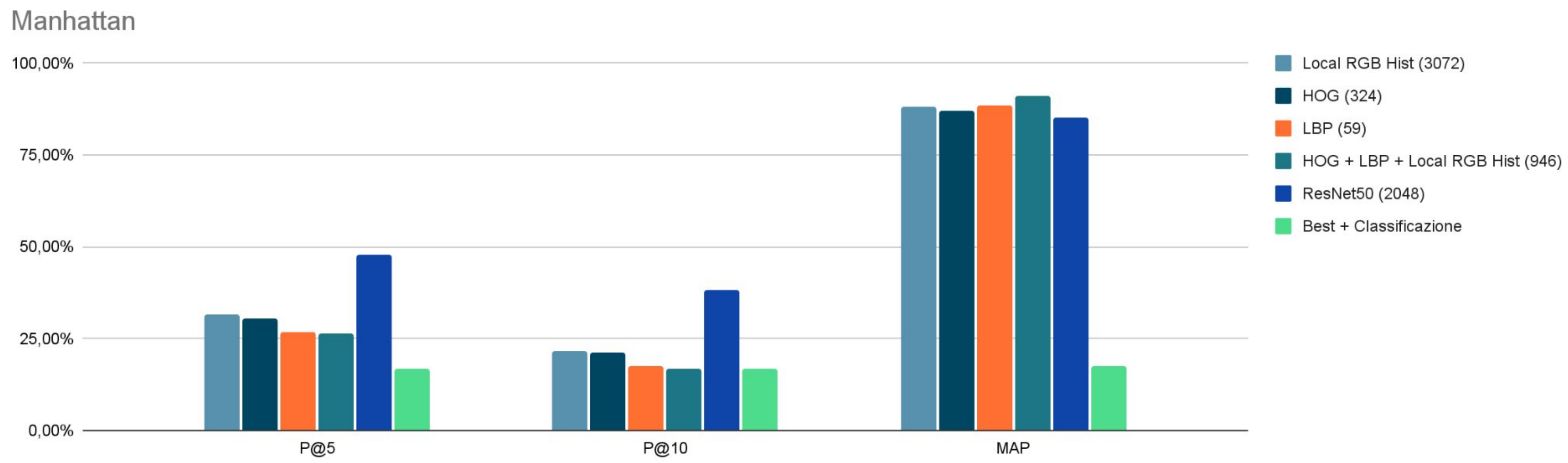
Retrieval - Valutazione

Performance simili a Chebyshev

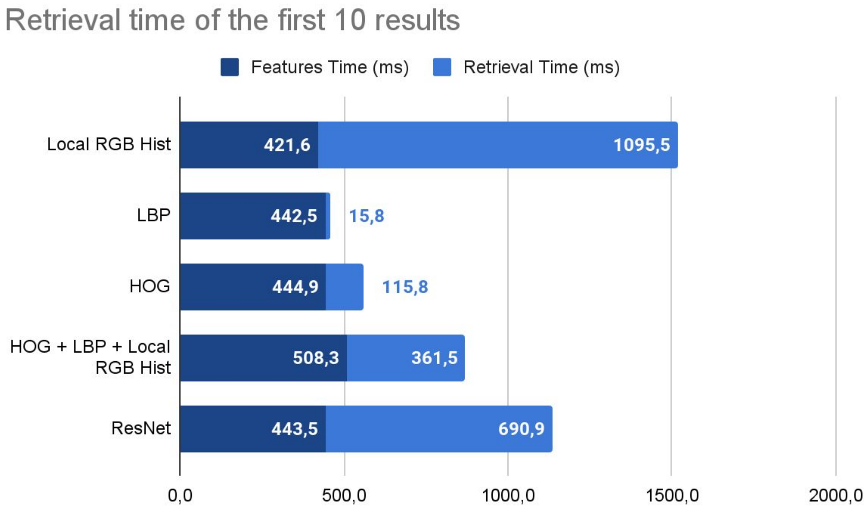
P@5, P@10 più alte per Local RGB Hist e HOG

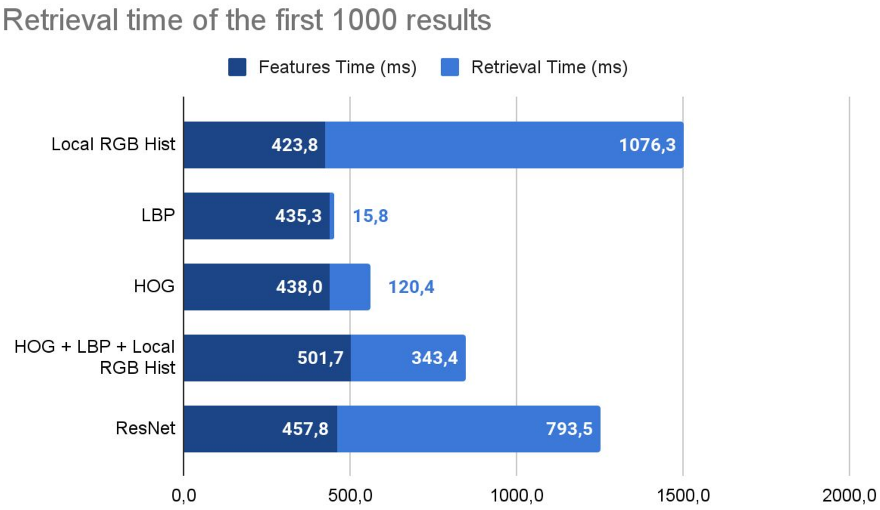
MAP più alta per HOG + LBP + Local RGB Hist

Abbiamo anche considerato la **classificazione** per la feature che ha dato il risultato migliore, ovvero HOG + LBP + Local RGB Hist.



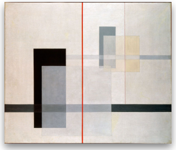
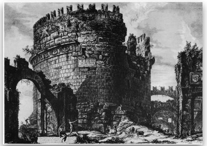
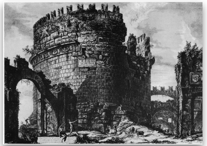
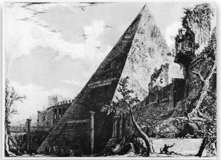
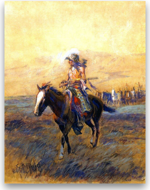
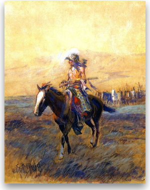
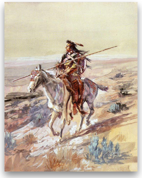
Retrieval - Tempi





Tempi ottenuti su una media di 10 retrieval di immagini diverse

Retrieval - Esempi

	query	results				
resnet50 euclidean	**abstract** 	abstract  0,05	abstract  26,76	abstract  29,92	abstract  29,96	abstract  30,27
LBP chebyshev	**cityscape** 	cityscape  0,0000	cityscape  0,0060	cityscape  0,010	cityscape  0,0102	cityscape  0,0104
HOG manhattan	**genre painting** 	genre painting  0,00	genre painting  7,10	other  8,15	genre painting  8,39	genre painting  8,49

Retrieval - Esempi

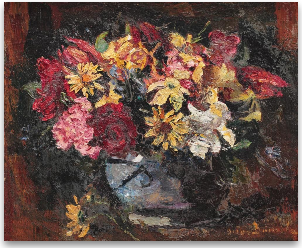
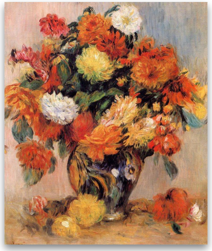
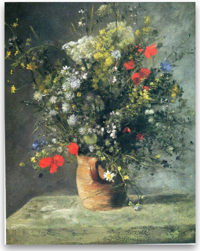
HOG con distanza euclidea

immagine nel dataset

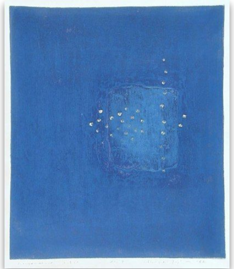
immagine acquisita

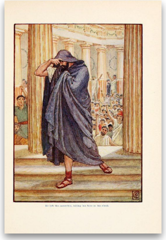
immagine segmentata



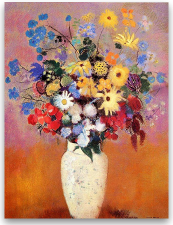







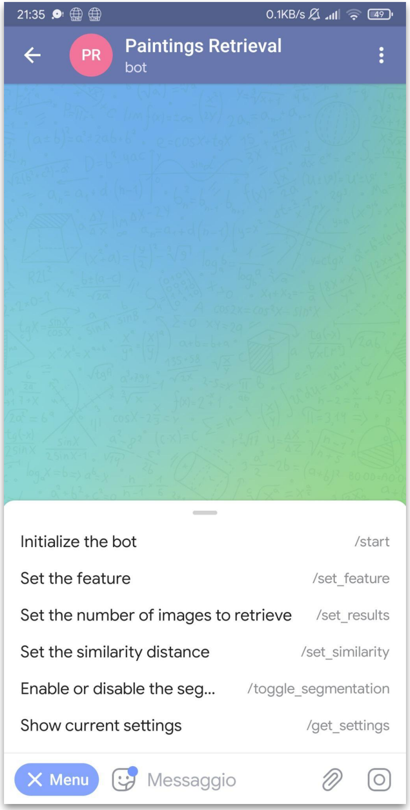



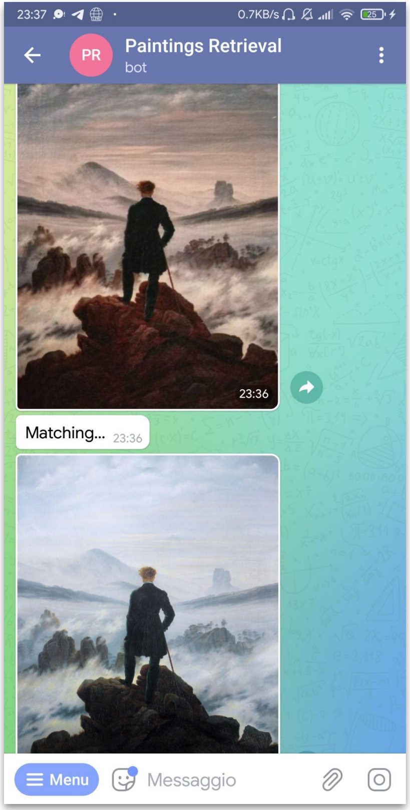


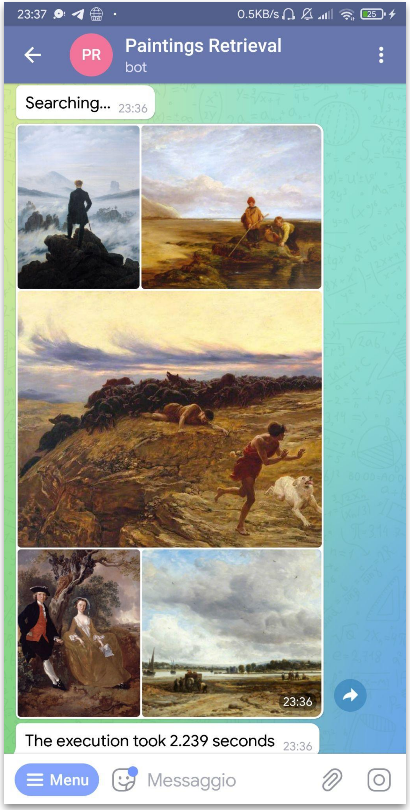


Bot Telegram





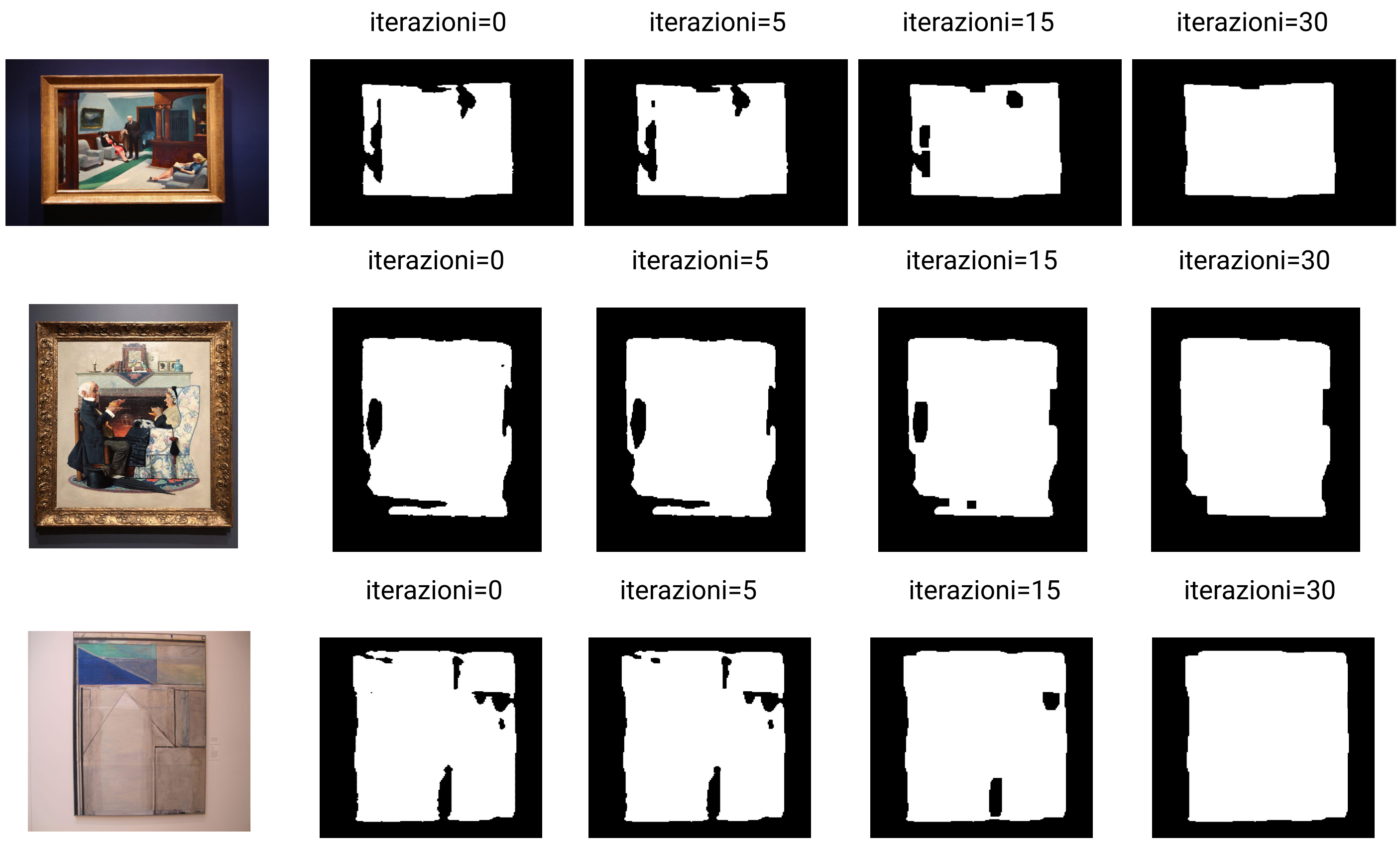


Conclusione e Sviluppi Futuri

- Collezionare più dati per la segmentazione, affrontare il problema delle occlusioni e separare dipinti da persone ed oggetti.
- Generare ground truth per il ranking.
- Migliorare il sistema tramite feedback da parte dell'utente.
- Provare architetture diverse da ResNet50 (VGG16, etc) per la features extraction.
- Migliorare la classificazione attraverso strategie di data augmentation (es. cropping).
- Combinare ResNet50 con altre handcrafted features.

Extra

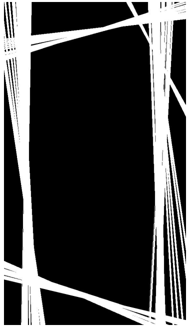
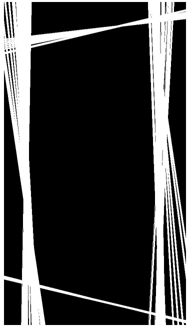
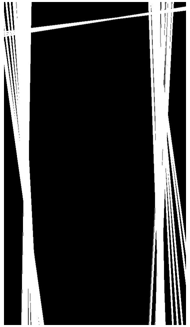
Soglie morfologia close



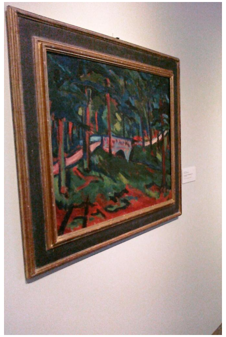
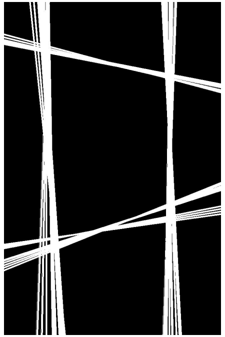
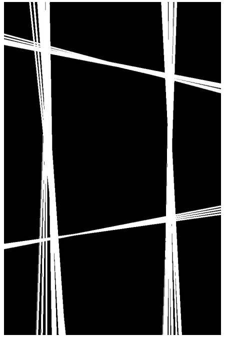
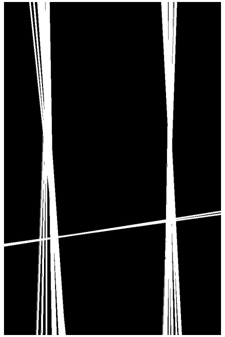
close con kernel quadrato 3x3 al variare del numero di iterazioni

Soglie hough angolo

T=30, K=15 | T=30, K=30 | T=30, K=35

T=30, K=15 | T=30, K=30 | T=30, K=35

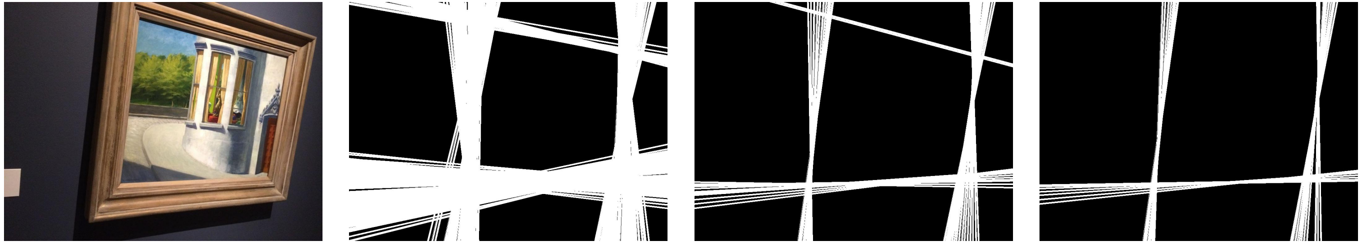





filtro le rette per angolo
45-K <= mod(angolo, 90) <= 45+K

con K = 15 - 0/40 errate
con K = 30 - 0/40 errate
con K = 35 - 3/40 errate

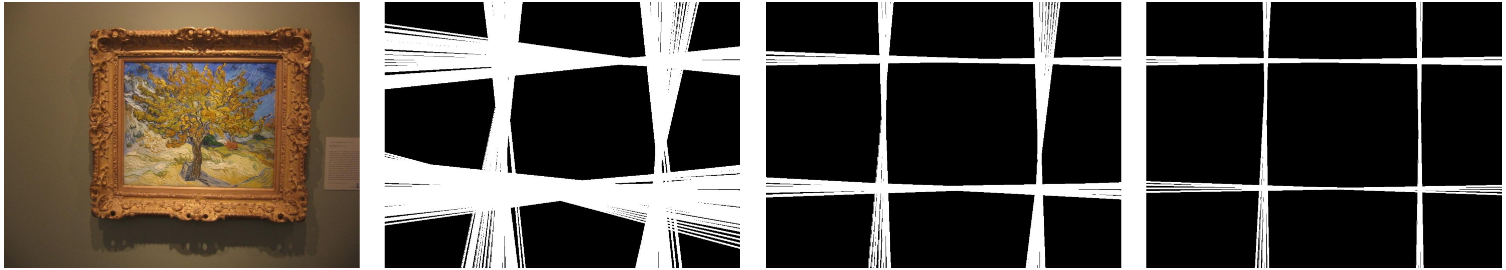
Soglie hough threshold

T=10, K=30 T=30, K=30 T=50, K=30



con T = 10 - 0/40 errate
con T = 30 - 0/40 errate
con T = 50 - 1/40 errate

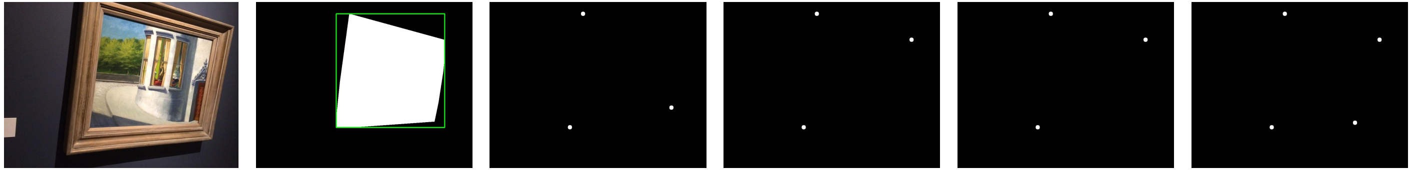
T=10, K=30 T=30, K=30 T=50, K=30



Soglie corner detection (min distance)

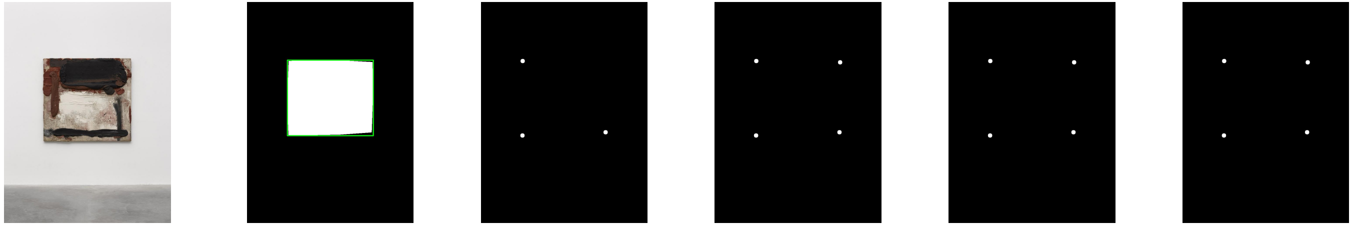
min_dist = min(h, w) * k

k = 0.95 k = 0.9 k = 0.8 k = 0.75



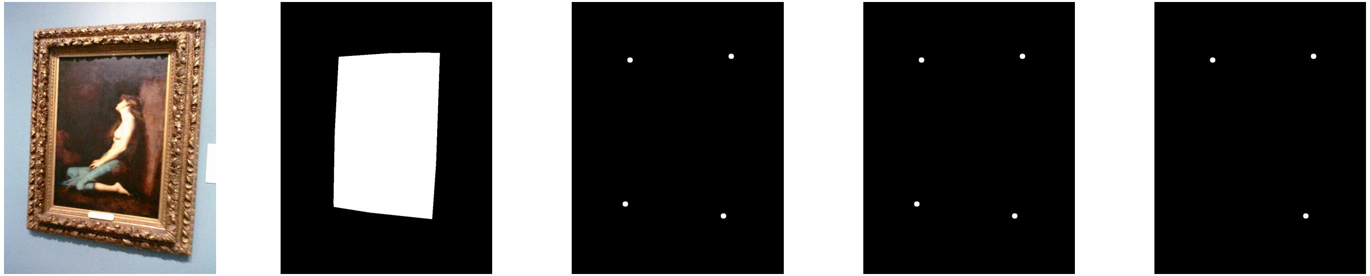
con k = 0.9 - 5/40 errate
con k = 0.8 - 2/40 errate
con k = 0.8 - 1/40 errate
con k = 0.75 - 0/40 errate

k = 0.95 k = 0.9 k = 0.8 k = 0.75



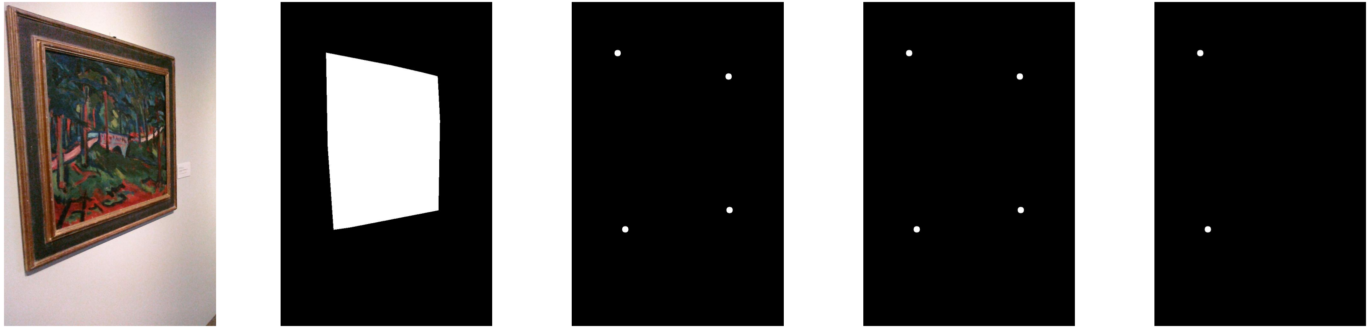
Soglie corner detection (threshold)

T=0.01 T=0.1 T=0.5



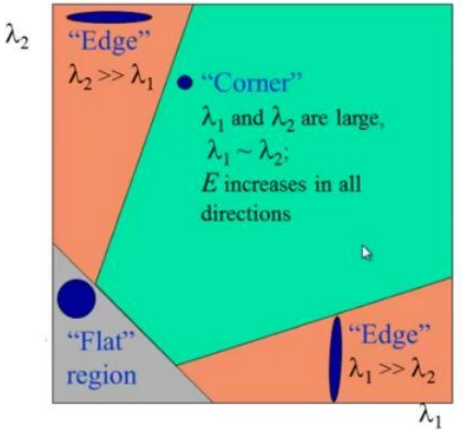
con T <= 0.1 - 0/40 errate
con T= 0.5 - 4/40 errate

T=0.01 T=0.1 T=0.5



Harris Corner Detection

- Immagine grayscale
- Filtro gaussiano per il rumore
- Approssimazione dei gradienti con sobel
- Per ogni pixel si calcola una funzione di risposta R in un intorno locale 3x3



funzione di traslazione dell'intorno somma degli scarti quadratici

$$SSD(\Delta x, \Delta y) = \sum_{(x_k, y_k) \in W} (I(x_k, y_k) - I(x_k + \Delta x, y_k + \Delta y))^2$$

$$SSD(\Delta x, \Delta y) \approx (\Delta x \quad \Delta y) M \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

$$M = \sum_{(x,y) \in W} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \sum_{(x,y)\in W} I_x^2 & \sum_{(x,y)\in W} I_x I_y \\ \sum_{(x,y)\in W} I_x I_y & \sum_{(x,y)\in W} I_y^2 \end{bmatrix}$$

$$R = \det(M) - k(\mathrm{tr}(M))^2 = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2$$

where k is an empirically determined constant; $k \in [0.04, 0.06]$

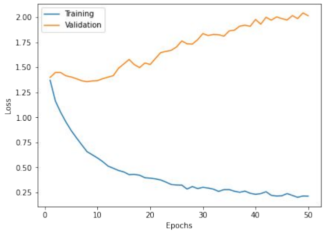
Gli autovettori del tensore di struttura determinano le direzioni di massima e minima variazione della somma dei quadrati residui, e i corrispondenti autovalori λ_1 e λ_2 quantificano la variazione lungo le rispettive direzioni

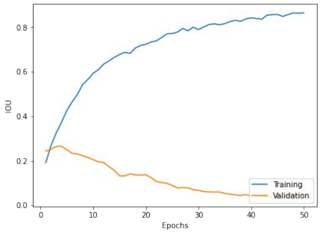
Metodo Shi-tomasi:

$$R = \min(\lambda_1, \lambda_2)$$

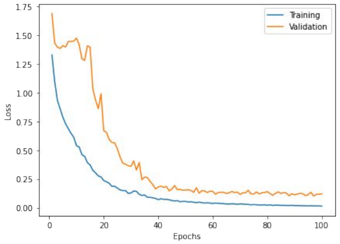
Addestramento Unet

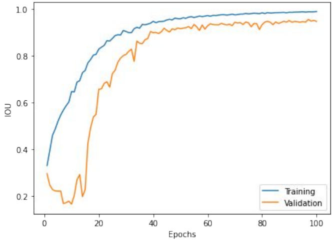
1) batch: 32, epoche: 50
test_loss: 2.101,
test_iou_score: 0.0317





3) batch: 8, epoche: 100,
test_loss: 0.23,
test_iou_score: 0.895

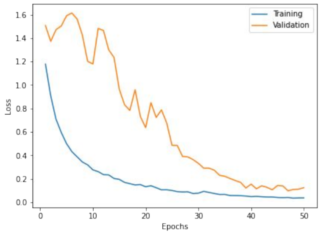


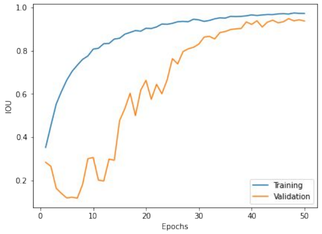


per tutte le prove
ottimizzatore: adam
learning rate: 1e-4

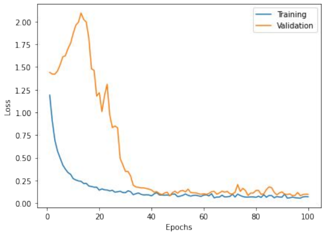
grafici con curve loss train e validation e iou_score

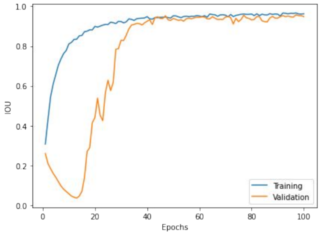
2) batch: 8, epoche: 50,
test_loss: 0.25,
test_iou_score: 0.88





4) batch: 8, epoche: 100,
test_loss:0.1518,
iou_score: 0.9222





SIFT (Scale Invariant Feature Transform)

un approccio piramidale con diverse scale viene utilizzato per essere invariante a scala

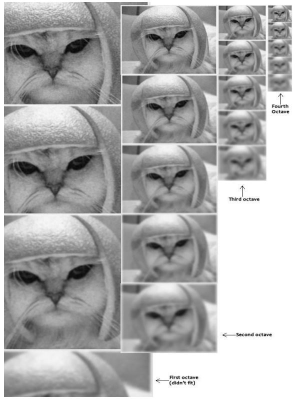
stima gradienti come differenza di gaussiane

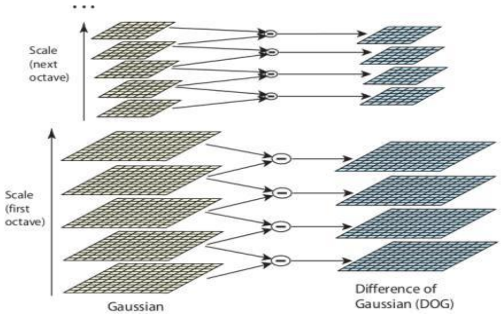
i keypoints vengono trovati analizzando un intorno di 26 pixel nella scala precedente e successiva.

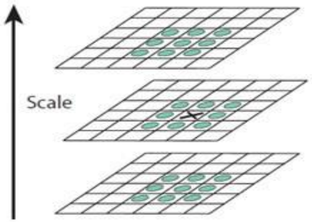
alcuni keypoints vengono scartati se minori di una certa soglia.

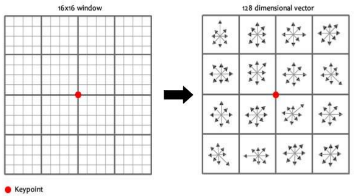
per ogni keypoints
viene calcolato un istogramma di orientazioni
(36 bin 10 orientazioni ciascuno)

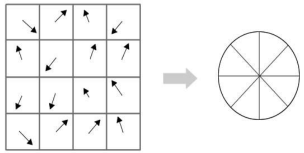
in finestre 16x16 che poi vengono ridotte a 4x4
ottenendo un vettore 4x4x8=128
che viene poi normalizzato









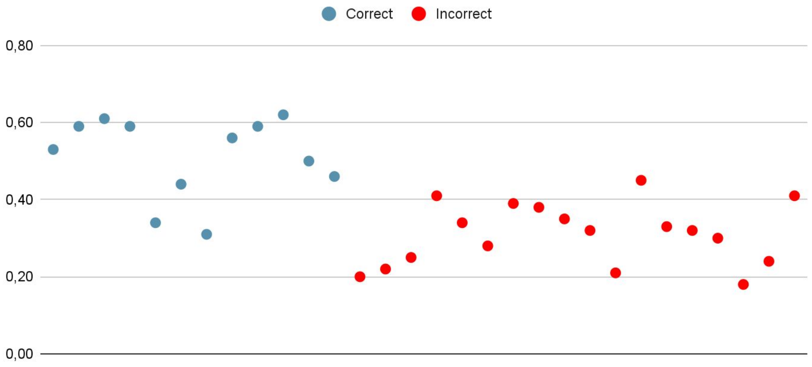


rotation invariance: l'orientazione del keypoint viene sottratta ad ogni bin dell'istogramma.

illumination invariance:
ogni valore sopra 0.2 viene fissato a 0.2

Matching - Lowe's Ratio

Lowe's ratio: 0.75



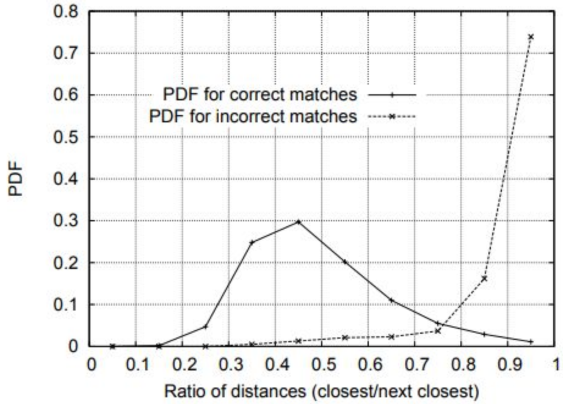
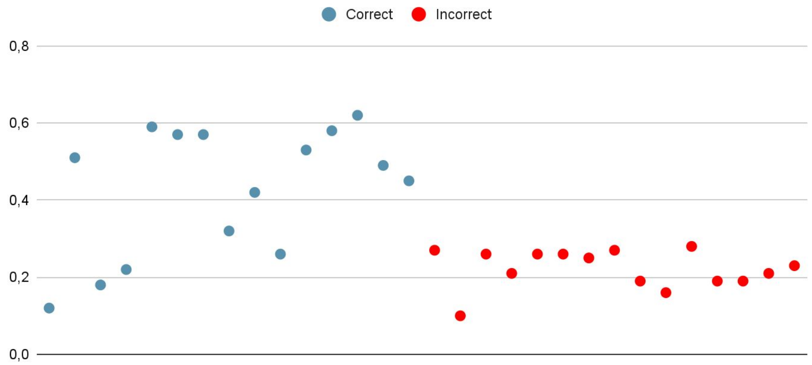


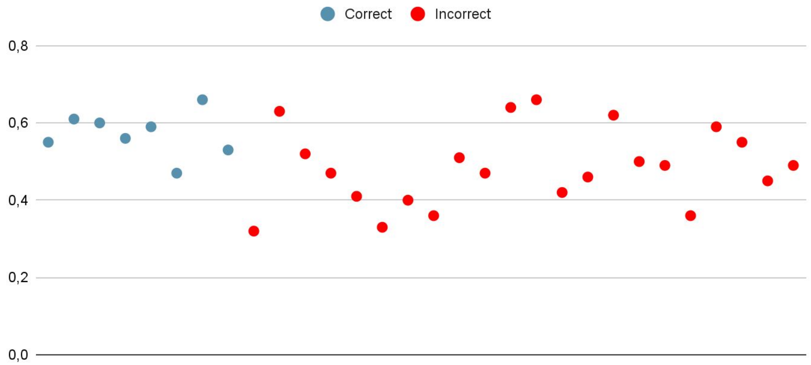
Figure 11: The probability that a match is correct can be determined by taking the ratio of distance from the closest neighbor to the distance of the second closest. Using a database of 40,000 keypoints, the solid line shows the PDF of this ratio for correct matches, while the dotted line is for matches that were incorrect.

LOWE, David G. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 2004, 60.2: 91-110.

Lowe's ratio: 0.7



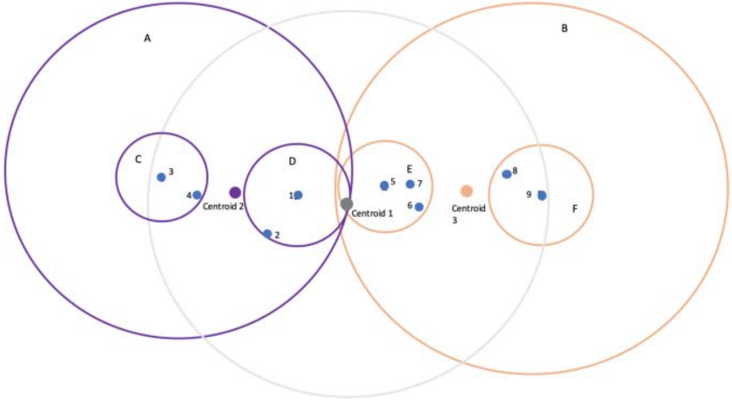
Lowe's ratio: 0.8

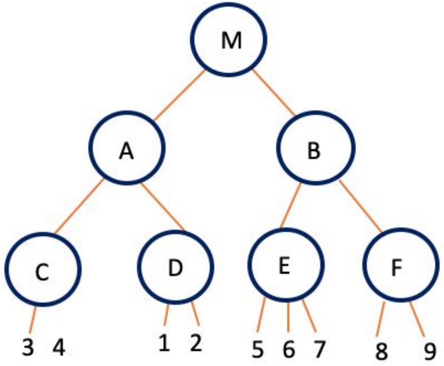


Prove su 30 immagini

Ball Tree

Struttura ad albero che organizza e dispone i punti in uno spazio metrico. Ogni nodo definisce un ipersfera (ball) che contiene un sottoinsieme di punti.





https://jakevdp.github.io/blog/2013/04/29/benchmarking-nearest-neighbor-searches-in-python/

test di 5000 dati, 50 query per genere

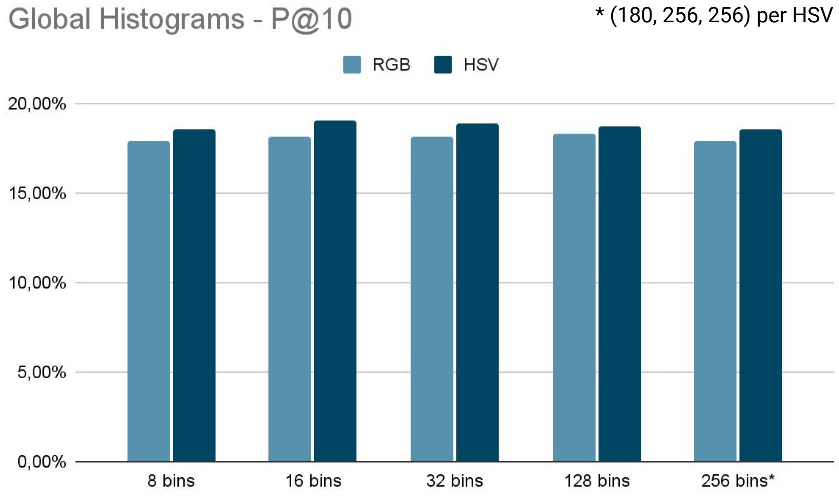
Handcrafted Features - Parametri

Istogramma RGB: 128 bins, P@10=18,3%

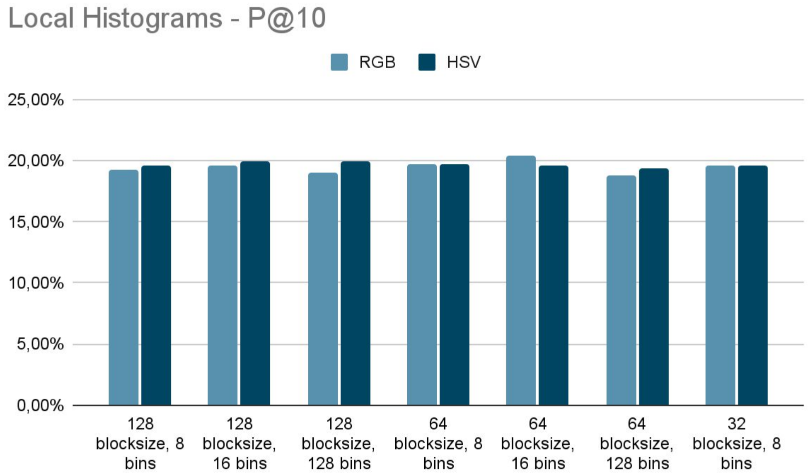
Istogramma HSV: 16 bins, P@10=19,0%

Locale RGB - 64 blocksize, 16 bins, P@10=19,67%

Locale HSV - 128 blocksize, 16 bins, P@10=19,97%



dimensioni: 24 48 96 384 768/692*



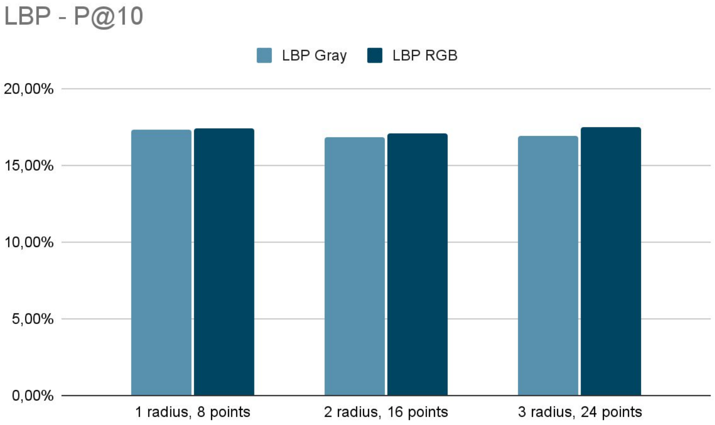
dimensioni: 384 768 6144 1536 3072 24576 6144

Handcrafted Features - Parametri

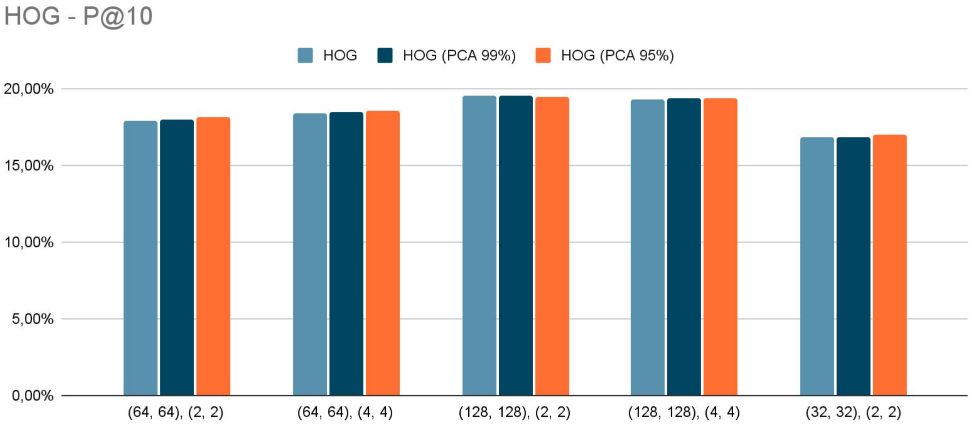
LBP: RGB, raggio 1, 8 punti, P@10: 17,41%

HOG: pixels per cell (128, 128), cells per block (2, 2), P@10: 19,50%

Con la PCA le performance migliorano leggermente

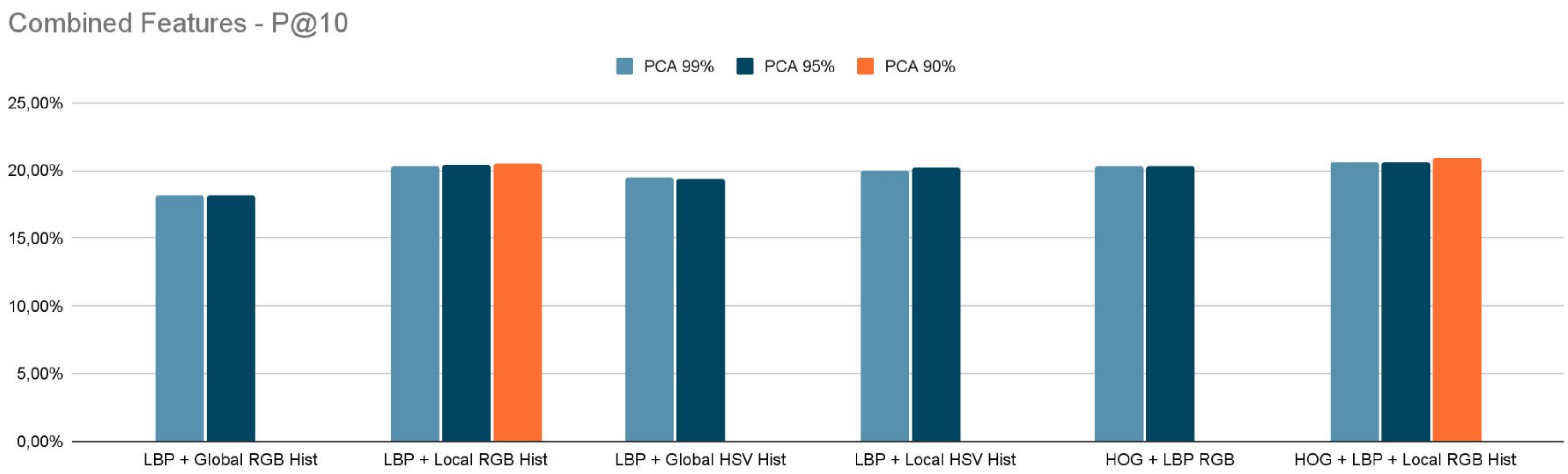


	1 radius, 8 points	2 radius, 16 points	3 radius, 24 points
Gray:	59	243	555
RGB:	177	729	1665



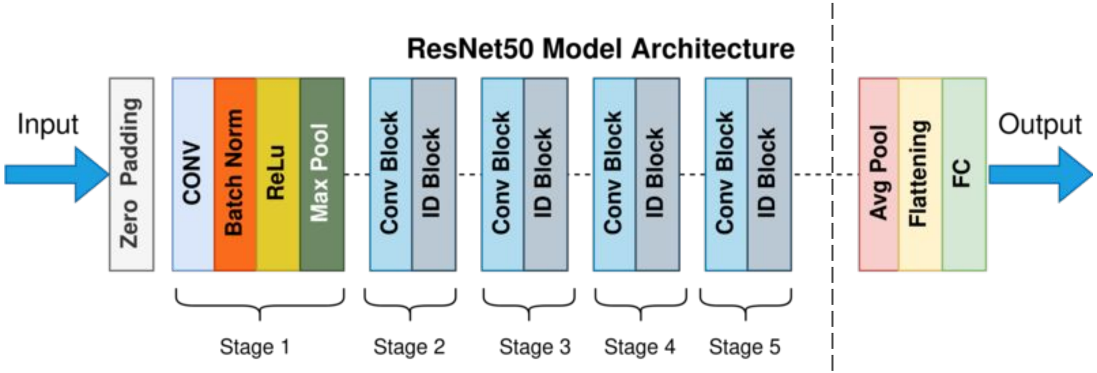
	(64, 64), (2, 2)	(64, 64), (4, 4)	(128, 128), (2, 2)	(128, 128), (4, 4)	(32, 32), (2, 2)
	1764	3600	324	144	8100
PCA 99%:	716	493	159	104	2225
PCA 95%:	322	248	73	61	1178

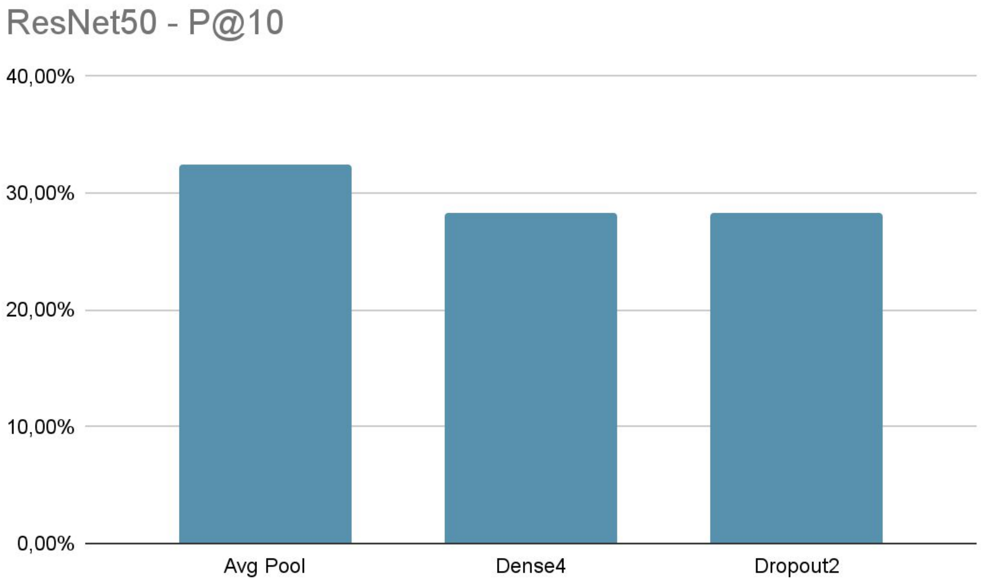
Handcrafted Features - Combinazioni



Dimensioni:	443	3131	107	827	501	3455
PCA 99%:	85	1990	39	521	157	2059
PCA 95%:	37	1141	28	325	69	1179
PCA 90%:		732				758

Deep Features





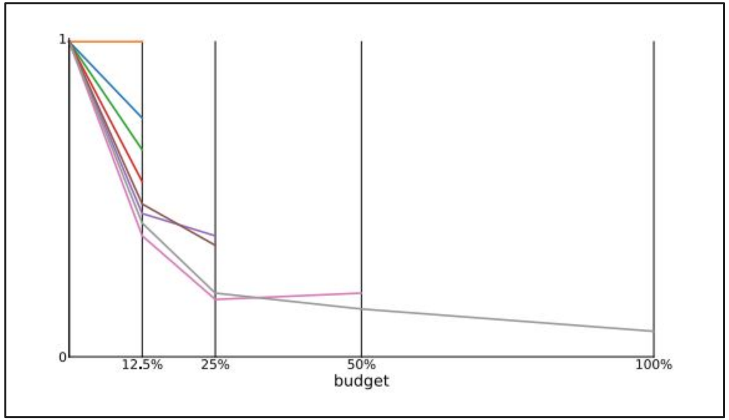
Dimensioni Features:

- **Avg_pool (flatted)** 2048
- **Dense4** 1024
- **Dropout2** 1024

Hyperband

Hyperband extends the **Successive Halving** algorithm.

The idea behind the original Successive Halving algorithm its about uniformly allocate a budget to a set of hyper-parameter configurations, evaluate the performance of all configurations, throw out the worst half, and repeat until one configuration remains.



It requires the **number of configurations** *n* and some finite **budget** *B* as inputs to the algorithm.

Than *B/n* resources are allocated on average across the configurations.

Hyperband

However, in many cases it's not known if it's better to consider many configurations or not.

It runs several Successive Halving runs with different budgets and number of configurations, to find the best set.

It begins with the maximum exploration to ends up with a classical random search, in which every configuration is allocated with R resources.

Algorithm 1: HYPERBAND algorithm for hyperparameter optimization.

input : R, η (default $\eta = 3$)
initialization : $s_{\max} = \lfloor \log_\eta(R) \rfloor$, $B = (s_{\max} + 1)R$
for $s \in \{s_{\max}, s_{\max} - 1, \ldots, 0\}$ **do**
  $n = \lceil \frac{B}{R} \frac{\eta^s}{(s+1)} \rceil$, $r = R\eta^{-s}$
  // begin SUCCESSIVEHALVING with (n, r) inner loop
  $T =$ get_hyperparameter_configuration(n)
  for $i \in \{0, \ldots, s\}$ **do**
    $n_i = \lfloor n\eta^{-i} \rfloor$
    $r_i = r\eta^i$
    $L = \{$run_then_return_val_loss$(t, r_i) : t \in T\}$
    $T =$ top_k$(T, L, \lfloor n_i/\eta \rfloor)$
  end
end
return *Configuration with the smallest intermediate loss seen so far.*

L. Li, K. Jamieson, G. De Salvo, R. A. Talwalkar, and A. Hyperband, “A novel bandit-based approach to hyperparameter optimization,” Computer Vision and Pattern Recognition, arXiv: 1603.0656, 2016