Progetto grayscale2rgb

Introduzione

Ogni pattern del dataset viene fornito come 16 immagini in scala di grigio ritraenti lo stesso oggetto illuminato da 16 angolazioni diverse. Scopo di questo paper è quello di proporre dei metodi per trasformare le 16 immagini di ogni pattern in una singola immagine RGB, così da poter utilizzare classificatori preallenati (es Alexnet).

Le immagini originali (in scala di grigi) sono state scattate con una luce bianca, dunque, l'idea sulla quale si basano i metodi proposti è di simulare che ogni luce associata ad un'angolazione sia di un colore diverso. Abbiamo realizzato un gruppo di metodi che associa ai fasci luminosi solo i colori rosso (r), verde (g) e blu (b) e un metodo che sfrutta il piano colore HSV (Hue, Saturation, Value of brigthness).

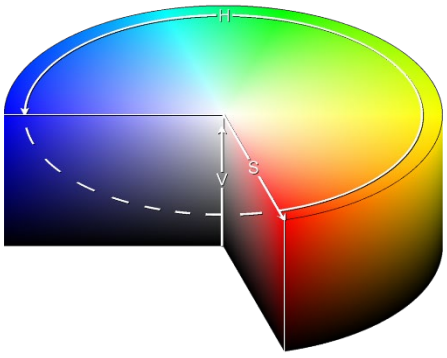
Abbiamo inoltre individuato un potenziale bias: alcune classi di immagini sono diversificate anche dall'angolo di illuminazione iniziale, pertanto, si è deciso di realizzare metodi che randomizzano l'immagine di partenza, pur mantenendo l'ordine sequenziale delle stesse. Per completezza abbiamo poi testato il classificatore sia su dataset creati in questo modo, sia senza questo accorgimento.

Descrizione dei metodi

Metodo HSV

La codifica per i colori HSV utilizza tre dimensioni:

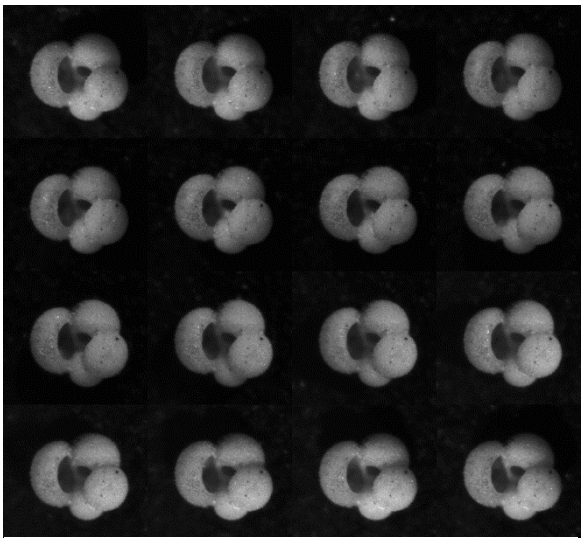
- Hue (H): misura la tonalità su un piano parallelo all'asse verticale V. A 0° si trova il rosso, a 120° il verde e a 240° il blu (in Matlab il valore è mappato in [0,1]).
- Saturation (S): valore in [0,1], 0 rappresenta il centro del cerchio, 1 la circonferenza
- Value of brigthness (V), valore in [0,1], misura la luminosità del colore.



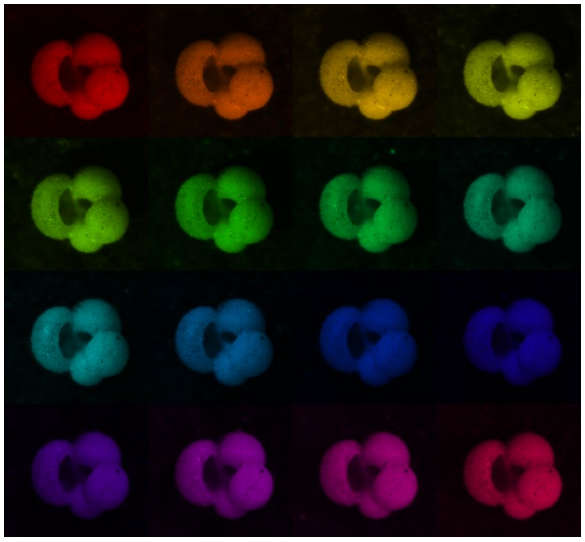
Risulta immediato, pertanto, associare l'angolo di illuminazione di ogni immagine di partenza ad un valore di Hue con il valore del pixel alla luminosità V e settando la saturazione a 1. Questo porta ad avere immagini illuminate da angolazioni adiacenti colorate con tonalità simili: in questo modo immagini contenenti informazioni

potenzialmente ridondanti (è ragionevole pensare che angolazioni di illuminazione limitrofe portino informazioni simili riguardo alla forma dell'oggetto) risultano anche di colori simili.

Inoltre, associando un Hue casuale ad ogni immagine si andrebbe a sovrapporre, nella fase di costruzione dell'immagine RGB, colori diversi nella medesima porzione della foto, ottenendo dunque più bianco.



16 immagini originali



16 immagini colorate su piano HSV

Si procede convertendo i valori HSV delle 16 immagini in valori RGB, sommando i quadrati del valore di ogni pixel in ognuno dei tre piani colore e riscalando il risultato nel range [0,1]; in questo modo si ottiene un aumento del contrasto. È così ottenuto un primo dataset di allenamento per il classificatore (HSV)

Sulle immagini ottenute si applicano metodi di post processing nell'ordine: imlocalbrighten(…), il quale aumenta la luminosità delle porzioni di immagini troppo scure, e imreducehaze(…) il quale aumenta saturazione e vividezza dell'immagine.
Così viene composto un secondo dataset (HSVPP).

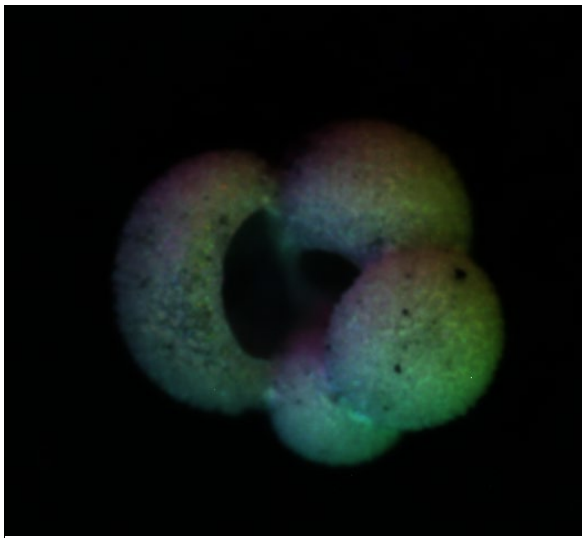


Immagine risultato di hsv

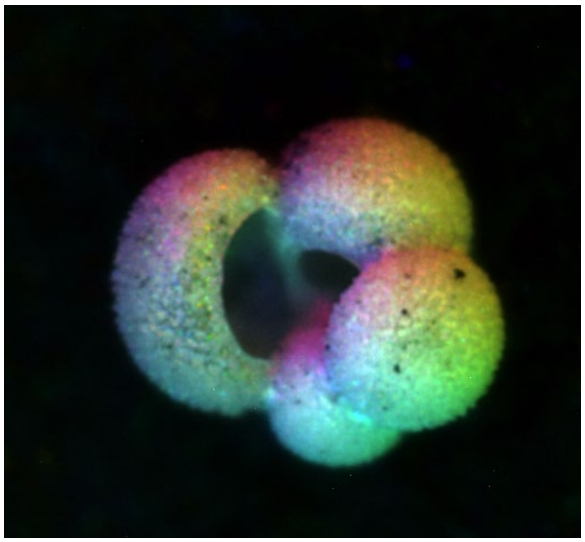


Immagine risultato di HSVPP

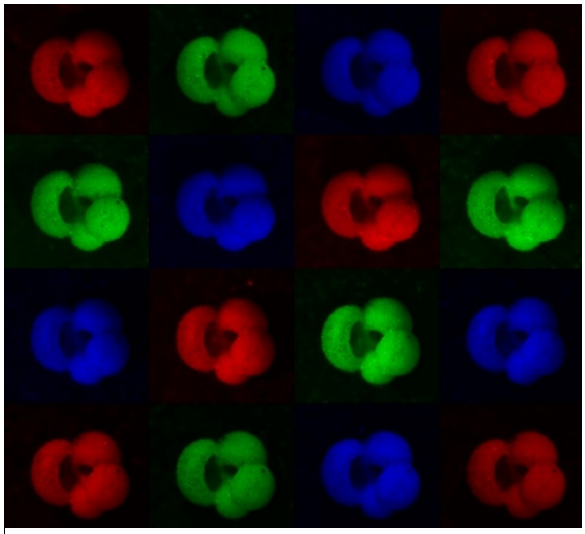
Pseudo codice

```
for i = randomStart([1:16])
   img = readimage(i)

   hsv(:,:,1) = 16 * (i-1);
   hsv(:,:,2) = ones();
   hsv(:,:,3) = img;

   imgOut = imgOut + hsv2rgb(hsv).^2
end

imgOut = rescale(imgOut,0,1);
saveimg(imgO); %Salvataggio immagine HSV

imgOut = imlocalbrighten(imgOut);
imgOut = imreducehaze(imgOut);
saveimg(imgO); %Salvataggio immagine HSVPP
```

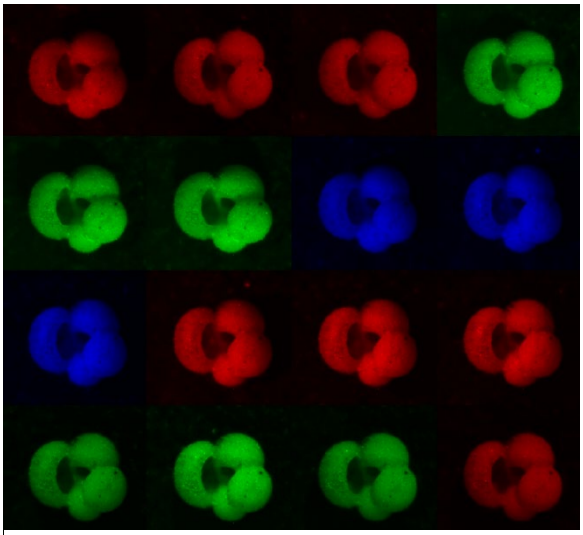
Metodo RGB

In questo metodo ogni immagine iniziale viene associata ad un piano colore in RGB.

Settando il valore della variabile groupingBy = n, le immagini originali vengono mappate di n in n in ogni piano colore.



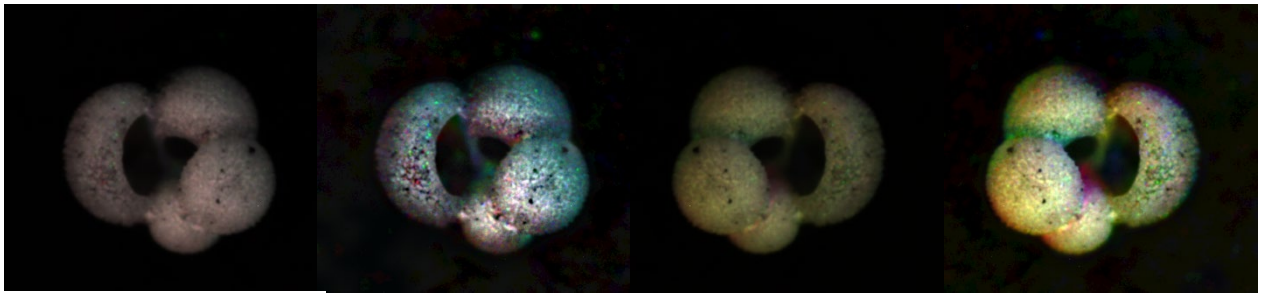
Grouping by 1



Grouping by 3

I quadrati dei valori così ottenuti vengono sommati a quelli della matrice dell'immagine finale per poi venir scalati nell'intervallo [0,1]. Da qui si ottiene il primo dataset (RGB<groupingBy>).

A queste immagini vengono poi applicati metodi di post proccessing, nell'ordine, imlocalbrighten e imreducehaze (*vedi sopra*) per ottenere un ulteriore dataset (RGB<groupingBy>PP)



RGB1 *RGB1PP* *RGB3* *RGB3PP*

Pseudo codice

```
groupingBy = 3; %a scelta dell'utente
pw = 2;

for i = randomStart([1:16])
   imgs(:,:,i) = readimage(i);
end

while d <= 16
   for channel = 1:3
      for i = 1:groupingBy
         if(d <= 16)
            RGB(:,:,chan) = RGB(:,:,chan) + imgs(:,:,d).^pw;
            d = d + 1;
         end
      end
   end
end

RGB = rescale(RGB,0,1);
saveimg(RGB); %salvataggio per dataset RGB3

RGB = imlocalbrighten(RGB);
RGB = imreducehaze(RGB);
saveimg(RGB); %salvataggio per dataset RGB3PP
```

Analisi della complessità

La complessità del metodo di trasformazione in immagine a colori è lineare rispetto al numero di immagini in scala di grigi e quadratico rispetto alla dimensione delle immagini. Sia T il tempo di esecuzione di un metodo su un pattern (16 immagini di partenza):

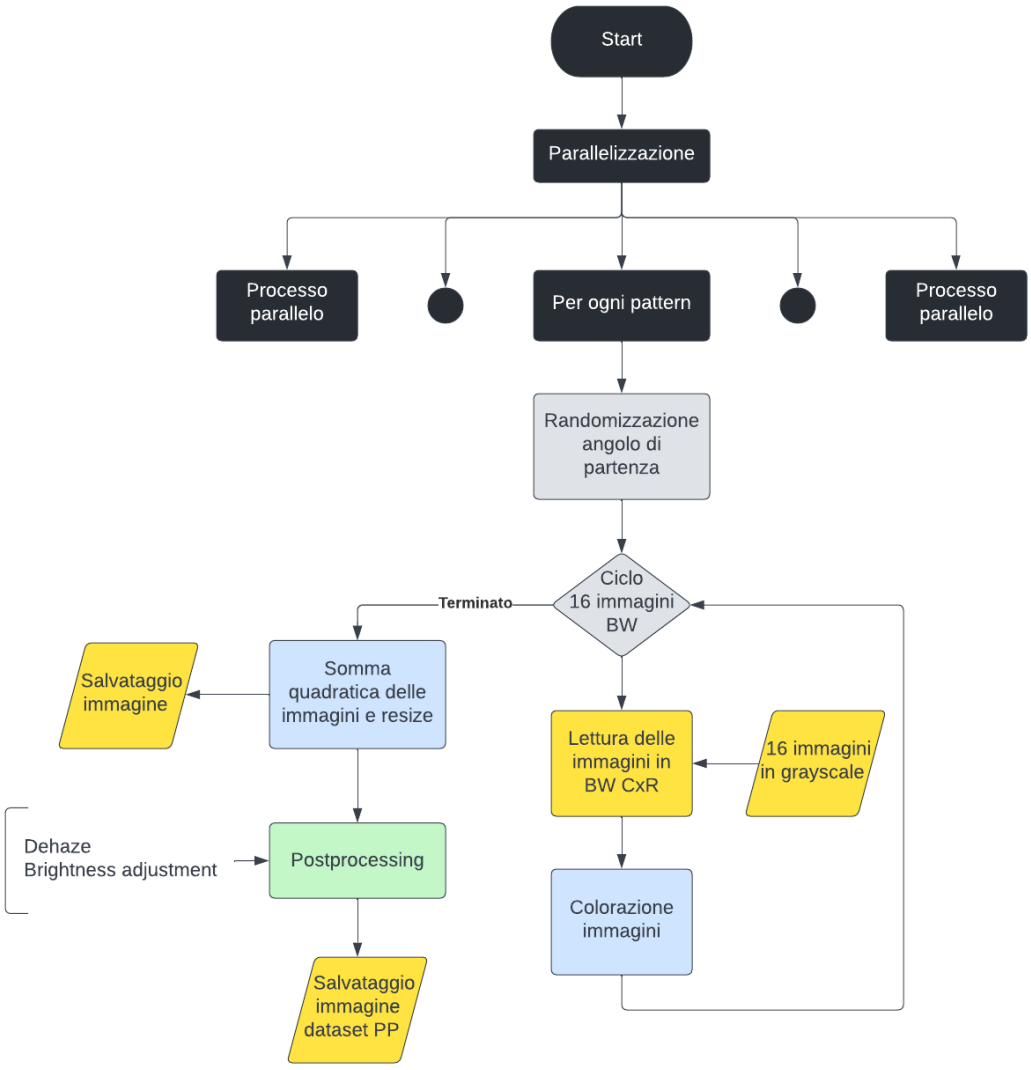
- 1/3*T è necessario per le operazioni di lettura/scrittura
- 1/3*T per la conversione delle 16 immagini in un'immagine RGB
- 1/3*T per i metodi di post-processing (dove presenti)

Su un Intel i7-8750H con 16 Gb di RAM e SSD e scheda grafica NVIDIA GTX 1050Ti sono necessari:

- 0.5 s circa per l'esecuzione di HSVPP per un pattern (16 immagini BW → 1 immagine RGB)

- 0.3 s circa per l'esecuzione di RGB1PP per un pattern (16 immagini BW → 1 immagine RGB)

Il flowchart generale dei metodi proposti è:



Prestazioni

I test sono stati effettuati con tutti i metodi sulla rete Alexnet adeguatamente modificata per permettere l'output di 7 classi invece che 100 e con parametri:

- Metodo di ottimizzazione: Adam
- Learning rate iniziale: 4×10^{-5}
- Batch size: 30
- Epoche massime: 20

Si sono così ottenuti i seguenti risultati:

Metodo	Fold1	Fold2	Fold3	Fold4	Media
RGB1	72.14%	71.39%	72.7%	72.7%	71.19%
RGB1 no rand	76.32%	76.11%	68.52%	71.87%	73.21%
RGB1PP	70.19%	71.67%	71.03%	69.36%	70.56%
RGB1PP no rand	76.60%	71.64%	73.82%	69.36%	72.86%
RGB3	70.47%	69.44%	72.42%	69.64%	70.49%
RGB3 no rand	74.65%	76.11%	77.72%	71.03%	74.88%
RGB3PP	70.75%	68.89%	72.70%	72.98%	71.33%
RGB3PP no rand	70.19%	78.06%	73.82%	79.94%	**75.50%**
HSV	66.30%	69.17%	65.18%	59.61%	65.06%
HSVPP	72.98%	69.44%	73.26%	73.26%	71.33%
HSVPP no rand	70.19%	72.78%	73.26%	71.81%	**72.02%**
PCA	68.80%	70.00%	73.82%	61.56%	67.54%
Percentile	66.76%	63.41%	72.35%	66.20%	67.54%

* Le righe con riportato la dicitura "no rand" si riferiscono a test eseguiti con dataset creati senza randomizzazione dell'inizio della sequenza delle immagini in BW.

Conclusioni

Tutti i metodi sviluppati hanno portato ad un incremento dell'accuracy del classificatore rispetto ai metodi di letteratura.

Analizzando i risultati ottenuti, in generale i classificatori allenati sui dataset costruiti senza randomizzazione dell'inizio della sequenza performano sensibilmente meglio. Ad un'analisi più approfondita del dataset, il bias ipotizzato risulta essere presente ma non per tutti i pattern di tutte le classi. Bisognerebbe organizzare il dataset in maniera tale che l'ordine di illuminazione delle immagini sia identico in ogni pattern di ogni classe e rieffettuare i test. Per un'applicazione reale del metodo proposto è inoltre necessario che l'acquisizione dei nuovi pattern avvenga con il medesimo ordine di illuminazione.