

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

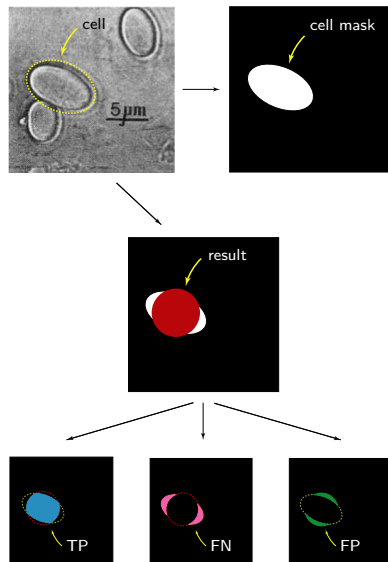
Image segmentation

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

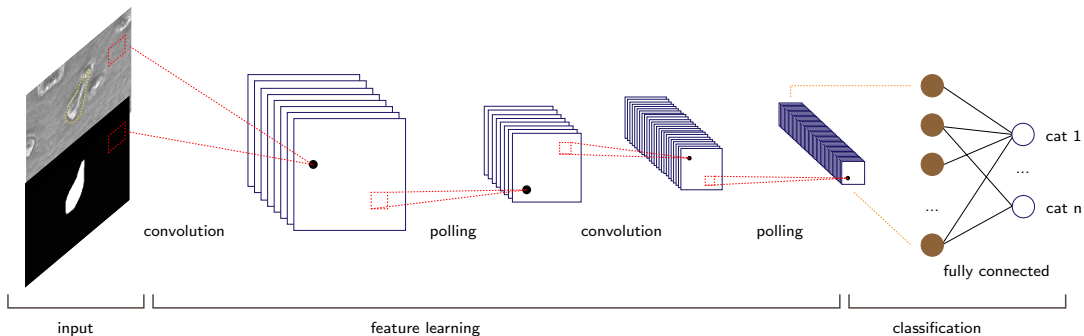
$$CM = \frac{TP}{TP+FN} = \frac{TP}{\text{Total area in GT}}$$

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

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



Transfer learning

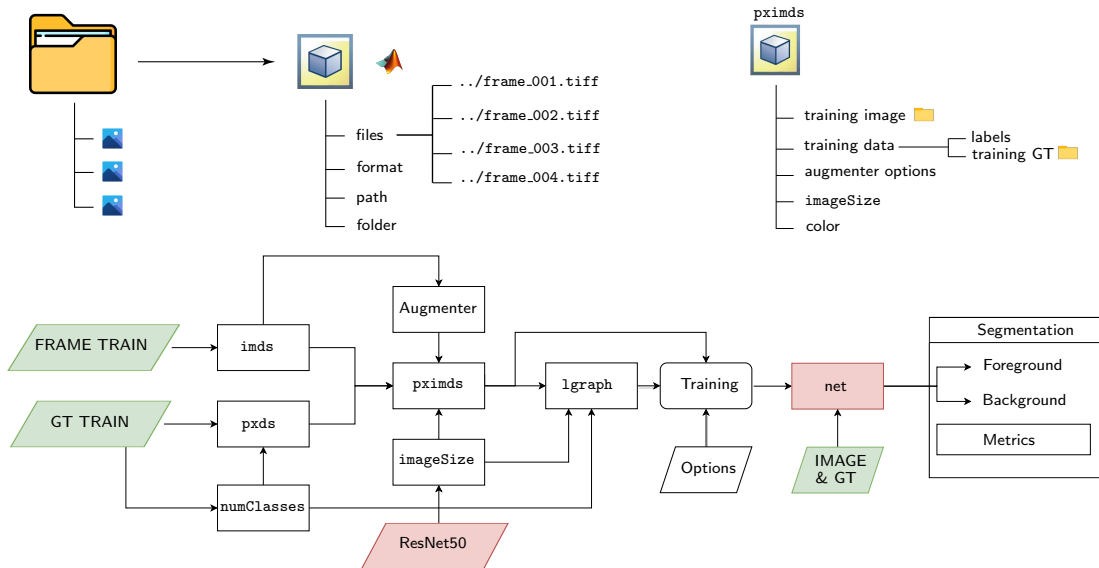


$\{B\}==1$
 $\{N\}==0$

```
1 pxds=pixelLabelDatastore(...  
2 strcat(newPath, 'dataset\GT-TRAIN') ,...  
3 ["N", "B"], [0 1]);
```

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

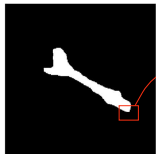
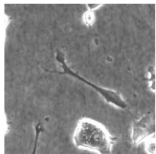
Training dataset



Classification layer

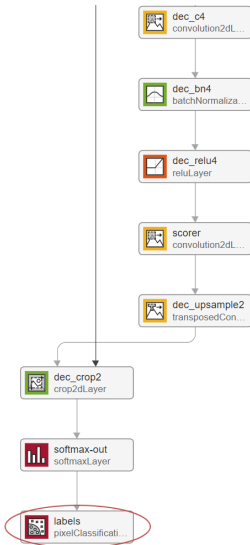
```

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

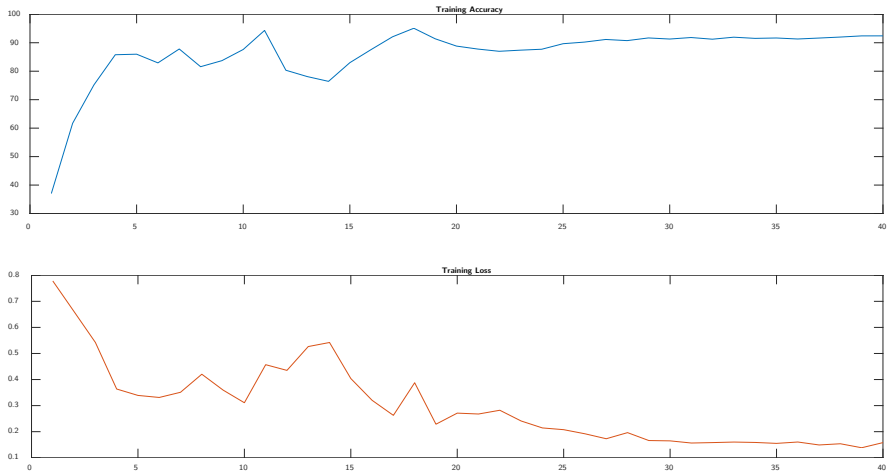


outputSize

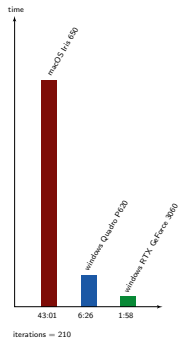
B	B	B	B	B	B	B	B	B
B	B	B	B	B	B	B	B	B
B	B	B	B	B	B	B	B	B
N	B	B	B	B	B	B	B	B
N	N	B	B	B	B	B	B	B
N	N	N	B	B	B	B	B	B
N	N	N	N	B	B	B	B	B
N	N	N	N	N	B	B	B	B
N	N	N	N	N	N	B	B	B
N	N	N	N	N	N	N	B	B
N	N	N	N	N	N	N	N	N
N	N	N	N	N	N	N	N	N
N	N	N	N	N	N	N	N	N
N	N	N	N	N	N	N	N	N
N	N	N	N	N	N	N	N	N
N	N	N	N	N	N	N	N	N
N	N	N	N	N	N	N	N	N
N	N	N	N	N	N	N	N	N



Training

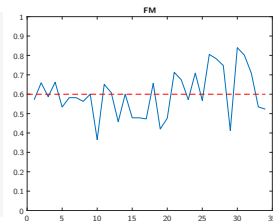


GPU consuming

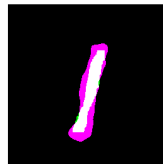
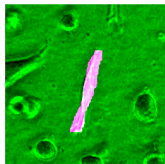
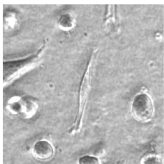


Application

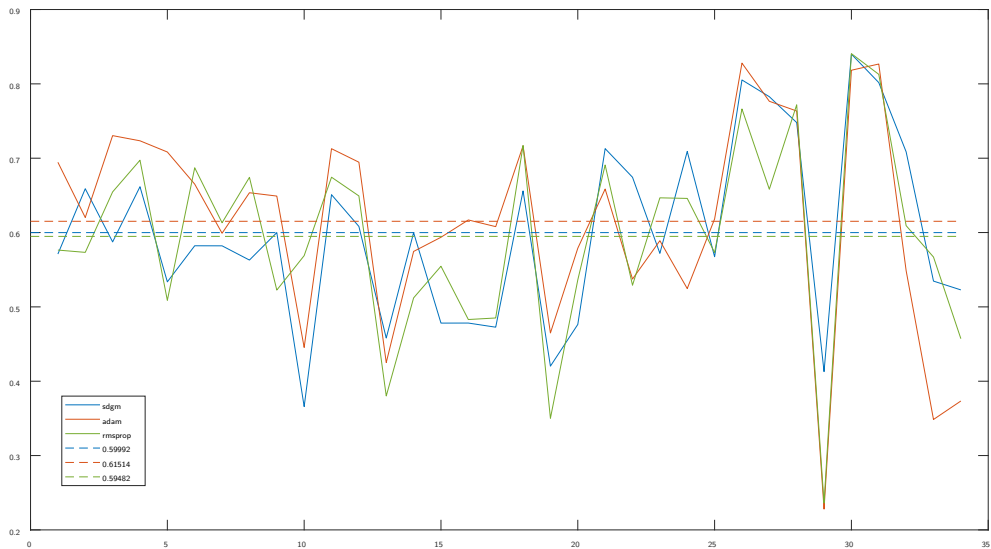
```
1 for l = 1:length(f_test)
2     testImage=imread([strcat(dataPath, '/FRAME_TEST_SEG/'), f_test(l).name]);
3     C_test = semanticseg(testImage, net);
4     D=C_test=='B';
5     GTImage=imread([strcat(dataPath, '/GT_TEST/'), gt_train(l).name]);
6     [TP, FP, FN, CR, CM, FM_test(l)]=evaluation_segmentation(...
7     bwareafilt(D,1), GTImage);
8     imshowpair(testImage, bwareafilt(D,1), 'montage');
9     pause(0.5); drawnow;
10    clear C_test D testImage;
11 end
```



n. 25:

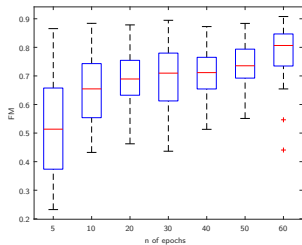
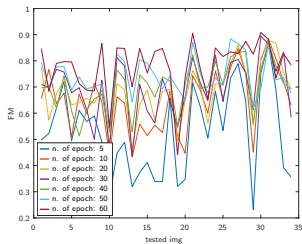


Solver

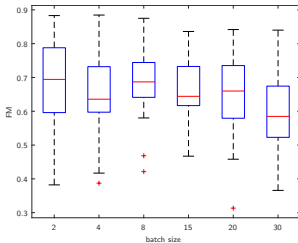
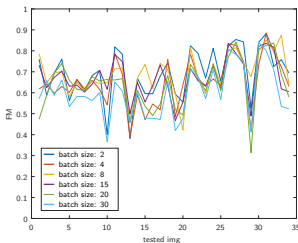


Training options

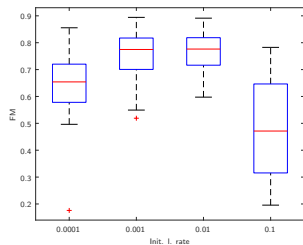
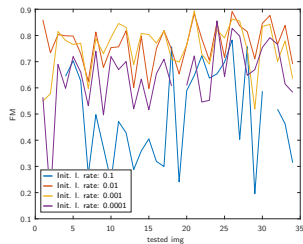
numbers of epochs



mini batch size



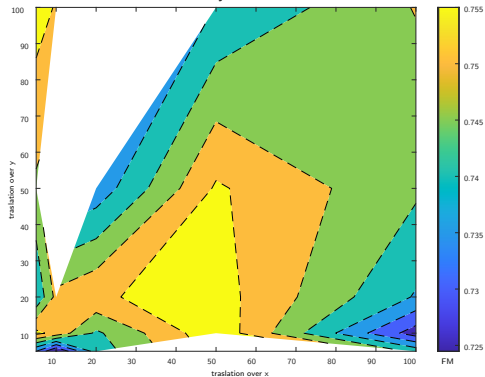
initial learning rate



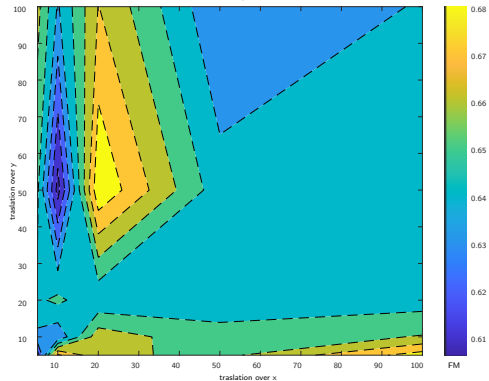
only rotation
mean: 0.4659

only reflection
mean: 0.7989

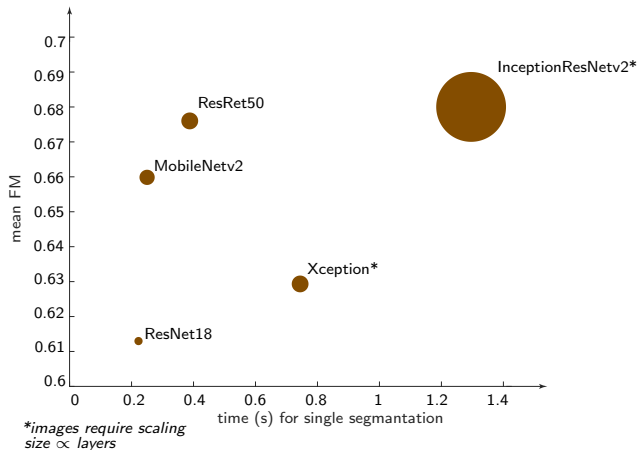
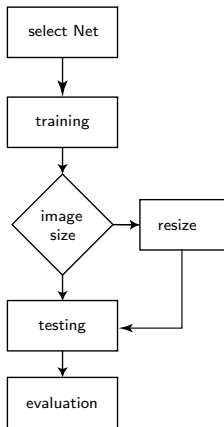
only traslation



all



Pretrained networks

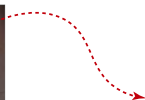
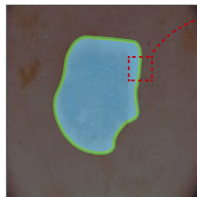


```
1 clear pximds lgraph
2 [pximds, lgraph]=prepareMyNet( net , 'netName' , imds , pxds );
3 [net_net , info_net , FM_test_net , compTime_net]=trainAndTest( pximds , lgraph , ...
4 dataPath , f_test , gt_train );
5 [accuracy_net , loss_net , FM_mean_net]=figureAccAndLoss( info_net , FM_test_net )
```

Border identification



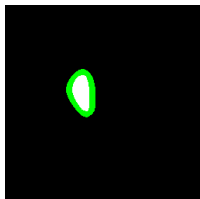
NET



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

```

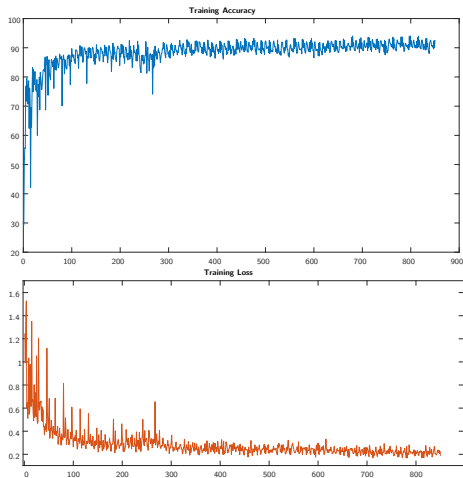
1 pathImage = strcat(newPath, 'dataset\mole\Immagini');
2 pathGT=strcat(newPath, 'dataset\mole\Segmentazioni');
3 pathExport=strcat(newPath, 'dataset\mole\PROCESSED\');
4 [imds, pxds]=optimizeDataset(pathImage, pathGT, pathExport, 8, imageSize);
5 %
6 imdsTEST = imageDatastore(strcat(newPath, 'dataset\mole\TEST\Immagini'));
    
```



```

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

Training



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

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

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

Segmentation

