

A deep learning approach to superpixels segmentation

Théo Dumont – Tutor: Bruno Figliuzzi

Center for Mathematical Morphology
Mines ParisTech - PSL Research University

June, 19th, 2020

- 1** Introduction
 - Segmentation
 - Superpixels
 - Superpixels segmentation
 - Ambitions
- 2** Dataset generation
 - Eikonal
 - Training
- 3** Network Architecture
 - Model
 - Loss function
- 4** Experiments and results
 - Hyperparameters
 - Results
- 5** Discussion



(a) *The original image*



(b) *The segmented image*

Figure: An image (a) and its segmented image (b).

- **context:** image perception
- **idea:** partitioning an image into groups of similar pixels



Figure: Example of a superpixel segmentation. From left to right: original image, original image with its calculated superpixels outlines and resulting superpixel segmentation in average color representation.

- **idea:** over-segmentation of an image
- the superpixels segmentation is *not unique*

- **Boundary recall:** ratio of the number of true positive contour pixels and of the number of contour pixels in the ground truth segmentation, with a tolerance of 2 pixels
- **Compactness:**

$$\text{COMP}(S_i) = \sum_{S_i} \frac{|S_i|}{N} \frac{4\pi A(S_i)}{P(S_i)^2}$$

- **Undersegmentation error:** Leakage of a superpixel overlapping with a ground truth segment:

$$\text{UE}_{NP}(G, S) = \frac{1}{N} \sum_{G_i} \sum_{S_j \cap G_i \neq \emptyset} \min \{ |S_j \cap G_i|, |S_j - S_j \cap G_i| \}$$

where S_j is a superpixel and G_i is a ground truth segment.

Undersegmentation Error

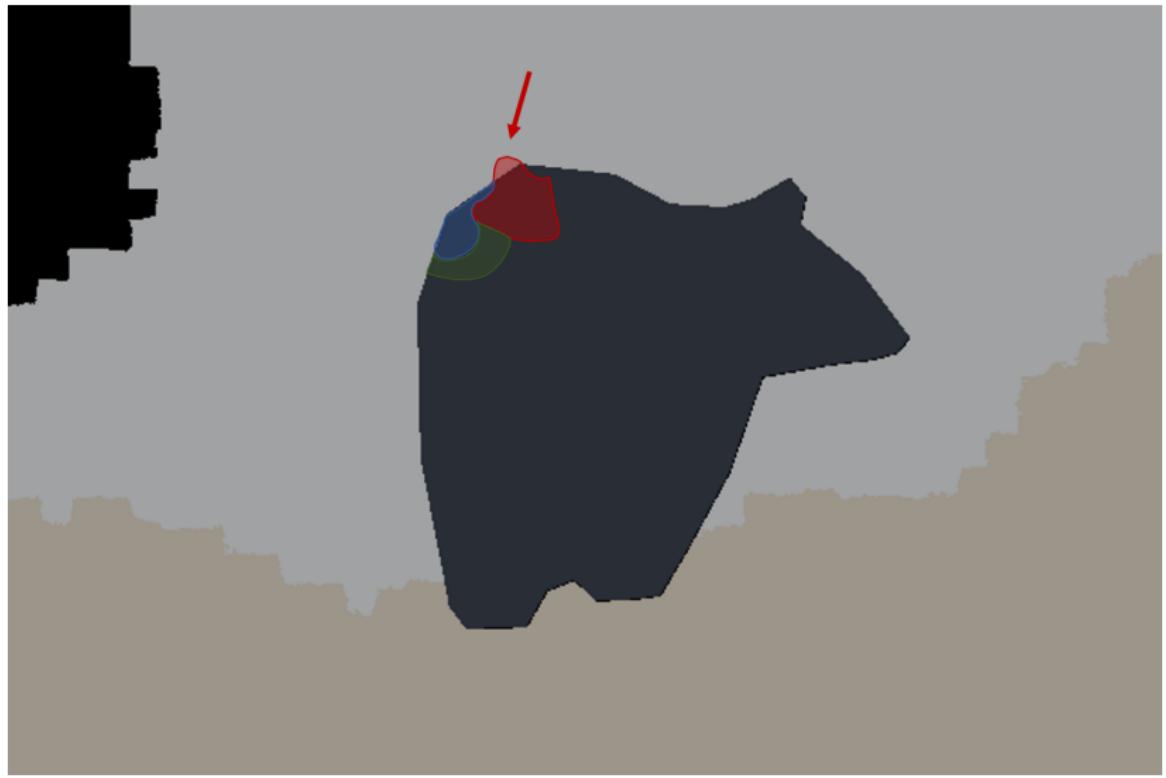


Figure: Undersegmentation illustration.

- interest of deep learning here;
- generating deep learning based over-segmentations? lack of proper training datasets;
- use a CNN to improve SOTA.

- 1** Introduction
 - Segmentation
 - Superpixels
 - Superpixels segmentation
 - Ambitions
- 2** Dataset generation
 - Eikonal
 - Training
- 3** Network Architecture
 - Model
 - Loss function
- 4** Experiments and results
 - Hyperparameters
 - Results
- 5** Discussion

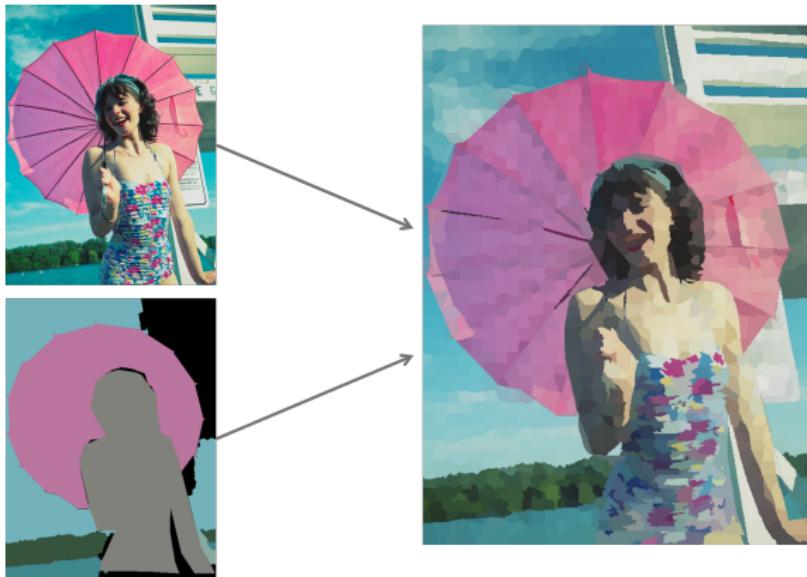


Figure: Eikonal algorithm principle¹.

¹cf. [1], *Fast Marching Based Superpixels Generation*, Kaiwen Chang and Bruno Figliuzzi, 2019.

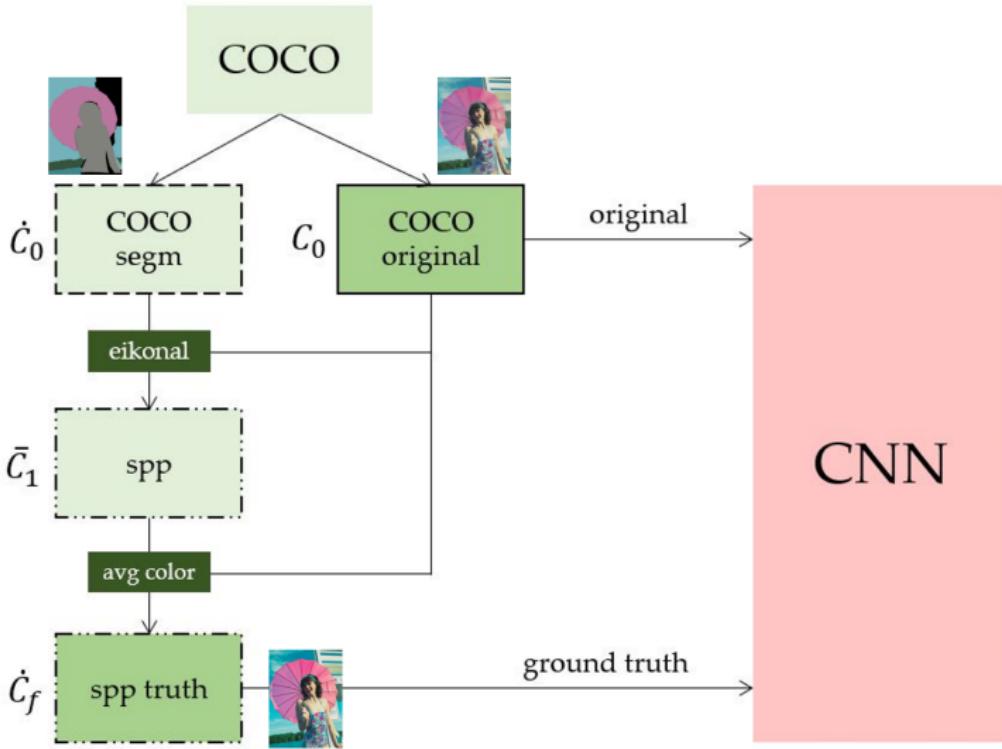


Figure: Training process using the COCO dataset².

²cf. [2], Microsoft COCO: Common Objects in Context, <http://cocodataset.org>, 2014.

Table of content

- 1** Introduction
 - Segmentation
 - Superpixels
 - Superpixels segmentation
 - Ambitions
- 2** Dataset generation
 - Eikonal
 - Training
- 3** Network Architecture
 - Model
 - Loss function
- 4** Experiments and results
 - Hyperparameters
 - Results
- 5** Discussion

Context Aggregation Network (CAN)³

L^1	L^2	$L^d = L^7$
$m \times n \times 3$	\longrightarrow	$m \times n \times 24$

Table: *Layers of the network*

For each layer:

- 1** dilated convolution
- 2** Adaptative Batch Normalization (ABN)
- 3** Leaky Rectified Linear Unit (LReLU)

³cf. [3], *Fast Image Processing with Fully-Convolutional Networks*, Qifeng Chen, Jia Xu and Vladlen Koltun, 2017.

Dilated convolutions

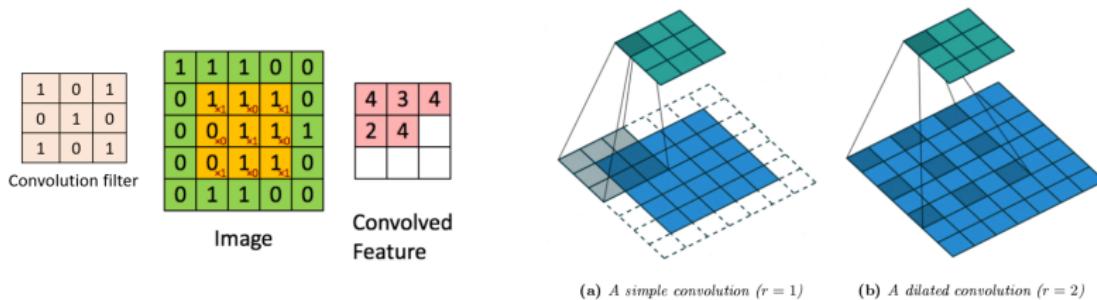


Figure: Simple and dilated convolutions.

- The output $C(x)$ of a pixel x is:

$$\begin{aligned} C(x) &:= (L_j *_r K_{i,j})(x) \\ &= \sum_{a+rb=x} L_j(a)K_{i,j}(b) \\ &= \sum_b L_j(x - rb)K_{i,j}(b) \end{aligned}$$

- exponential growth: global information aggregation

ABN

Adaptive normalization function Ψ :

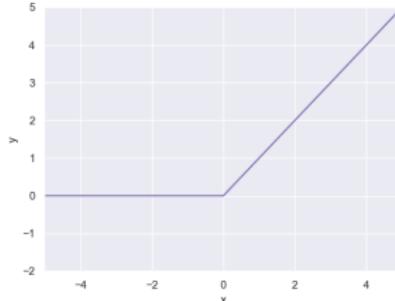
$$\Psi(x) = a \cdot x + b \cdot BN(x),$$

where BN is the classic batch normalization, defined as:

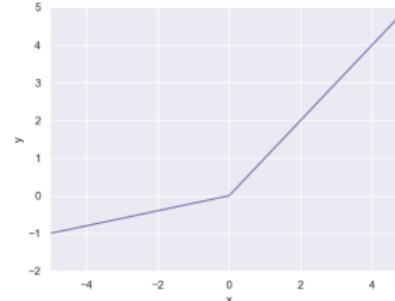
$$BN(x) = \frac{x - E[\text{batch}]}{\sqrt{\text{Var}[\text{batch}] + \epsilon}} * \gamma + \beta.$$

LReLU

$$\Phi(x) = \max(\alpha x, x), \text{ with } 0 < \alpha < 1.$$



(a) ReLU

(b) LReLU ($\alpha = 0.2$)

$$L_i^s = \Phi \left(\Psi^s \left(b_i^s + \sum_j L_j^{s-1} *_{r_s} K_{i,j}^s \right) \right)$$

Layer L_s		1	2	3	4	5	6	7
Conv	Input w_s	3	24	24	24	24	24	24
	Output w_{s+1}	24	24	24	24	24	24	3
	Receptive field	3×3	1×1					
	Dilation r_s	1	2	4	8	16	1	1
	Padding	1	2	4	8	16	1	0
ABN		Yes						
LReLU		0.2	0.2	0.2	0.2	0.2	0.2	No

Table: Our Context Aggregation Network

L_2 Loss

$$L = \frac{1}{N} \sum_{i=1}^N |\hat{f}(I)_i - f(I)_i|^2$$



Figure: A segmented image. Many regions have a gradient equal to 0.

Total Variation (TV) Loss

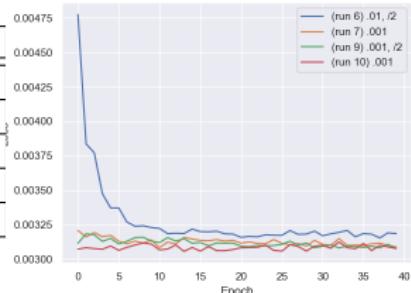
$$L_{TV} = \frac{1}{N} \sum_{i=1}^N |\hat{f}(I)_i - f(I)_i|^2 + \alpha_{TV} \frac{1}{N} \sum_{i=1}^N |(\nabla f(I))_i|$$

- 1** Introduction
 - Segmentation
 - Superpixels
 - Superpixels segmentation
 - Ambitions
- 2** Dataset generation
 - Eikonal
 - Training
- 3** Network Architecture
 - Model
 - Loss function
- 4** Experiments and results
 - Hyperparameters
 - Results
- 5** Discussion

Experiments

id	lr_0	decay	thresh.	start from
6	10^{-2}	$\times .5$ ev. 2 ep.	10^{-4}	run 0 ep.5
7	10^{-3}	No		run 0 ep.40
8	10^{-2}	No		run 0 ep.40
9	10^{-3}	$\times .5$ ev. 2 ep.	10^{-4}	run 0 ep. 40
10	10^{-3}	No		run 9 ep.40

(a) Runs for learning rate tuning, with $d = 7$



(b) Runs 6, 7, 9 and 10

Figure: Tuning of learning rate: validation loss for different runs.

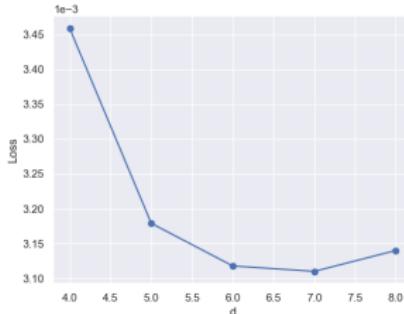


Figure: Tuning of network size: loss values on the validation set.

batch_size	epochs	d	lr_0	decay for lr_0	α_{TV}
32	80	7	10^{-2}	10^{-3} after 10 ep.	0

Table: Hyperparameter values

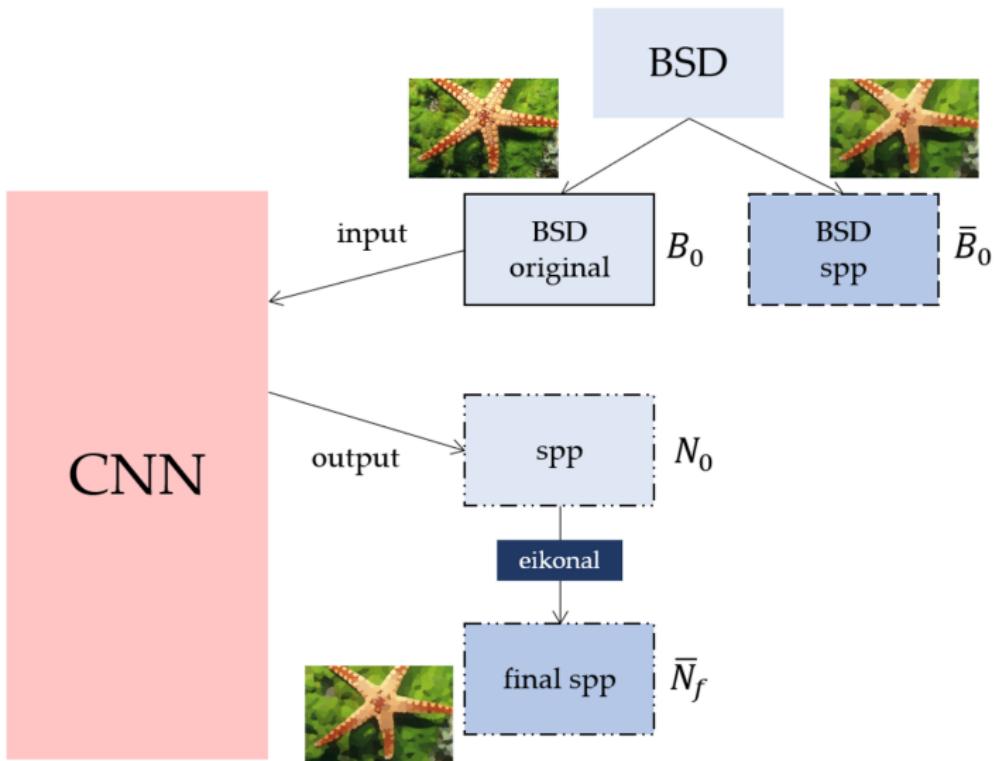
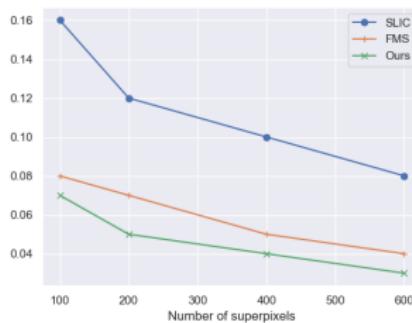
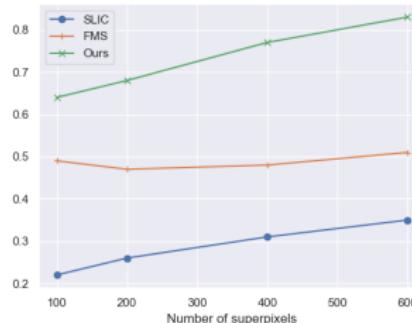


Figure: Testing process.

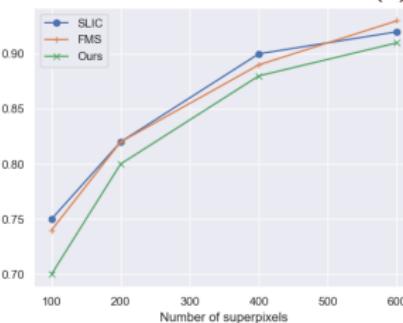
Results



(a) Undersegmentation Error



(b) Compactness



(c) Boundary Recall

Figure: Comparisons of metrics on the BSDS500 dataset⁴

⁴<https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.html>.

- 1** Introduction
 - Segmentation
 - Superpixels
 - Superpixels segmentation
 - Ambitions
- 2** Dataset generation
 - Eikonal
 - Training
- 3** Network Architecture
 - Model
 - Loss function
- 4** Experiments and results
 - Hyperparameters
 - Results
- 5** Discussion

- CNN can achieve good performance on superpixel segmentation
- discussion on TV-loss



- U-Net
- COCO dataset lacks quality
 - 1 train on COCO and finetune on BSD?
 - 2 add gaussians to each superpixel?
 - 3 pre-process image with grid?



- application in research in grainy materials imaging, with indicator of the center of the particules

Essential References

[1], *Fast Marching Based Superpixels Generation*, Kaiwen Chang and Bruno Figliuzzi, 2019

[2], *Microsoft COCO: Common Objects in Context*,
<http://cocodataset.org>, 2014

[3], *Fast Image Processing with Fully-Convolutional Networks*, Qifeng Chen, Jia Xu and Vladlen Koltun, 2017

...

Code: github.com/theodumont/superpixels-segmentation

Thank you very much to M. Bruno Figliuzzi.