UNIVERSITÀ DEGLI STUDI DI PADOVA

Dipartimento di Ingegneria Dell'Informazione

Corso di Laurea in

Ingegneria Informatica

Plankton Image Preprocessing

Studenti:

Michele Russo
Dario Mameli
Alberto Spaziani

Anno accademico 2021/2022

Abstract

In questo lavoro, vengono sperimentate tecniche di image preprocessing per migliorare la classificazione automatica di immagini rappresentanti diverse specie di plancton.
Il problema della classificazione è risolvibile mediante l'implementazione di reti neurali profonde, di cui per il caso in esame sono protagoniste le reti neurali convoluzionali.
Le cosiddette CNN, sono infatti uno strumento largamente usato per affrontare problemi relativi alla classificazione di immagini, soprattutto ai fini della ricerca scientifica, di cui peraltro sono esse stesse vasto oggetto di interesse.
E' noto che, al fine di migliorare le prestazioni di una CNN nella classificazione di un dataset di immagini, sia cruciale l'utilizzo di tecniche di pre processing sulle immagini stesse, per meglio evidenziare le caratteristiche principali di ciascuna classe.
Dunque questo è quanto è stato applicato nel presente lavoro, nel caso specifico di un dataset di immagini di plancton.

1. Introduzione

1.1 Plancton

Il Plancton è un gruppo di diversi organismi che vivono in ambiente marino, fluviale o lacustre, e che non possono nuotare contro corrente. Questi organismi costituiscono una fonte di cibo primaria per molte specie acquatiche. Pertanto lo studio della loro composizione, e dei loro cambiamenti nel tempo, è di una valenza molto elevata nell'ambito della ricerca, e in particolare per il controllo degli ecosistemi marini.
La tassonometria del plancton si compone di vari rami, e ciò logicamente comporta delle difficoltà per la classificazione delle varie specie, proprio perché le differenze tra una famiglia ed un'altra sono molto sottili.
I criteri base utilizzati in ambito di ricerca sono i seguenti:
1- forma
2- composizione corpuscolare interna

1.2 Dataset

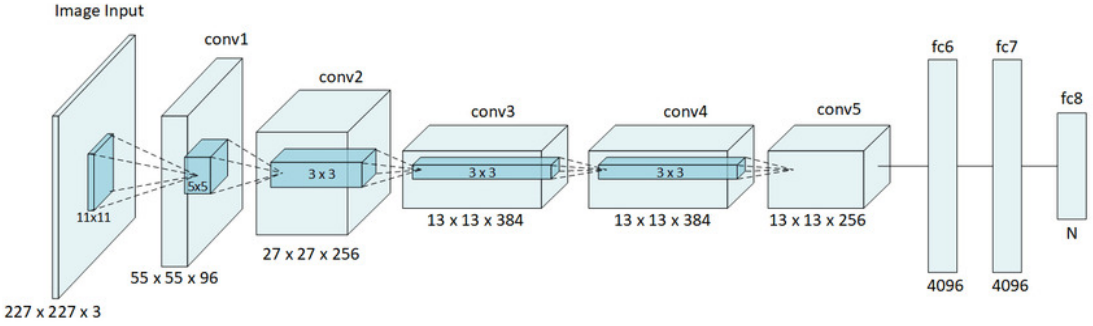
Si veda [3] della bibliografia per ottenerlo.

1.3 AlexNet e ResNet

Nel nostro lavoro vengono testate principalmente due reti, ossia AlexNet e ResNet-50, in quanto VGG 16 è risultata essere eccessivamente pesante per l'hardware a disposizione.

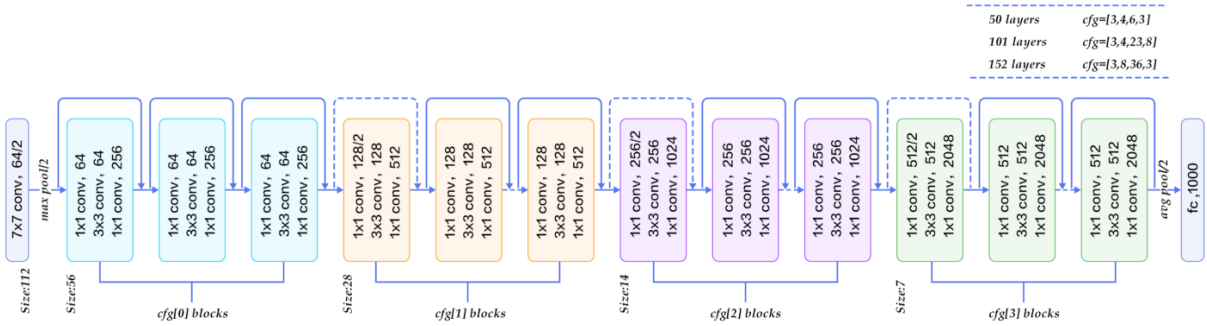
AlexNet

AlexNet è una rete neurale convoluzionale profonda 8 layers.
Il modello testato è stato pre addestrato su più di un milione di immagini prese dal database ImageNet di Google, ed è in grado di classificare fino a 1000 categorie diverse di oggetti, quali possono essere tastiere, mouse, matite, fino a diverse specie animali.
Questo ha portato la rete ad apprendere una notevole quantità di rappresentazioni delle features di una vasta gamma di immagini.
La rete accetta in input immagini di dimensione 227-227.
La figura seguente, schematizza la struttura di AlexNet.[1]

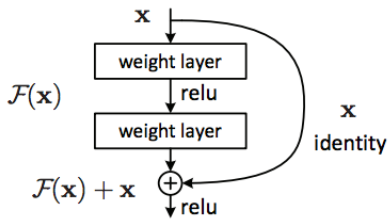


ResNet-50

ResNet-50 è una rete neurale convoluzionale profonda 50 layers.
Come per AlexNet, il modello testato è stato pre addestrato su più di un milione di immagini dal database ImageNet di Google, ed è in grado di classificare immagini fino a 1000 categorie diverse di oggetti. Anche in questo caso dunque la rete ha appreso un'elevata varietà di rappresentazioni delle features per una vasta gamma di immagini.
In questo caso l'input è costituito da immagini di dimensione 224-224.
La figura seguente schematizza la struttura di ResNet-50.[2]



Questa rete è interessante, in quanto pur avendo 50 livelli di profondità non incorre nel cosiddetto problema del vanishing gradient che perseguita le altre CNN, implementando un meccanismo di cosiddetto "skip connection", che connette direttamente l'output di un layer con l'input di un layer successivo, azzerando la funzione di attivazione.[2]



Prestazioni

Un primo passo nell'approccio allo sviluppo del codice è stato quello di confrontare i due modelli di rete per avere un'idea di quale sarebbe stata più efficace per lo scopo del nostro problema di classificazione.
Abbiamo dunque ripetutamente testato il funzionamento delle due reti nel caso di input nella forma di immagini con come unico preprocessing applicato il resize (vedi 2.2), ottenendo i seguenti risultati (parziali) relativi alla accuracy:

- ResNet-50
  [0.742311770943796,0.740190880169671]
  Media: 74,13%
- AlexNet
  [0.721633085896076,0.700424178154825]
  Media: 71,1%

In base all'esempio riportato (e agli altri non riportati), si vede come ResNet-50 abbia una performance superiore, da cui la scelta di non proseguire con la rete AlexNet, ma solo con ResNet-50.

2. L'algoritmo

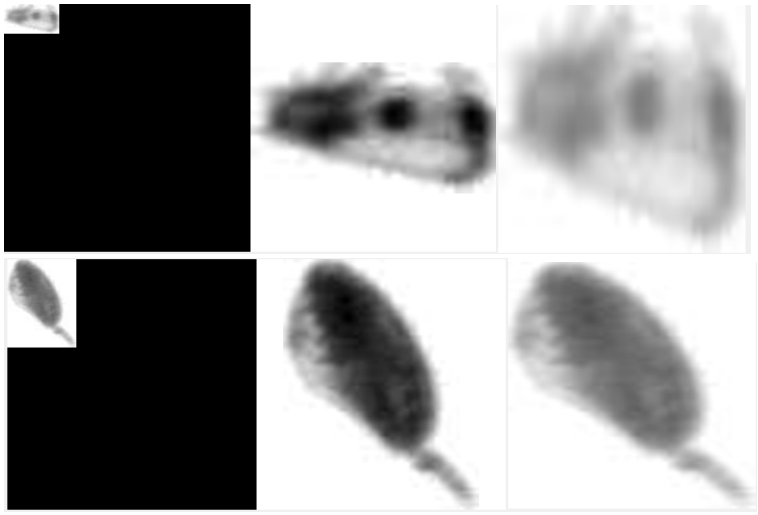
2.1 Approccio generale

La procedura generale applicata è quella di una variante di k-fold cross-validation in cui k-1 fold vengono usati per il training e 1 per il testing a rotazione. Nel caso in esame k=2 e pertanto si hanno 1 sessione di training e 1 di testing che si susseguono, senza una fase intermedia di validation, totalizzando 2 sessioni di training e 2 di testing.

2.2 Resize

Dall'analisi del data-set, ci siamo accorti che la funzione di resize di matlab, diminuiva di molto la qualità, già di per sé non molto elevata, delle immagini del plancton. Al fine di ridimensionare questo problema abbiamo creato una nuova funzione che fosse in grado di aumentare la grandezza delle immagini fino alla grandezza dell'input di resnet(224-224), senza perdere eccessivamente in qualità. La funzione prende il lato più lungo dell'immagine in input, lo ingrandisce fino a 224, conta il numero di pixel per l'ingrandimento e poi, successivamente, ingrandisce l'altra dimensione di un numero di pixel pari a quelli calcolati. Infine viene aggiunto il padding per colmare il gap tra 224 e l'ultima dimensione.

```
siz=size(IM);
s1=224;
s2=224;
if siz(1)>siz(2)
    s2=NaN;
    s1=224;
elseif siz(2)>siz(1)
    s1=NaN;
    s2=224;
else
    s1=224;
    s2=224;
end
FM=imresize(IM,[s1 s2]); %resize mantenendo aspect ratio
c=size(FM);
c(1)=224-c(1);
c(2)=224-c(2);
c(1)=cast(c(1)/2,'uint8');
c(2)=cast(c(2)/2,'uint8');
paddedImage = padarray(FM,[c(1) c(2)],255); %aggiungo padding con bianco
```



L’immagine di destra è ottenuta con un resize normale, mentre quella a sinistra con il metodo proposto sopra.
Si riportano quindi i risultati dell’applicazione del metodo resize della libreria matlab a confronto con quello descritto, sul test set.

	Accuracy (ACC)	ACC media
Resize matlab	[0.7270, 0.7147]	72.09%
Resize con padding	[0.7423, 0.7402]	74.13%

2.3 Data augmentation

Dopo il pre processing generale (di cui al 2.2) e dopo i metodi di pre processing di cui si tratterà nel dettaglio al capitolo 3, vengono utilizzate delle modalità standard di data augmentation al fine di aumentare il numero di pattern disponibili e raggiungere una convergenza più robusta, quali riflessioni, riscalamenti, rotazioni e traslazioni delle immagini.

```
%creazione pattern aggiuntivi, data augmentation
imageAugmenter = imageDataAugmenter( ...
    'RandXReflection',true, ...
    'RandXScale',[1 2], ...
    'RandYReflection',true, ...
    'RandYScale',[1 2],...
    'RandRotation',[-10 10],...
    'RandXTranslation',[0 5],...
    'RandYTranslation', [0 5]);
trainingImages = augmentedImageSource(imageSize,trainingImages,categorical(y'),'DataAugmentation',imageAugmenter);
```

2.4 Transfer Learning

A questo punto viene costruita la rete.

L’allenamento di una rete neurale spesso richiede un numero elevato di patterns con i quali essere addestrata, e un’enorme disponibilità computazione, che ovviamente non tutti hanno. Per questo si sono sviluppate delle tecniche volte al riutilizzo di reti pre-addestrate.
Questa metodologia, ci permette fin da subito di raggiungere delle prestazioni in termini di accuratezza più elevate, riducendo sensibilmente i tempi relativi all’addestramento.
Esistono vari modelli di reti pre-addestrate, tra cui le già citate Alexnet e ResNet.
La tipologia di Transfer Learning utilizzata per provare i vari metodi di pre-processing è il fine-tuning. Esso consiste nella sostituzione degli ultimi 3 layer della rete pre-addestrata, con dei nuovi relativi al problema preso in analisi. Nel caso di ResNet-50, questo deve avvenire mediante il grafo della rete (lgraph),

```
lgraph = layerGraph(net);
lgraph = removeLayers(lgraph, {'ClassificationLayer_fc1000','fc1000_softmax','fc1000'});

layers = [
    fullyConnectedLayer(numClasses,'WeightLearnRateFactor',20,'BiasLearnRateFactor',20)
    softmaxLayer
    classificationLayer];

lgraph = addLayers(lgraph,layers);
lgraph = connectLayers(lgraph,'avg_pool','fc');
%analyzeNetwork(lgraph)

netTransfer = trainNetwork(trainingImages,lgraph,options);
```

Viene dunque avviato il training, testing e calcolo dell’accuracy sul test set.

2.5 Tipologia di apprendimento

Il tipo di apprendimento utilizzato è supervisionato, ovvero, tutti i pattern del training set presentano già nota la propria classe di appartenenza.

3. Metodi di preprocessing

Gli esperimenti portati avanti, sono stati effettuati su immagini in grayscale.
Il metodo di resize è applicato in tutti i successivi metodi.

3.1 globalTraining

Questo metodo si occupa di andare a esaltare le informazioni relative alle cosiddette “setae” (o setole) del plancton e relativa forma esterna, estraendo le features corrispondenti.
Gli step si compongono nel seguente modo:

- **filtro bilaterale**: applica un filtro bilaterale Gaussiano all’immagine grayscale (come nel caso in esame) oppure RGB, preservandone i bordi. Questo filtro digitale non lineare permette di rimuovere del rumore dalle immagini e smussare la forma del plankton.
  “Sostituisce l'intensità di ogni pixel con una media ponderata dei valori di intensità dei pixel vicini. Questo peso può essere basato su una distribuzione gaussiana. Fondamentalmente, i pesi dipendono non solo dalla distanza euclidea dei pixel, ma anche dalle differenze radiometriche (ad esempio, differenze di intervallo, come l'intensità del colore, la distanza della profondità, ecc.).”[5]
- **aggiustamento del contrasto**: modifica i valori di intensità dell’immagine grayscale saturando 1% dei pixel con valore minore e 1% di quelli con intensità maggiore.
- **Sobel edge detection**: esalta i bordi nei punti in cui il gradiente dell’immagine è massimo, usando l’approssimazione di Sobel. Effettua una misurazione del gradiente in un’immagine 2-D enfatizzando le regioni con elevata frequenza spaziale, che corrispondono ai bordi. E’ tipicamente usato per trovare un’approssimazione del modulo del gradiente in ogni punto dell’immagine.[6]

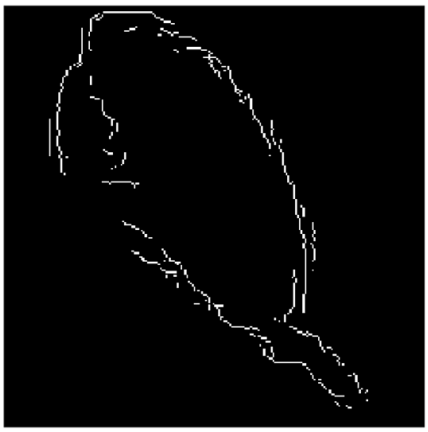
CODICE:

```
function [I]=globalTraining(img)
    J = imbilatfilt(img);
    J=imadjust(J);
    M=edge(J,'sobel');
    I = M;
end
```

Di seguito vengono mostrati i risultati relativi all’applicazione del metodo:

Originale

Bilaterale, contrasto



Sobel

	Accuracy (ACC)	ACC media
Originale	[0.742311770943796,0.740190880169671]	74,13%
globalTraining	[0.632025450689290,0.648462354188759]	64,03%

N.B. Originale = nessun preprocessing

Si osserva come sia calata l'accuracy media. Ulteriori considerazioni sul risultato verranno presentate al 3.3.

3.2 local Training

Questo metodo ha come obiettivo quello di estrarre le features locali del plancton, in particolare la loro texture interna e trasparenza, esaltando il nucleo e citoplasma degli organismi.
Viene adoperato come metodo il canny edge detector.

- **Canny edge detection**: Esalta i bordi dell'immagine andando a cercare i punti di massimo locale del gradiente dell'immagine. Calcola il gradiente usando la derivata di un filtro Gaussiano. Usa due soglie per identificare bordi marcati e deboli, includendo nell'output quelli deboli qualora connessi a quelli marcati. Usando due soglie il metodo è più resistente al rumore e ha una probabilità maggiore di identificare i bordi effettivamente deboli.

CODICE:

```
function [I]=localTraining(img)
    M = edge(img,'canny');
    I = M;
end
```

Vengono quivi riportati i risultati dell'applicazione del metodo:

Originale Canny

	Accuracy (ACC)	ACC media
Originale	[0.742311770943796,0.740190880169671]	74,13%
globalTraining	[0.632025450689290,0.648462354188759]	64,03%
localTraining	[0.682926829268293,0.685577942735949]	68,43%

Le prestazioni sono aumentate rispetto al globalTraining, ma rimangono inferiori all'addestramento sulle immagini originali. Nel successivo paragrafo si formulerà un'ipotesi esplicativa.

3.3 combinedTraining

I metodi globalTraining e localTraining sono risultati essere piuttosto inefficienti se presi singolarmente, in quanto vanno ad estrarre le features rispettivamente esterne ed interne. Pertanto l'ideale sarebbe effettuare un preprocessing unico utilizzando la sovrapposizione degli effetti dei due metodi.
Si osserva dallo studio riferito alla voce [4] che la rete progettata per la classificazione del plankton va in ultima analisi a combinare gli output degli addestramenti sulle global features, sulle local features e sui pattern originali.
Da questa idea nasce il combinedTraining, che crea una maschera binaria data da un'operazione di OR sulle due maschere ottenute dai metodi global e local, andando poi a sovrapporre tale maschera all'immagine originale, combinando così tutti gli effetti.
In aggiunta, prima dell'estrazione delle features, è stato utilizzato un metodo di sharpening, non presente nei requisiti dei due preprocessing precedentemente implementati.
Più in particolare gli steps sono:

- filtro bilaterale

- aggiustamento del contrasto
- **sharpening**: si definisce sharpness il contrasto fra due colori differenti. Questo metodo aumenta il contrasto lungo i contorni dove si incontrano colori differenti

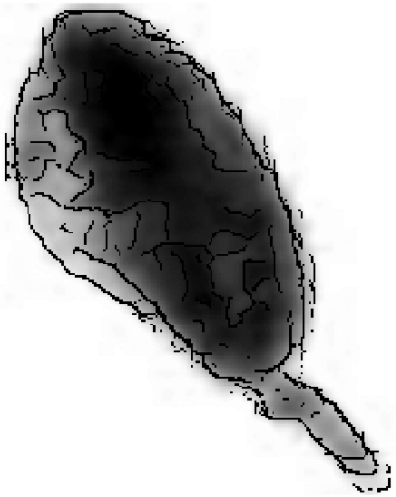
Bilaterale, contrasto Sharpening

- Sobel edge detector e Canny edge detector in parallelo (stesso input)
- Operatore OR sui due output
- Overlay della maschera sul pattern originale

CODICE:

```
function [I]=combinedTraining(img)
    B = imbilatfilt(img);
    B=imadjust(B);
    J = imsharpen(B);
    S=edge(J,'sobel');
    C = edge(J,'canny');
    I = imoverlay(J, S | C, 'black');
end
```

Si riportano i risultati ottenuti dall'applicazione del metodo:



Originale Output

	Accuracy (ACC)	ACC media
Originale	[0.742311770943796,0.740190880169671]	74,13%
globalTraining	[0.632025450689290,0.648462354188759]	64,03%
localTraining	[0.682926829268293,0.685577942735949]	68,43%
combinedTraining	[0.773064687168611,0.778897136797455]	77,6%

Si osserva un miglioramento netto rispetto ai singoli metodi di estrazione delle features globali e locali, e un buon miglioramento rispetto al training sui pattern originali.

3.4 filtri base

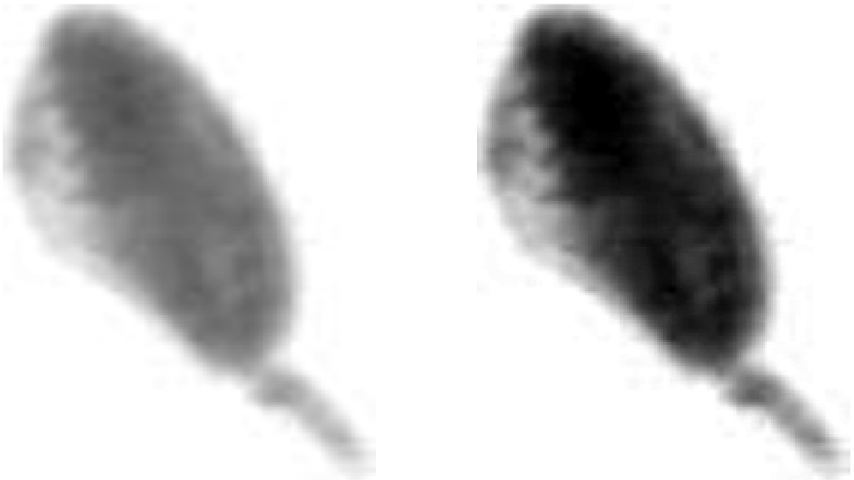
Il preprocessing 3.3 ha prodotto risultati positivi sul miglioramento dell'accuracy sul test set. Tuttavia, al fine di meglio determinare quali filtri tra quelli utilizzati fossero più efficaci, si è deciso di testare gruppi minori di filtri.
In questo metodo si testano i filtri di base applicati nei precedenti metodi:

- aggiustamento del contrasto
- sharpening

CODICE:

```
elseif type==4 % training base
    IM=paddedImage;
    IM = imadjust(IM);
    IM = imsharpen(IM);
```

I risultati dell'applicazione del metodo si riportano in seguito:



Originale Output

	Accuracy (ACC)	ACC media
Originale	[0.742311770943796,0.740190880169671]	74,13%
globalTraining	[0.632025450689290,0.648462354188759]	64,03%
localTraining	[0.682926829268293,0.685577942735949]	68,43%
combinedTraining	[0.773064687168611,0.778897136797455]	77,6%
base filters	[0.790031813361612,0.755037115588547]	77,25%

La performance è comparabile a quella di combinedTraining. E' dunque possibile che la maschera utilizzata in tale metodo (3.3) non sia tanto efficace nell'esaltazione delle features quanto risultano invece essere i filtri relativi all'aggiustamento del contrasto e allo sharpening dei bordi. Tuttavia il miglioramento del 3.3 su questo, seppur in termini di decimi percentuali, è comunque constatabile.

3.5 filtro bilaterale

In questo metodo viene applicato il preprocessing base 3.4 a seguito di un filtro bilaterale.

- filtro bilaterale
- aggiustamento del contrasto
- sharpening

CODICE:

```
elseif type==5 % training filtro bilaterale
    IM=paddedImage;
    IM = imbilatfilt(IM);
    IM = imadjust(IM);
    IM = imsharpen(IM);
```

Si riportano i risultati ottenuti dall'applicazione del metodo:

Originale

Bilaterale

Base filters

	Accuracy (ACC)	ACC media
Originale	[0.742311770943796,0.740190880169671]	74,13%
globalTraining	[0.632025450689290,0.648462354188759]	64,03%
localTraining	[0.682926829268293,0.685577942735949]	68,43%
combinedTraining	[0.773064687168611,0.778897136797455]	77,6%
base filters	[0.790031813361612,0.755037115588547]	77,25%
filtro bilaterale	[0.798515376458112,0.775185577942736]	78,69%

Si osserva come l'accuratezza media sia migliorata, raggiungendo la massima performance fra i metodi finora descritti. Evidentemente il filtro bilaterale risulta essere molto efficace nel suo scopo di ridurre il rumore nelle immagini, esaltandone le features di conseguenza.

3.6 filtro anisotropico

In questo metodo viene applicato il filtro anisotropico e successivamente i filtri di base descritti al 3.4.

- **filtro anisotropico**: rimuove il rumore ad alta frequenza nell'immagine in maniera adattiva preservando i bordi dell'immagine.
- aggiustamento del contrasto
- sharpening

CODICE:

```
elseif type==6 % training filtro anisotropico
    IM=paddedImage;
    IM = imdiffusefilt(IM);
    IM = imadjust(IM);
    IM = imsharpen(IM);
```

Si riportano in seguito i risultati dovuti all'applicazione del metodo:

Originale

Anisotropico

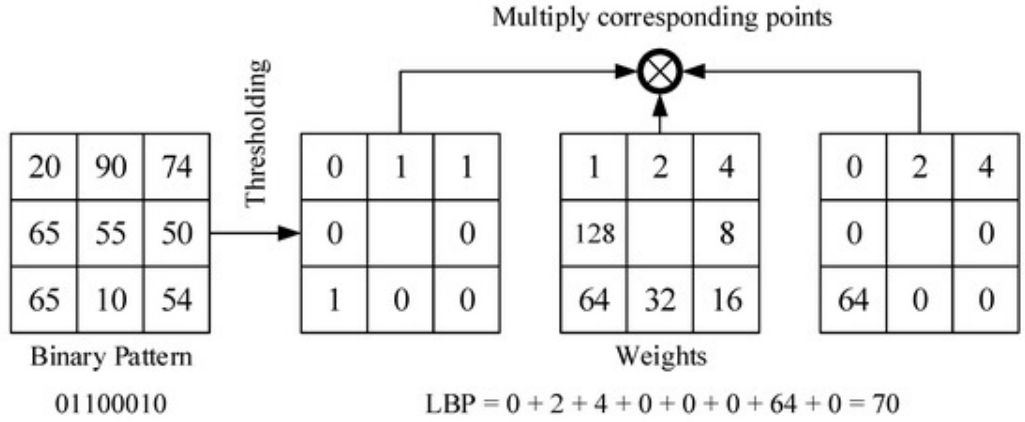
Base filters

	Accuracy (ACC)	ACC media
Originale	[0.742311770943796,0.740190880169671]	74,13%
globalTraining	[0.632025450689290,0.648462354188759]	64,03%
localTraining	[0.682926829268293,0.685577942735949]	68,43%
combinedTraining	[0.773064687168611,0.778897136797455]	77,6%
base filters	[0.790031813361612,0.755037115588547]	77,25%
filtro bilaterale	[0.798515376458112,0.775185577942736]	78,69%
filtro anisotropico	[0.787380699893955,0.759809119830329]	77,36%

Si può osservare come il filtro anisotropico abbia una performance peggiore rispetto a quello bilaterale nella cancellazione del rumore, peraltro con un incremento minimo rispetto al suo non utilizzo (metodo 3.4).

3.7 Local Binary Pattern

L'algoritmo di Local Binary Pattern si basa su un operatore non parametrico che sintetizza la struttura locale di una immagine. Ad un particolare pixel Pc dell'immagine con coordinate (Xc, Yc), l'operatore LBP è definito come una sequenza binaria ordinata di confronti di intensità di colore fra il pixel Pc e i pixel appartenenti al vicinato considerato. In particolare, se l'intensità del pixel adiacente è maggiore o uguale dell'intensità del pixel centrale Pc allora viene usato il valore 1 altrimenti 0.



Il risultato dell'applicazione di LBP sull'immagine, è riportato a sinistra. Per ottenere questo effetto abbiamo centrato i valori dell'immagine sullo zero , sottraendo -128, poiché l'immagine di partenza presentava molti valori sullo zero(255), quindi l'immagine finale veniva molto simile ad un'immagine completamente bianca.

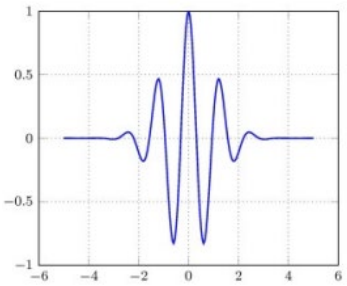
	Accuracy (ACC)	ACC media
Originale	[0.742311770943796,0.740190880169671]	74,13%
globalTraining	[0.632025450689290,0.648462354188759]	64,03%
localTraining	[0.682926829268293,0.685577942735949]	68,43%
combinedTraining	[0.773064687168611,0.778897136797455]	77,6%
base filters	[0.790031813361612,0.755037115588547]	77,25%
filtro bilaterale	[0.798515376458112,0.775185577942736]	78,69%
filtro anisotropico	[0.787380699893955,0.759809119830329]	77,36%
LBP	0.692470837751856	69,25%

Le prestazioni ottenute con questo metodo sono tra le peggiori. Questo probabilmente è dovuto al fatto che il dataset non sia tanto adatto all'utilizzo di questa tecnica di preprocessing, soprattutto alla luce del fatto che abbiamo aree completamente bianche ed altre completamente scure.

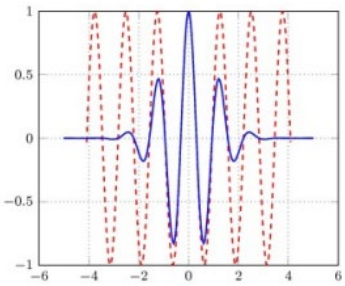
3.8 Discrete 2D Wavelet Transform

La trasformata Wavelet è un'altra operazione per l'estrazione di informazioni da texture, originata come variante della trasformata di Fourier.

Rispetto alla trasformata di Fourier la Wavelet presenta una trasformata localizzata sia nel dominio del tempo, sia che nel dominio delle frequenze, mentre la trasformata di Fourier solo in quella delle frequenze. Il vantaggio che ne deriva è relativo alla creazione delle funzioni d'onda, che nella wavelet variano sia in posizione che in frequenza, in modo da avere estensione limitata, mentre la trasformata di Fourier inserisce un gran numero di funzioni seno, in modo che si annullino a vicenda, per rappresentare segnali transienti.

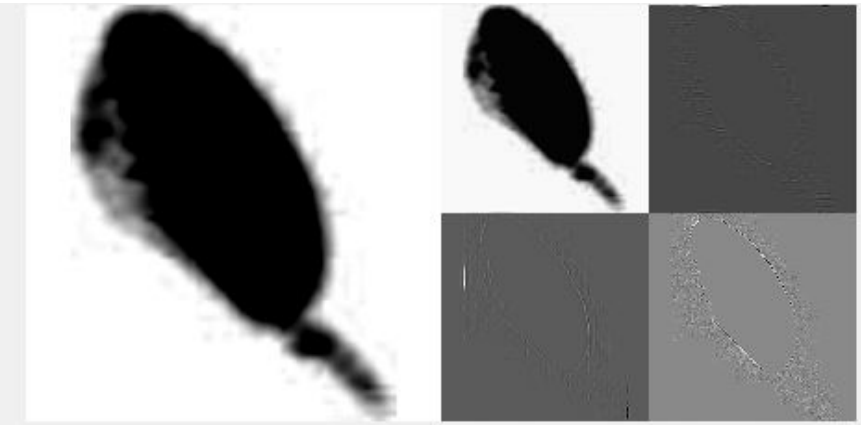


esempio di wavelets

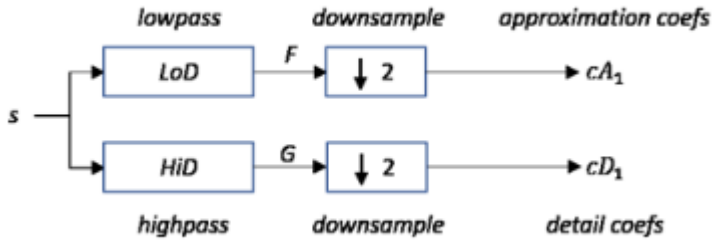


comparazione con segnale sinusoidale

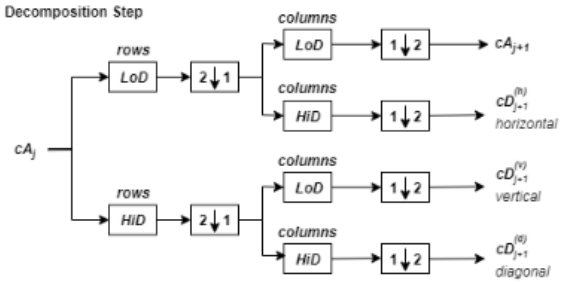
La Wavelet bidimensionale produce un'immagine composta da quattro quadranti, ciascuno con significato diverso. Il primo consiste in un'approssimazione dell'immagine originale e può essere usato anche per effettuare compressione con perdita. Gli altri tre quadranti evidenziano rispettivamente feature verticali, orizzontali e diagonali.



La wavelet 2D è molto simile all'algoritmo della wavelet 1D. Infatti la wavelet bidimensionale è ottenuta prendendo il tensori della monodimensionale e le funzioni di scalamento.
La wavelet monodimensionale, parte da un segnale, per creare due set di coefficienti: di approssimazione e di dettaglio; ottenuti dalla convoluzione del segnale s rispettivamente con un filtro passa-basso ed uno passa-alto



la bidimensionale applica il processo sopra descritto prima sulle righe, e poi successivamente sulle colonne, producendo quindi le 4 immagini sopra riportate.



where

- $2\downarrow 1$ – Downsample columns: keep the even-indexed columns
- $1\downarrow 2$ – Downsample rows: keep the even-indexed rows
- rows X – Convolve with filter *X* the rows of the entry
- columns X – Convolve with filter *X* the columns of the entry

CODICE:

```
elseif type==8
    paddedImage=paddedImage-175;
    [cA,cH,cV,cD] = dwt2(paddedImage,'sym4','mode','per');
    I1=cat(2,cA,cH);
    I2=cat(2,cV,cD);
    IM=cat(1,I1,I2);
```

Si mostrano in seguito i risultati sul test set relativi all'applicazione del metodo

	Accuracy (ACC)	ACC media
Originale	[0.742311770943796,0.740190880169671]	74,13%
globalTraining	[0.632025450689290,0.648462354188759]	64,03%
localTraining	[0.682926829268293,0.685577942735949]	68,43%
combinedTraining	[0.773064687168611,0.778897136797455]	77,6%
base filters	[0.790031813361612,0.755037115588547]	77,25%
filtro bilaterale	[0.798515376458112,0.775185577942736]	78,69%
filtro anisotropico	[0.787380699893955,0.759809119830329]	77,36%
LBP	0.692470837751856	69,25%
Wavelet	[0.763520678685048,0.741251325556734]	75,24%

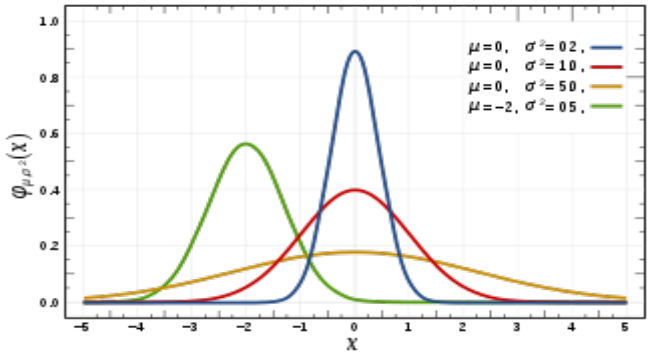
Il miglioramento offerto dalla wavelet non porta a grossi vantaggi in termini di prestazioni, probabilmente dovuto al fatto che l'immagine presenta zone completamente bianche, ed altre interamente scure, quindi la matrice di approssimazione resta molto fedele all'originale, come è possibile vedere sopra, mentre il contributo delle restanti tre immagini è marginale, dato che i valori delle matrici ad esse correlati sono pressappoco identici.

3.9 Deconvoluzione Gaussiana

Dallo studio del dataset, è stato possibile notare una forte presenza di rumore. Proprio per questo motivo, si è deciso di provare vari filtri in modo da attenuarlo. Uno di quelli usati è stata la deconvoluzione Gaussiana. In generale il rumore Gaussiano è un rumore additivo ai segnali, il rumore ha una densità di probabilità uguale a quella della distribuzione normale:

$$p_G(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}}$$

La funzione gaussiana dipende strettamente dai valori di sigma, indica la varianza , graficamente indica l'ampiezza della curva, mentre μ indica il valore atteso, e graficamente indica dove è centrata la curva



Tipicamente la presenza di questo rumore nelle immagini è dovuta alla scarsa illuminazione, oppure al rumore stesso del circuito elettrico, in caso di macchine estremamente sensibili.
La deconvoluzione è il passo che permette la rimozione di questo rumore dalle immagini.

Come si può notare, sulla destra l'immagine di output appare definita un po' meglio rispetto alla sua controparte originale sulla sinistra.

CODICE:

```
elseif type==7 % training filtro gaussiano
    IM=paddedImage;
    PSF = fspecial('gaussian',5,5);
    IM = deconvlucy(IM,PSF,5);
    IM = imadjust(IM);
    IM = imsharpen(IM);
```

Si riportano in seguito i risultati relativi all'applicazione del metodo sul test set:

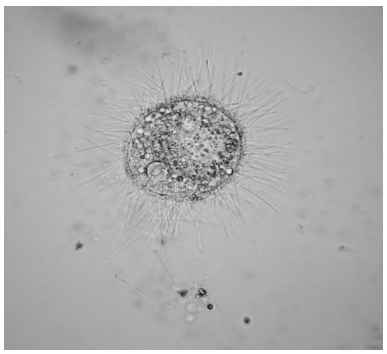
	Accuracy (ACC)	ACC media
Originale	[0.742311770943796,0.740190880169671]	74,13%
globalTraining	[0.632025450689290,0.648462354188759]	64,03%
localTraining	[0.682926829268293,0.685577942735949]	68,43%

combinedTraining	[0.773064687168611,0.778897136797455]	77,6%
base filters	[0.790031813361612,0.755037115588547]	77,25%
filtro bilaterale	[0.798515376458112,0.775185577942736]	78,69%
filtro anisotropico	[0.787380699893955,0.759809119830329]	77,36%
LBP	0.692470837751856	69,25%
Wavelet	[0.763520678685048,0.741251325556734]	75,24%
Deconv Gauss	[0.774655355249205,0.769883351007423]	77,23%

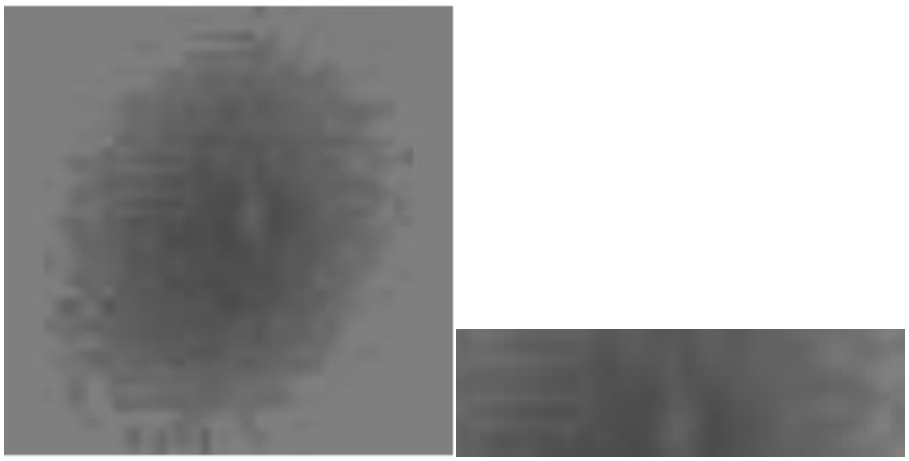
Si osserva un miglioramento del 3% rispetto all'originale, quindi si ha un miglioramento per quanto riguarda la visualizzazione dell'immagine per il modello, ma nonostante questo il miglioramento non è sufficiente da poter dire che il tipo di errore presente nelle immagini in input è proprio quello Gaussiano.

4. Commenti finali

Rispetto alle immagini originali si nota un miglioramento generale utilizzando tutti i metodi tranne global, local e LBP, per le motivazioni già addotte ai rispettivi punti 3.1, 3.2, 3.7. Inoltre, tra i metodi testati, risulta vincente a livello di incremento prestazionale l'applicazione del filtro bilaterale, con un margine di circa il 4%, dimostrando come sia il migliore nel campo della riduzione del rumore rispetto al nostro dataset di riferimento. Ciononostante, questo miglioramento ottenuto non è il massimo che ci si potrebbe aspettare. Sicuramente un motivo è che il dataset è costituito da immagini molto rumorose e pertanto non è possibile utilizzare al meglio tutti i criteri di distinzione utilizzati nella tassonometria biologica, come riportato nell' 1.1.
Si riporta in seguito un esempio di immagine di plankton al microscopio con risoluzione elevata:



Di fatto, come si nota dalle immagini riportate come esempi nel capitolo 3, rispetto all'immagine sopra riportata non è possibile distinguere chiaramente i corpuscoli interni che sono importantissimi per la distinzione tra classi che presentano forme simili.
Parte di questo problema riteniamo possa essere dovuto da un'eccessiva perdita durante la compressione delle immagini del dataset. Infatti, come si può notare dall'immagine sotto riportata, risulta evidente una divisione in artefatti della stessa, che certamente è indice di una qualità non ottimale per la classificazione.



5. Bibliografia

[1] Hemmer, Martin & Khang, Huynh Van & Robbersmyr, Kjell & Waag, Tor & Meyer, Thomas. (2018). Fault Classification of Axial and Radial Roller Bearings Using Transfer Learning through a Pretrained Convolutional Neural Network. Designs. 2. 56. 10.3390/designs2040056.
[2] https://blog.devgenius.io/resnet50-6b42934db431
[3] https://www.dropbox.com/s/gdm8n8lqxn6v7iq/Datas_44.mat?dl=0
[4] https://www.dropbox.com/s/gph8d9yuzbpizeg/PlanktonPreProcessing.pdf?dl=0
[5] https://www.no-regime.com/ru-it/wiki/Bilateral_filter
[6] https://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.html
[7] https://it.wikipedia.org/wiki/Wavelet
[8]https://it.mathworks.com/help/wavelet/ug/two-dimensional-discrete-wavelet-analysis.html?s_tid=mwa_osa_a
[9]https://it.mathworks.com/help/wavelet/ref/dwt.html#:~:text=Description,-example&text=example-,%5B%20cA%20%2C%20cD%20%5D%20%3D%20dwt(%20x%20%2C%20wname%20),vector%20cD%20of%20the%20DWT