Deep learning e segmentazione per la biologia cellulare

Utilizzo del transfer learning in Matlab per l'identificazione di cellule in microscopia

Alessandro Mastrofini
Elaborazione di Immagini
Università degli Studi di Roma Tor Vergata

2022

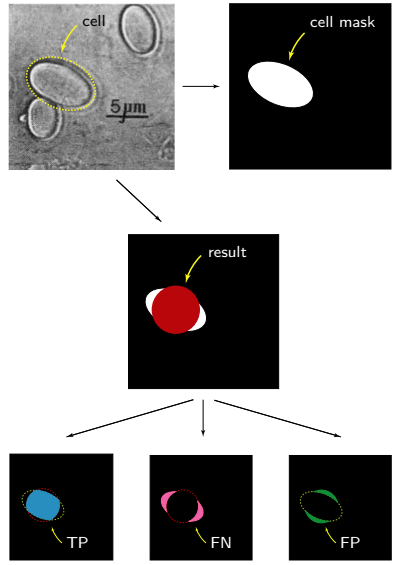
Image segmentation

- Pixel-based
- Edge-based
- Region-based
- Model-based
- **Supervised methods**

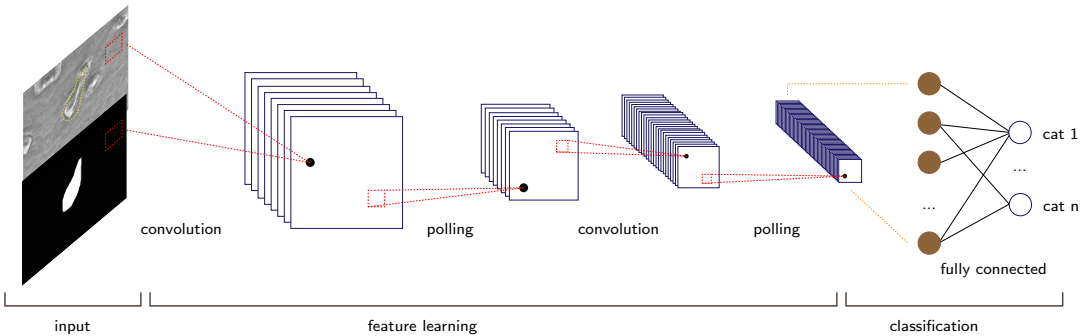
$$\text{CM} = \frac{TP}{TP+FN} = \frac{TP}{\text{Total arai in GT}}$$

$$\text{CR} = \frac{TP}{TP+FP} = \frac{TP}{\text{Total area in BW}}$$

$$\text{FM} = \frac{2 \cdot CM \cdot CR}{CM+CR} \in [0;1]$$



Transfer learning

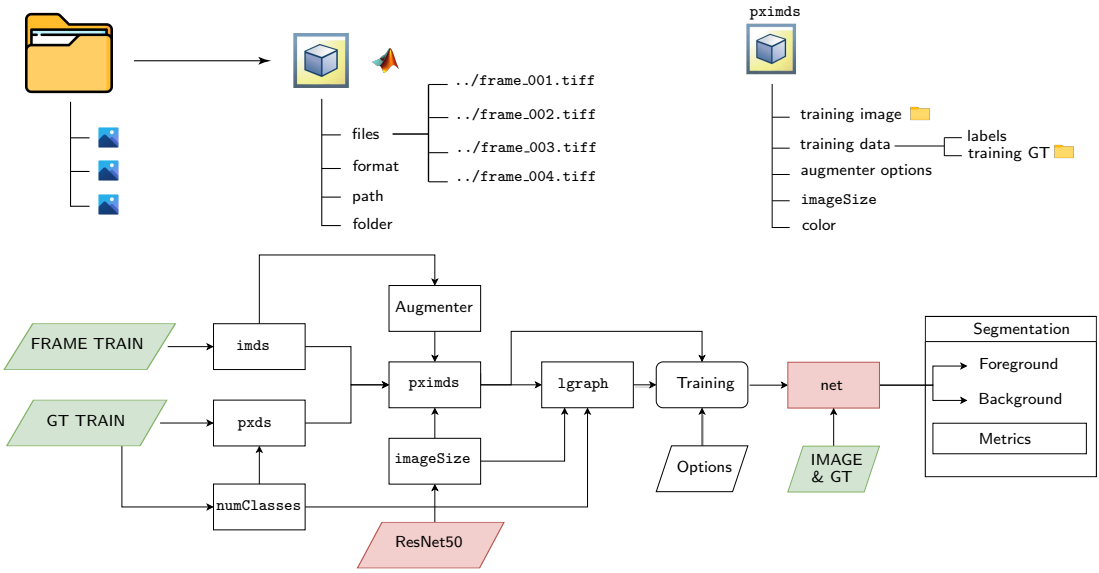


{B}==1
{N}==0

```
pxds=pixelLabelDatastore(...
strcat(newPath,'dataset\GT_TRAIN'),...
["N","B"],[0 1]);
```

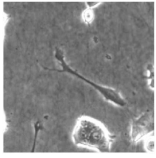
```
pxLayer = pixelClassificationLayer(...
'Name','labels','Classes',tbl.Name,...
'ClassWeights',classWeights);
```

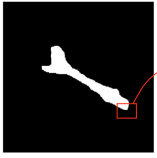
Training dataset



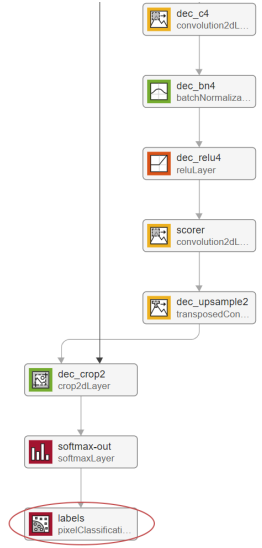
Classification layer

```
lgraph = deeplabv3plusLayers(imageSize,numClasses,"resnet50");
% balance predominance of 0
tbl = countEachLabel(pximds);
totalNumberOfPixels = sum(tbl.PixelCount);
frequency = tbl.PixelCount / totalNumberOfPixels;
classWeights = 1./frequency;
pxLayer = pixelClassificationLayer('Name','labels','Classes',...
tbl.Name,'ClassWeights',classWeights);
lgraph = replaceLayer(lgraph,"classification",pxLayer);
options = trainingOptions('sgdm','MaxEpochs',30, ...
'MiniBatchSize',8, 'Plots','training-progress');
[net, info]= trainNetwork(pximds,lgraph,options);
```

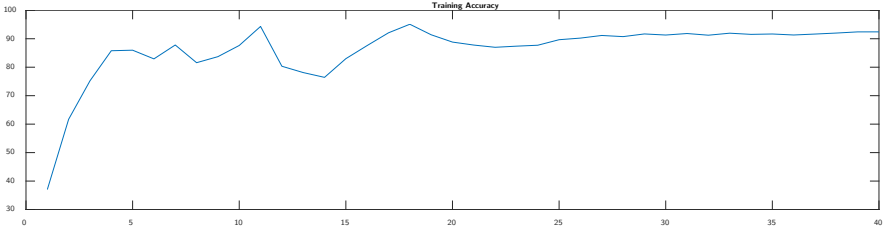


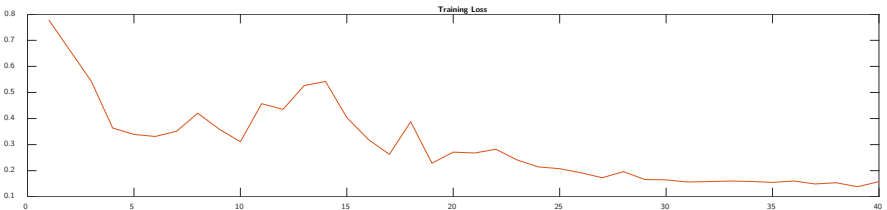


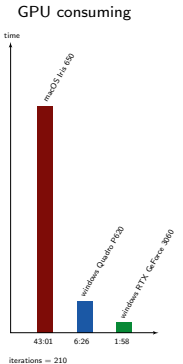
outputSize



Training

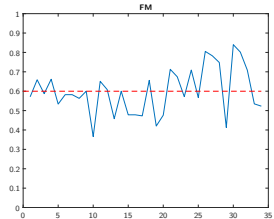

Training Accuracy
100
90
80
70
60
50
40
30
0
5
10
15
20
25
30
35
40


Training Loss
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
5
10
15
20
25
30
35
40

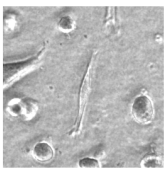
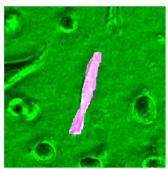

GPU consuming
time
macOS Iris 690
windows Quadro P630
windows RTX GeForce 3060
43:01
6:26
1:58
iterations = 210

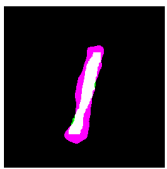
Application

```
for I = 1:length(f_test)
  testImage=imread([strcat(dataPath,'/FRAME_TEST_SEG/'),f_test(I).name]);
  C_test = semanticseg(testImage,net);
  D=C_test=='B';
  GTImage=imread([strcat(dataPath,'/GT_TEST/'),gt_train(I).name]);
  [TP,FP,FN,CR,CM,FM_test(I)]=evaluation_segmentation(...
  bwareafilt(D,1),GTImage);
  imshowpair(testImage,bwareafilt(D,1),'montage');
  pause(0.5); drawnow;
  clear C_test D testImage;
end
```

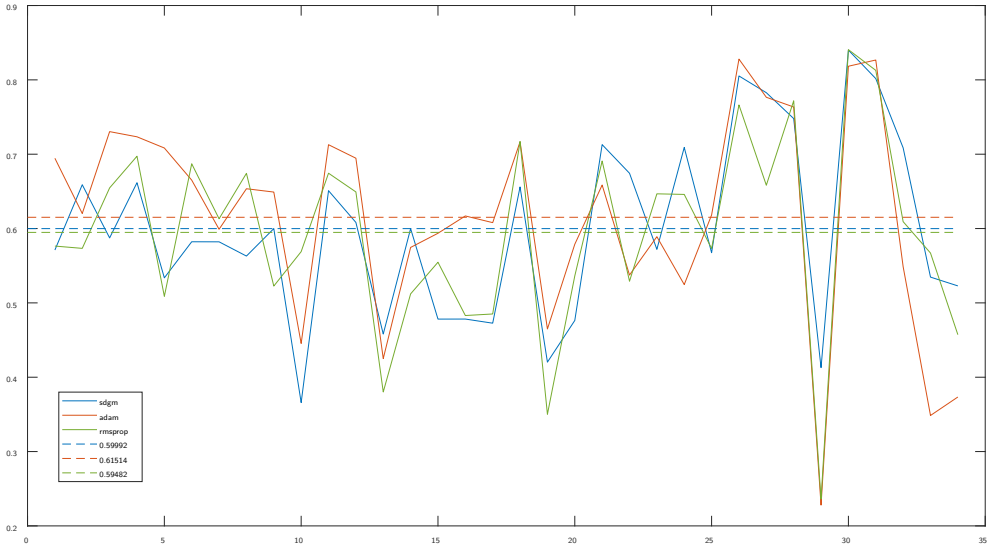


n. 25:

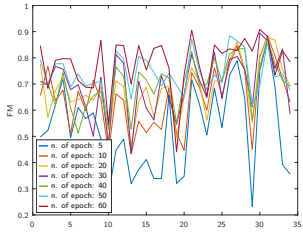


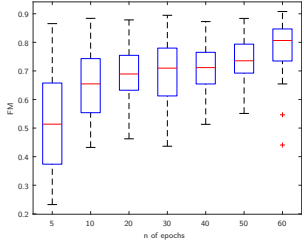
Solver


0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
5
10
15
20
25
30
35
sdgm
adam
rmsprop
0.59992
0.61514
0.59482

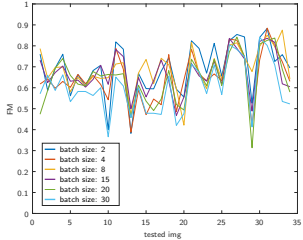
Training options

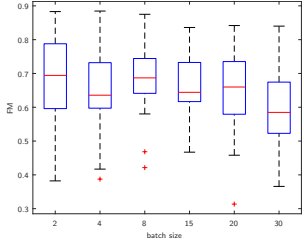
numbers of epochs



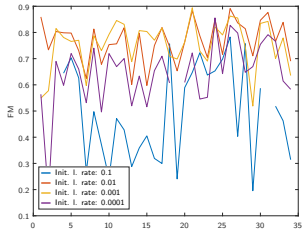


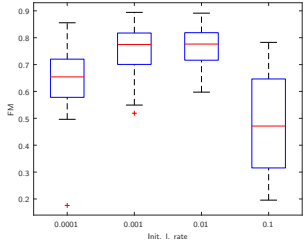
mini batch size





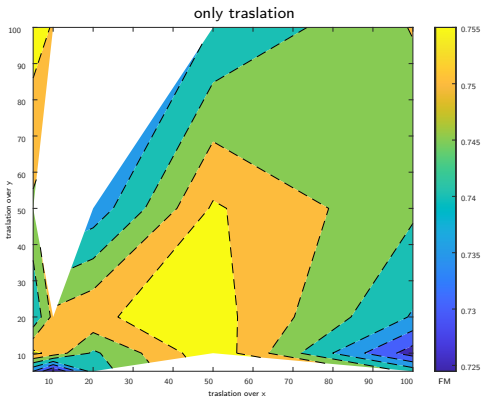
initial learning rate



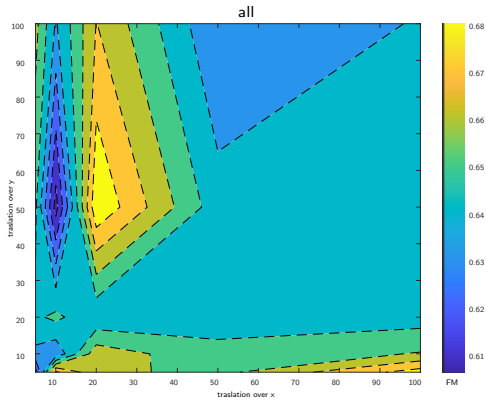


Augmenter

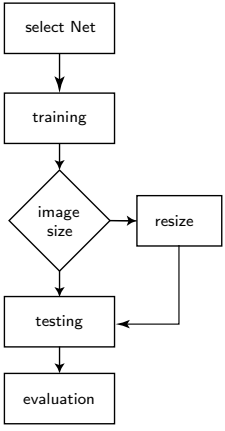
only rotation
mean: 0.4659

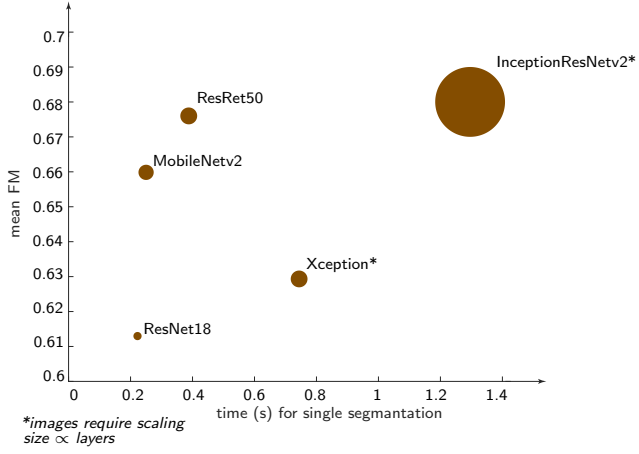


only reflection
mean: 0.7989



Pretrained networks



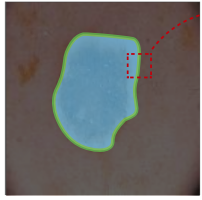


images require scaling
size ∝ layers

```
clear pximds Igraph
[pximds,Igraph]=prepareMyNet(net,'netName',imds,pxds);
[net_net,info_net,FM_test_net,compTime_net]=trainAndTest(pximds,Igraph,...
dataPath,f_test,gt_train);
[accuracy_net,loss_net,FM_mean_net]=figureAccAndLoss(info_net,FM_test_net)
```

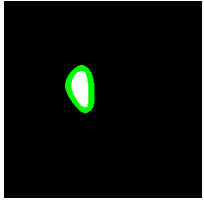
Border identification

NET



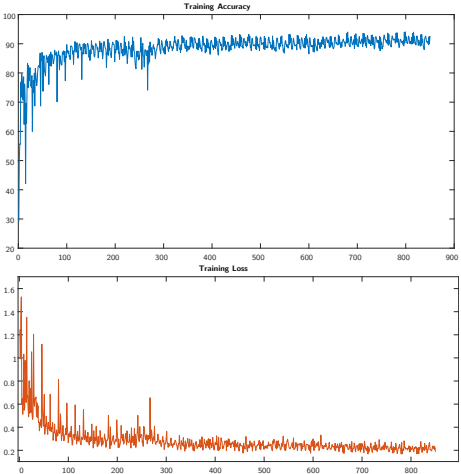
INSIDE	INSIDE	BORDER	BORDER	BACK
INSIDE	INSIDE	BORDER	BORDER	BACK
INSIDE	INSIDE	BORDER	BORDER	BACK
INSIDE	INSIDE	BORDER	BORDER	BACK
INSIDE	INSIDE	BORDER	BORDER	BACK
INSIDE	INSIDE	BORDER	BORDER	BACK
INSIDE	INSIDE	BORDER	BORDER	BACK
INSIDE	INSIDE	BORDER	BORDER	BACK
INSIDE	INSIDE	BORDER	BORDER	BACK
INSIDE	INSIDE	BORDER	BORDER	BACK
INSIDE	INSIDE	BORDER	BORDER	BACK
INSIDE	INSIDE	BORDER	BORDER	BACK

```
pathImage = strcat(newPath,'dataset\mole\Immagini');
pathGT=strcat(newPath,'dataset\mole\Segmentazioni');
pathExport=strcat(newPath,'dataset\mole\PROCESSED\');
[imds,pxds]=optimizeDataset(pathImage,pathGT,pathExport,8,imageSize);
%
imdsTEST = imageDatastore(strcat(newPath,'dataset\mole\TEST\Immagini'));
```



```
temp_GT=imread(string(startingGT.Files(i)));
temp_INT=imerode(temp_GT,SE);
temp_INT=imbinarize(temp_INT);
temp_GT=imbinarize(temp_GT);
temp_GT=uint8(temp_GT);
temp_INT=uint8(temp_INT);
imshowpair(255*temp_GT,255*temp_INT)
GT=temp_GT+temp_INT;
GT=imresize(GT,imageSize(1:2));
pathSplit=strsplit(string(startingGT.Files(i)),'\');
imwrite(GT,strcat(pathExport,'GT\',pathSplit(end)))
```

Training




```
dsTrain = transform(dsTrain,...
@(data)augmentImageAndLabel(data,xT,yT));
```

```
function data = augmentImageAndLabel(data,xT,yT)
for i = 1:size(data,1)
tform = randomAffine2d('Rotation',[0 360],...
'XReflection',true,'XTranslation', xTrans, ...
'YTranslation', yTrans);
rout = affineOutputView(size(data{i,1}), tform,...
'BoundsStyle','centerOutput');
data{i,1} = imwarp(data{i,1},tform,'OutputView',rout);
data{i,2} = imwarp(data{i,2},tform,'OutputView',rout);
end
end
```

```
[net, info]= trainNetwork(dsTrain,lgraph,options);
```

Segmentation