BlobNet

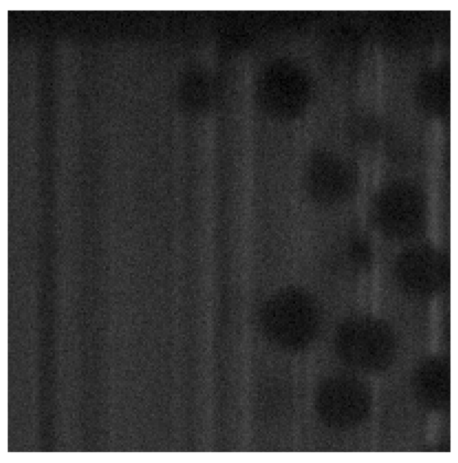
A Convolutional Neural Network to detect particles on images

D. Paulovics
Master Ondes, Atomes, Matière, Université Côte d'Azur

B. Figliuzzi
CMM, Mines ParisTech

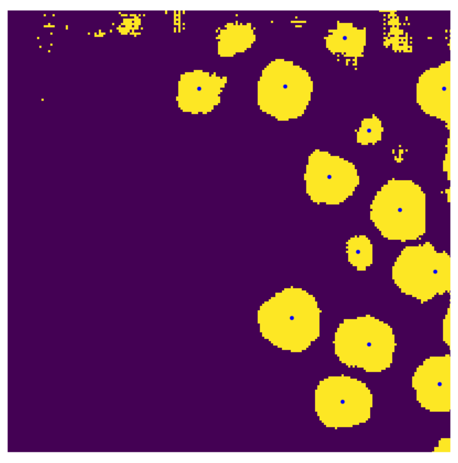
F. Blanc
Inphyni - CNRS - Université Côte d'Azur

1. Better results than classical segmentation



Crop from experimental image

Classical segmentation
(also used as Ground Truth)



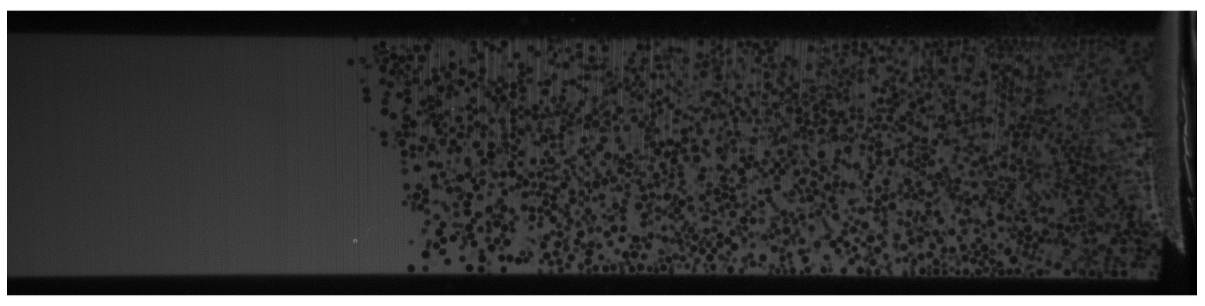
BlobNet segmentation

Objective: Find the positions of the centers of all particles and their corresponding surfaces in experimental images.

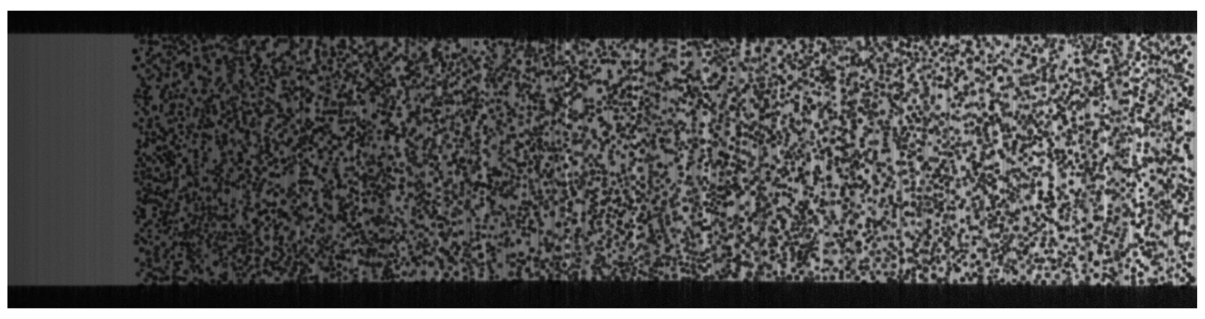
- Classical segmentation using the KMeans algorithm and calculating a distance image.
- Not autonomous, it needs adjustments for at least every different experimental setup.

- Segmentation by BlobNet convolutional neural network that was trained on classically segmented images.
- Produces consistently better segmentation results on validation images than classical segmentation.

2. Training on synthetic images



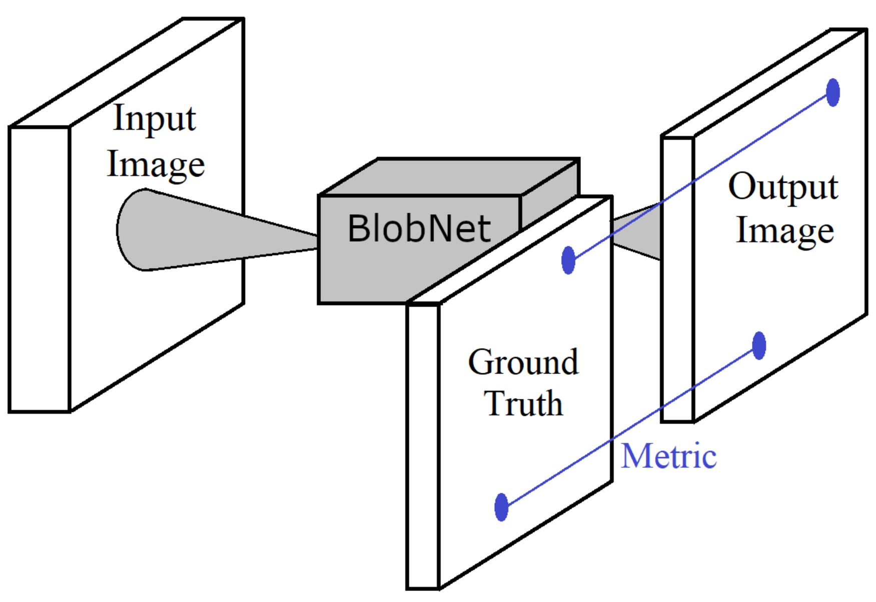
Experimental image



Randomly generated synthetic training image

- Synthetic image generation is a way of augmenting the quantity of training data.
- Generating images leads to *perfect* ground truth segmentation. This further enhances the network's performance.

3. How it works?



Representation of the training setup

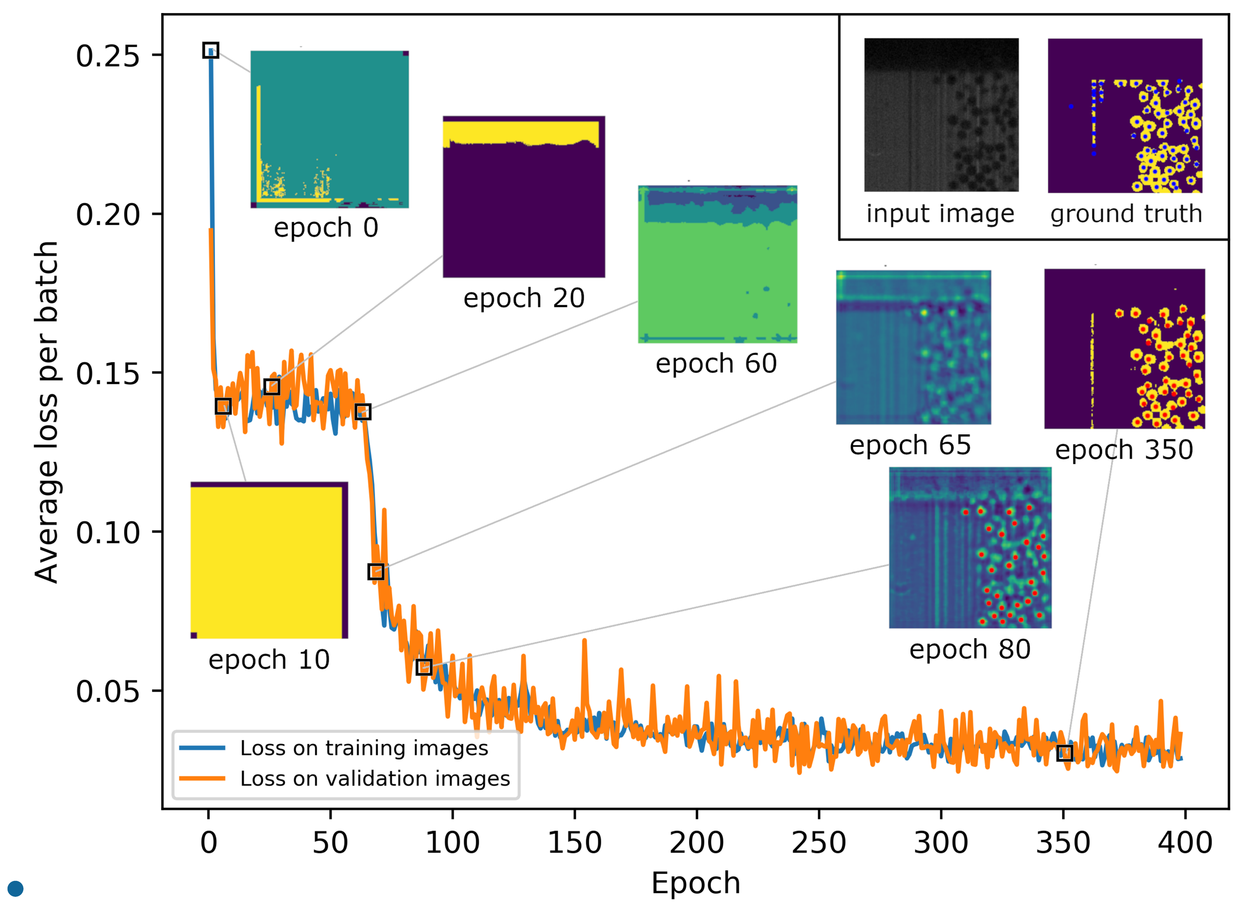
Training

- Randomly initialised weights
- Training loop:
  - → Network applied to a batch of Input Images
  - → **Loss** calculated on output image
  - → Weights update

Loss: L2 metric

The Loss L on an output image is calculated: $L = \frac{1}{n}\sum_{k=0}^{n-1}(x_k - y_k)^2$ Where x (*Input Image*) and y (*Ground Truth*) are tensors of arbitrary shapes, with n elements. The Loss on individual images is then averaged over a batch containing 8 images.

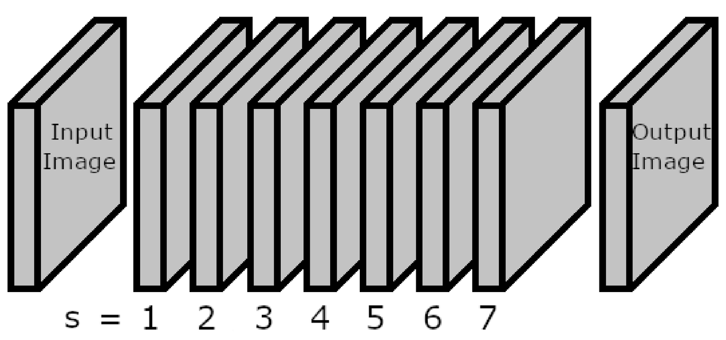
4. Training of the network



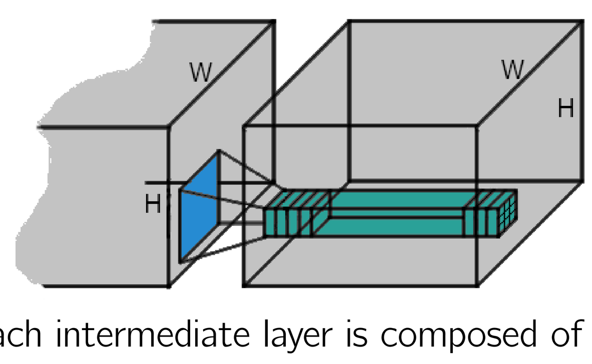
- Evaluation of losses throughout training process and network output on the same test image in different training stages

- During "Training", the network is applied to a batch of images, the loss on the output images is calculated, and the network parameters get modified in a way that the Loss decreases. -> Gradient Descent
- The best way to change parameters in order to decrease Loss is to change them to the opposite direction of the Loss function's gradient with respect to the given parameter -> Need to compute Gradient -> Backpropagation
- Numerical gradients are slow to compute and error prone. -> Staged computation of analytical gradient. The analytical gradient of every mathematical operation is calculated, and values backpropagate from the end of the function to the parameter at the beginning. Thus we know how to change the given parameter to reduce Loss.

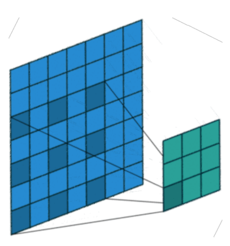
5. The structure of the Convolutional Neural Network



7 convolutional layers of BlobNet



Each intermediate layer is composed of 24 channels. Every channel convolves with each of the input channels with a 3×3 convolution kernel.



Graphical explanation of a dilated convolution with 3×3 kernel

- The input image is a $H \times W \times 3$ dimensional tensor in case of an RGB image
- 7 layers, each produces intermediate outputs of dimension $H \times W \times 24$. Every intermediate output is a 24 channel image.
- Each layer's every channel convolves with each channel of the previous layer. A **dilation** of $r_s = 2^{s-1}$ (s augments at each layer) is applied
  The i^{th} channel of layer $\mathbf{L}^s$ is computed from the previous layer's channels as follows:

$$\mathbf{L}_i^s = \Phi\left(\Psi^s\left(b_i^s + \sum_j \mathbf{L}_j^{s-1} \star_{r_s} \mathbf{K}_{i,j}^s\right)\right)^{[1]}$$

- Where $\mathbf{L}_j^{s-1}$ is the j^{th} channel of layer $\mathbf{L}^{s-1}$, b_i^s is a scalar bias and $\mathbf{K}_{i,j}^s$ is a 3×3 convolution kernel. The operator $\star_{r_s}$ is a dilated convolution with dilation r_s.
- The dilation in the convolution acts as a spatial filter. $r_s - 1$ holes are inserted between elements throughout convolution.
  - → First layer: $s = 1$, $r_s = 2^{1-1} = 1$. No holes, all spatial frequencies are seen.
  - → Second layer: $s = 2$, $r_s = 2^{2-1} = 2$. One hole is inserted (as on picture at left), high frequencies filtered.
  - → etc...

[1] : Q. Chen et al., Fast Image Processing with Fully-Convolutional Networks, ICCV, 2017

6. $\Psi^s(x)$ and $\Phi(x)$

- $\Psi^s(x) = \lambda_s x + \mu_s BN(x)$ is the **adaptive batch normalisation** function, where $BN(x)$ is the Batch Normalisation operator [2]. λ_s and μ_s are learneable parameters alongside all other parameters of the network.
- $\Phi(x) = max(\alpha x, x)$ is a point-wise nonlinearity, called Leaky Rectified Linear Unit (**LReLU**) [3], where we use $\alpha = 0.2$

[2] : S. Ioffe and C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, ICML, 2015
[3] : A. L. Maas et al., Rectifier nonlinearities improve neural network acoustic model, ICML, 2013

INPHYNI
INSTITUT DE PHYSIQUE DE NICE
MINES ParisTech
PSL
cnrs