Building Convolutional Neural Networks under the Expert's Control

Alexandre Xavier Falcão

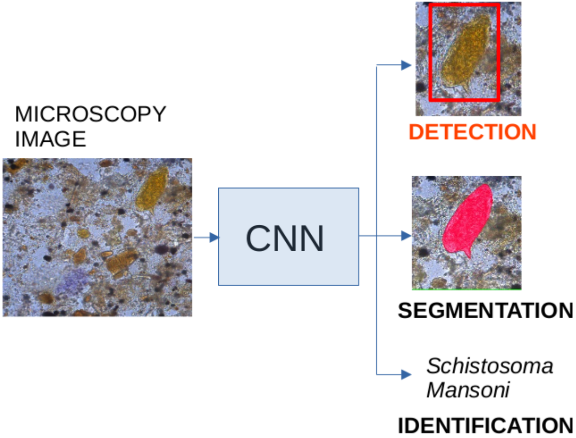
LIDS - IC - UNICAMP

afalcao@ic.unicamp.br

Invited by Laurent Najman (Labex Bézout)

Motivation

Convolutional neural networks (CNNs) can effectively solve problems such as detection, segmentation, and identification.



However, the price is usually a substantial human effort in data annotation and network adaptation.

Research goals

We wish to answer questions such as:

- R1 How to build a CNN layer by layer under the expert's control?
- R2 What is the simplest model for a given problem?
- R3 How can it be trained with minimum human effort?
- R4 Can we explain its decisions?

Research goals

We wish to answer questions such as:

R1 How to build a CNN layer by layer under the expert's control?

R2 What is the simplest model for a given problem?

R3 How can it be trained with minimum human effort?

R4 Can we explain its decisions?

This talk shows how to build a CNN layer by layer for object detection without backpropagation.

Agenda

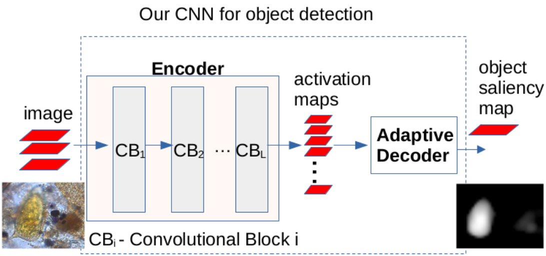
- How is our CNN for object detection?
- What are the expert's actions?
- How does training work?
- What are our most recent results?

How is our CNN for object detection?

It is composed of

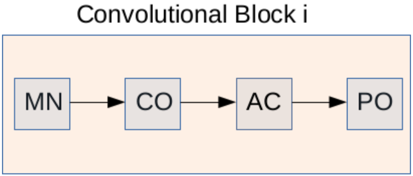
- an encoder for feature extraction: a sequence of convolutional blocks that creates intermediary and final activation maps.

- an adaptive decoder for object saliency estimation: a point-wise convolution followed by activation, whose weights change with the input image.



What is a convolutional block?

A convolutional block may contain several operations, such as normalization, convolution, activation, skip connection, and pooling.

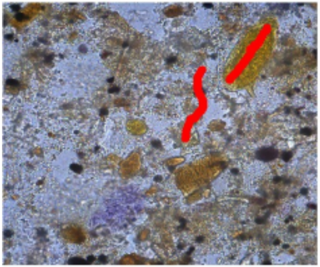
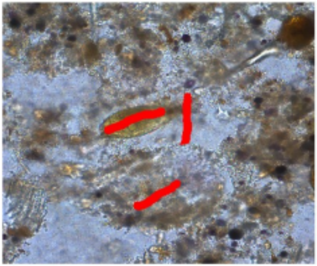


MN – marker-based normalization
CO – convolution with a filter bank
AC – activation (ReLU)
PO – pooling (max/average pool.)

We have adopted the above sequence of operations.

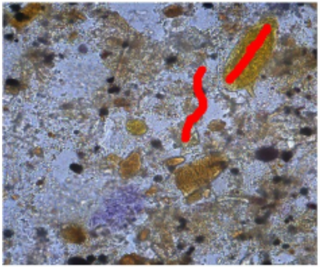
What are the expert's actions?

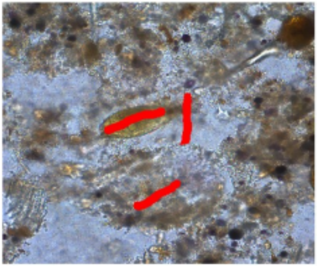
The expert draws markers on discriminative regions of a few representative images as input.

What are the expert's actions?

The expert draws markers on discriminative regions of a few representative images as input.

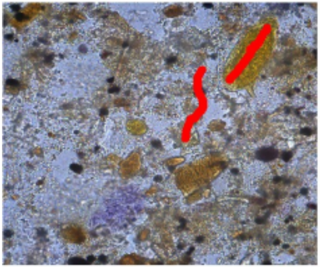


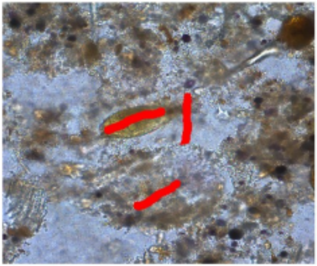


- Convolutional filters can be automatically estimated block by block from this input for a given encoder architecture.

What are the expert's actions?

The expert draws markers on discriminative regions of a few representative images as input.

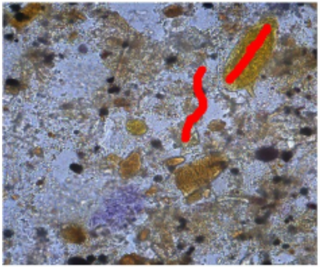


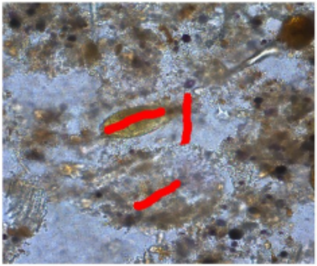


- Convolutional filters can be automatically estimated block by block from this input for a given encoder architecture.
- The methodology is called FLIM: Feature Learning from Image Markers [GRSL20, SIBGRAPI20, NEURIPS20, EMBC21, GRSL22, ARXIV23, SIPAIM23].

What are the expert's actions?

The expert draws markers on discriminative regions of a few representative images as input.





- Convolutional filters can be automatically estimated block by block from this input for a given encoder architecture.
- The methodology is called FLIM: Feature Learning from Image Markers [GRSL20, SIBGRAPI20, NEURIPS20, EMBC21, GRSL22, ARXIV23, SIPAIM23].
- The expert may intervene by adding/removing markers, eliminating filters, or selecting more images.

How does training work?

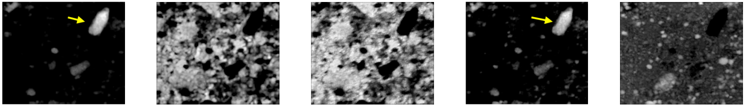
From image patches centered at marker pixels, we wish to

- identify groups of patches that represent patterns of interest in the images,
- estimate one filter **F** per group, such that the convolution between **F** and an image can
  - activate regions whose patterns are similar to the ones represented by **F** and
  - deactivate image regions with dissimilar patterns.

How does training work?

From image patches centered at marker pixels, we wish to

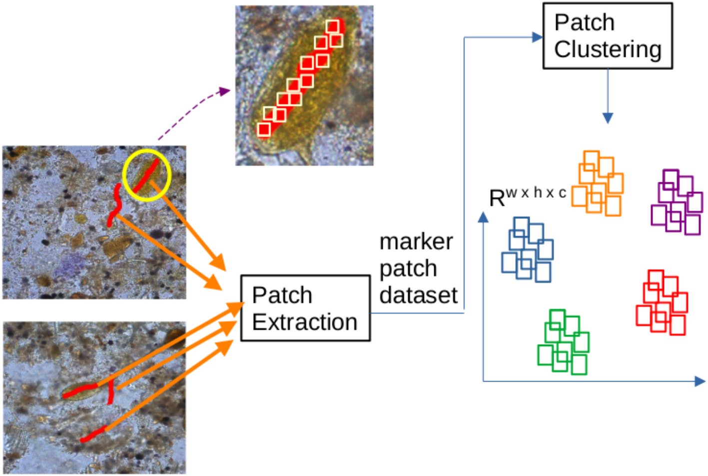
- identify groups of patches that represent patterns of interest in the images,
- estimate one filter **F** per group, such that the convolution between **F** and an image can
  - activate regions whose patterns are similar to the ones represented by **F** and
  - deactivate image regions with dissimilar patterns.



\+ - - + -

How does training work?

A patch $\mathbf{X}(p) \in \Re^{w \times h \times c}$ at a pixel p with width w, height h, and c channels is a local feature vector of size $n = w \times h \times c$.



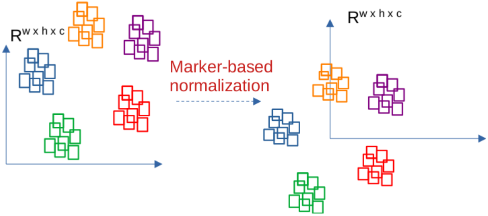
Patches centered at marker pixels are grouped into a given number of clusters.

How does training work?

Patches $\mathbf{X}(p) = (X_1(p), X_2(p), \ldots, X_n(p))$ in the marker patch dataset $\mathcal{X}$ are normalized as $\mathbf{Z}(p) = (Z_1(p), Z_2(p), \ldots, Z_n(p))$, where

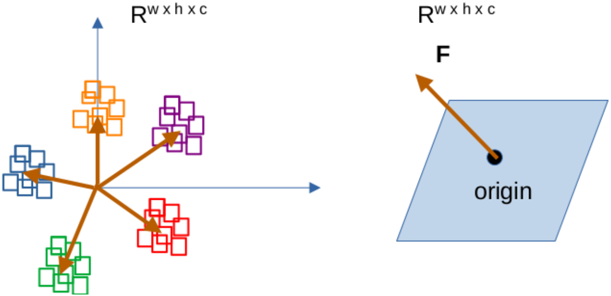
$$
\begin{aligned}
Z_i(p) &= \frac{X_i(p) - \mu_i}{\sigma_i + \epsilon}, \\
\mu_i &= \frac{1}{|\mathcal{X}|} \sum_{X(p) \in \mathcal{X}} X_i(p), \\
\sigma_i^2 &= \frac{1}{|\mathcal{X}|} \sum_{X(p) \in \mathcal{X}} (X_i(p) - \mu_i)^2,
\end{aligned}
$$

$i = 1, 2, \ldots, n$ and a very small $\epsilon > 0$



How does training work?

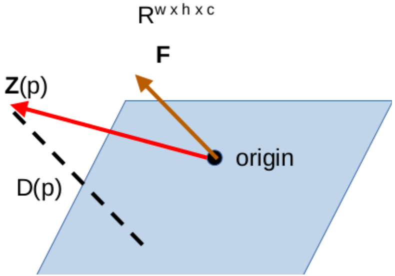
The filters $\mathbf{F} \in \Re^{w \times h \times c}$ are the cluster centers.



Each filter $\mathbf{F}$ is orthogonal to a hyperplane in $\Re^{w \times h \times c}$ and marker-based normalization aims to isolate each cluster in the positive side of the corresponding hyperplane.

How does training work?

The convolution $\hat{I} * \mathbf{F}$ between an image $\hat{I}$ with marker-based normalized patches $\mathbf{Z}(p)$ and $\mathbf{F}$ outputs an image $\hat{D}$ with pixel values $D(p) = \langle \mathbf{Z}(p), \mathbf{F} \rangle$.



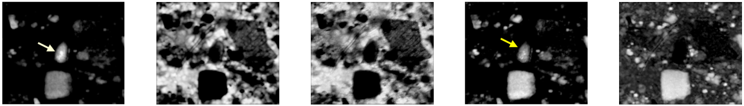
ReLU activation eliminates negative values of $D(p)$, and max-pooling aggregates positive activations within a given neighborhood of each pixel.

How does training work?

- The exact process is repeated for each convolutional block using the markers mapped onto the activation maps of the previous block.

How does training work?

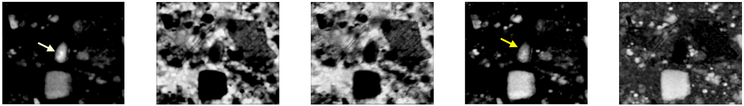
- The exact process is repeated for each convolutional block using the markers mapped onto the activation maps of the previous block.
- The expert may examine the activation maps of other images, remove redundant filters, or draw markers in new images.



\+ - - + -

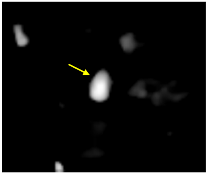
How does training work?

- The exact process is repeated for each convolutional block using the markers mapped onto the activation maps of the previous block.
- The expert may examine the activation maps of other images, remove redundant filters, or draw markers in new images.



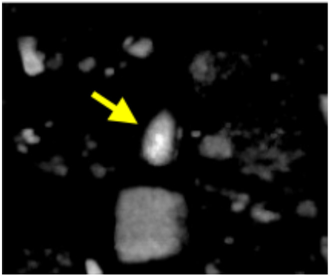
$+$ -   - $+$ -

- The adaptive decoder may be used to visualize the object saliency maps at the output of any block.

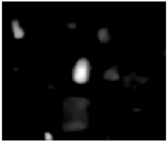


How does training work?

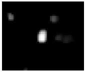
As “deeper” the encoder is, the expert should observe positive activation maps varying from suitable for object delineation to object detection.



block 1



block 2



block 3

block 4

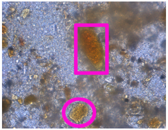
How does training work?

The adaptive decoder is a point-wise convolution (weighted average of the activation maps) followed by ReLU activation.

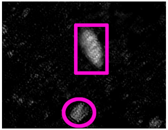
block 1 block 2 block 3 block 4

Positive activations are assigned to weight $w = 1$ and negative activations to weight -1 according to an adaptation function.

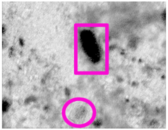
The underlying idea is to use negative activations to suppress false positives from the positive activations.



Image



Foreground filter

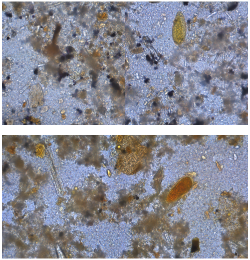


Background filter

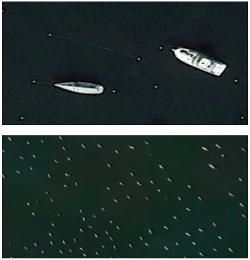
The weights are unsupervised and change with the input image – a concept never exploited in CNNs.

What are the most recent results?

We have evaluated our CNNs for ship detection in aerial images and parasite egg detection in microscopy images [ARXIV2023].



Parasite Eggs

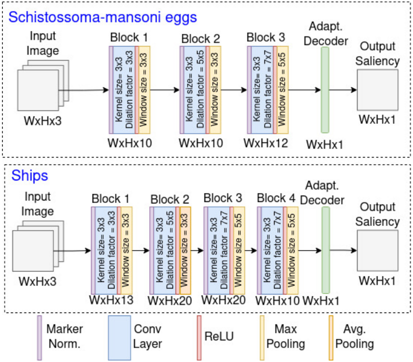


Ships

Ships appear in very different scales.

What are the most recent results?

We selected five images (1%) for training and 495 (79%) to find the best network architecture. The test set was fixed with 125 (20%) of the images.



What are the most recent results?

We used three baselines for comparison.

- A state-of-the-art saliency detection method (U^2Net[PR20]).
- A few-shot object detection approach (DETReg[CVPR22]).
- A few-shot salient object detector (SelfReformer[ARXIV22]).

What are the most recent results?

We used three metrics for evaluation.

- F-score: the harmonic mean between precision (P) and recall (R).
- Average Precision (AP): the area under the PR-curve (AUC) up to a given intersection-over-union (IoU) threshold.
- μAP: the mean of AP for IoU thresholds from 0.50 to 0.95.

Two IoU thresholds, 0.5 and 0.75, were used to define positive detections in F-score and AP.

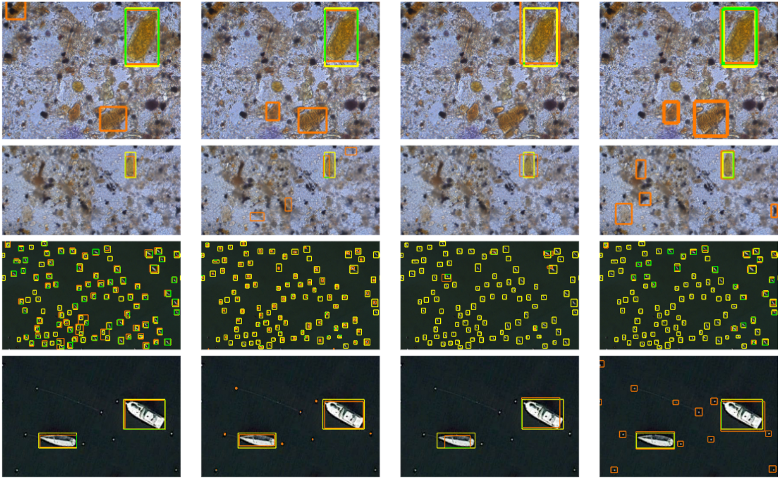
What are the most recent results?

The best and second-best results are in blue and green, respectively – our method is in bold.

Schistossoma Eggs	$F_2^{0.5}$	$AP^{0.5}$	$F_2^{0.75}$	$AP^{0.75}$	μAP
DETReg	0.634	0.279	0.421	0.146	0.155
U²Net	0.740	0.531	0.609	0.405	0.335
Self-Reformer	0.747	0.688	0.114	0.024	0.227
Adaptive-FLIM$_p$	**0.799**	**0.929**	**0.630**	**0.488**	**0.525**
Ships	$F_2^{0.5}$	$AP^{0.5}$	$F_2^{0.75}$	$AP^{0.75}$	μAP
DETReg	0.261	0.183	0.235	0.130	0.126
U²Net	0.371	0.366	0.164	0.161	0.169
Self-Reformer	0.251	0.219	0.122	0.105	0.109
Adaptive-FLIM$_s$	**0.393**	**0.322**	**0.210**	**0.145**	**0.150**

The higher the IoU, the higher the required quality of the bounding boxes.

What are the most recent results?



(a) **Adaptive-FLIM** (b) U²Net (c) Self-Reformer (d) DETReg

Prediction, Ground Truth, and Intersection.

- FLIM with an adaptive decoder introduces a new way to train CNNs without backpropagation.
- The method allows building flyweight CNNs for object detection that are thousands of times lighter than the baselines.
- One can devise different methods to select images, suggest regions for marker drawing, estimate filters from markers, guide the expert's actions, and evaluate network architectures.
- We are investigating the above topics towards a tool for the design of CNNs from image markers.

Acknowledgements

Merci

FAPESP, CNPq, CAPES, and collaborators (Leonardo João, Azael Sousa, Bianca dos Santos, Jancarlo Gomes, Silvio Guimarães, and Ewa Kijak).

Available codes:
`https://github.com/LIDS-UNICAMP/FLIM`
`https://github.com/LIDS-UNICAMP/ift`
`https://github.com/LIDS-UNICAMP/FLIM-Builder`

References

GRSL20 I. de Souza and A.X. Falcão. Learning CNN filters from user-drawn image markers for coconut-tree image classification. IEEE Geoscience and Remote Sensing Letters, doi: 10.1109/LGRS.2020.3020098, 2020, `https://arxiv.org/pdf/2008.03549.pdf`.

SIBGRAPI20 I.E. de Souza, B.C. Benato, and A.X. Falcão. Feature Learning from Image Markers for Object Delineation. *33rd SIBGRAPI*, doi: 10.1109/SIBGRAPI51738.2020.00024, 2020.

NEURIPS20 I.E. de Souza, B.C. Benato, F.L. Galcão and A.X. Falcão. Convolutional Neural Networks from Image Markers. Beyond BackPropagation: Novel Ideas for Training Neural Architectures, Neurips Workshop, 2020, `https://arxiv.org/pdf/2012.12108.pdf`.

EMBC21 A.M. Sousa, F. Reis, R. Zerbini, J.L.D. Comba, and A.X. Falcão. CNN Filter Learning from Drawn Markers for the Detection of Suggestive Signs of COVID-19 in CT Images. *43rd EMBC*, doi: 10.1109/EMBC46164.2021.9629806, 2021.

GRSL22 I.E. de Souza, C.L. Cazarin, M.R. Veronez, L. Gonzaga Jr, and A.X. Falcão. User-guided data expansion modeling to train deep neural networks with little supervision. *IEEE Geoscience and Remote Sensing Letters*, doi: 10.1109/LGRS.2022.3201437, 2022.

ARXIV23 L. Joao, A. Sousa, B. dos Santos, S. Guimaraes, J. Gomes, E. Kijak, and A. Falcao. Building Flyweight FLIM-based CNNs with Adaptive Decoding for Object Detection, arXiv 2306.14840, 2023.

SIPAIM23 M.A. Cerqueira, F. Sprenger, B.C.A. Teixeira, and A.X. Falcão. Building Brain Tumor Segmentation Networks with User-Assisted Filter Estimation and Selection. *18th SIPAIM*, doi 10.1117/12.2669770, 2023.

PR20 X. Qin, Z. Zhang, C. Huang, M. Dehghan, O. R. Zaiane, and M. Jagersand. U2-net: Going deeper with nested u-structure for salient object detection. *Pattern Recognition*, 2020.

ARXIV22 Y. K. Yun and W. Lin. Selfreformer: Self-refined network with transformer for salient object detection. arXiv preprint arXiv:2205.11283, 2022.

CVPR22 A. Bar, X. Wang, V. Kantorov, C. J. Reed, R. Herzig, G. Chechik, A. Rohrbach, T. Darrell, and A. Globerson. Detreg: Unsupervised pretraining with region priors for object detection. *Conference on Computer Vision and Pattern Recognition (CVPR)*, doi 10.1109/CVPR52688.2022.01420, 2022