Scale-free Texture Segmentation: Expert Feature-based versus Deep Learning strategies

Barbara Pascal, Vincent Mauduit, Nelly Pustelnik, Patrice Abry

Scale-free Texture Segmentation: Expert Feature-based versus Deep Learning strategies

Barbara Pascal, Vincent Mauduit, Nelly Pustelnik, and Patrice Abry
Univ Lyon, ENS de Lyon, Univ Claude Bernard Lyon 1, CNRS, Laboratoire de Physique
Lyon, France
firstname.lastname@ens-lyon.fr

***Abstract*—Texture segmentation constitutes a central task in image processing, classically based on two-step procedures consisting first in computing hand-crafted features devised from a priori expert knowledge and second in combining them into clustering algorithms. Deep learning approaches can be seen as merging these two steps into a single one with both discovering features and performing segmentation. Using fractal textures, often seen as relevant models in real-world applications, the present work compares a recently devised texture segmentation algorithm incorporating expert-driven scale-free features estimation into a Joint TV optimization framework against convolutional neural network architectures. From realistic synthetic textures, comparisons are drawn not only for segmentation performance, but also with respect to computational costs, architecture complexities and robustness against departures between training and testing datasets.**

***Index Terms*—Deep learning, CNN, Texture, Segmentation, Fractal, Total variation, Wavelets.**

I. Introduction

Context. Automated image segmentation constitutes a crucial task in image processing and computer vision, for many different purposes ranging from medical imaging [1] to autonomous driving [2]. In the last decade, the tremendous increase in computational/storage capabilities has triggered a massive use of deep learning for segmentation, because deep convolutional neural networks have the potential to discover and aggregate relevant information from large scale shapes to fine scale structures. Recently, numerous real-world applications, related to biological tissues [3], [4], geological samples [5], satellite images [6],... drove the focus specifically on *texture segmentation*, at the core of the present work. Texture segmentation however differs drastically from object detection. Indeed, textures are mostly characterized by small scale statistical properties rather than by geometry and large scale structures.

Related works. For years, texture segmentation was performed via a classical two-step procedure: First, prior knowledge or expert choice driven features are computed (e.g., Gabor, gradients, differences of oriented Gaussians,...) ; Second, these features are combined via a clustering algorithm (cf. [7] for a state-of-the-art review). Recently, research focus has been on combining these two steps into a single one to improve interface detection and thus segmentation performance. This has been first envisaged by retaining hand-crafted features but modifying classical frameworks (e.g., spectral clustering based on multiple local cues in [8], nonnegative matrix factorization in [9] or combining estimation and detection into a convex optimization formulation [10]). Recently, Deep Learning renewed this topic, jointly performing feature selection [11] as well as segmentation, first for semantic segmentation [12], [13], [14], [15], rapidly followed by texture segmentation [16], [7], [17].

Amongst others, fractal textures, also referred to as scale-free because their statistics are not controlled by particular scales, are considered relevant models for and massively used in numerous real-world applications very different in nature (cf. e.g., [18], [19], [20], [21], [22], [10]), ranging from physics of rough materials [6] to art investigations [23]. Interested readers are referred to several other contributions for further examples [21], [24], [25] and to a recent work [26], implementing a fractal dimension-based contour detector as a Convolutional Neural Network (CNN).

Goal, contributions and outline. Making use of mixtures of synthetic fractal textures (defined in Section II-A), the goal of the present work is to compare expert-knowledge driven texture segmentation exploiting a priori chosen scale-free features (such as local regularity) within an advanced unsupervised joint TV based optimization framework [10] (see Section II-B) versus *blind* or *non informed* supervised CNN based Deep Learning (DL) approaches (described in Section III) inspired from those in [7]. Comparisons are reported in Section IV not only in terms of absolute segmentation performance but also in terms of architecture complexities (three DL architectures are devised and compared), of learning complexities (computational costs and hyper-parameter tuning issues are discussed) as well as of robustness with respect to both the size of the training set and departures between training and testing sets, a realistic issue in real-world applications, where one cannot always fully control the *natural* variability encountered in data.

II. Unsupervised learningfor fractal textures

A. Fractal textures

Fractal textures. A fractal texture consists of a stationary Gaussian field $X(\underline{z})$, whose covariance structure is fully defined by its variance Σ^2 and a fractal parameter H (cf. e.g. [27] for detailed definition). Textured images correspond to discretization of the field X on a pixel grid, denoted $\boldsymbol{X} = (X_{\underline{n}})_{\underline{n}\in\Omega} \in \mathbb{R}^{N_1\times N_2}$.

Piecewise fractal textures. For the present study, we consider piecewise fractal textures, consisting of a mixture of Q fractal textures (cf. Fig. 1, top left, for an example with $Q = 3$), each characterized by parameters $\left(\Sigma_q^2, H_q\right)$, $q = \{1,\dots,Q\}$.

Local multiscale analysis. It has been well-documented that fractal textures can be well analyzed using multiscale transforms, such as wavelet coefficients and wavelet leaders (cf. e.g., [27]). Wavelet leaders $\boldsymbol{\mathcal{L}}_{j,\underline{k}}$ at scale j and location $2^j\underline{k}$, are defined as a supremum of all wavelet coefficients located into a small spatial neighborhood at all finer scales [27]. It has been shown then that textures can be well-characterized by a local multiscale analysis as

$$\mathcal{L}_{j,\underline{k}} \underset{2^j\to 0}{=} v_{\underline{n}} 2^{j h_{\underline{n}}}, \quad \underline{n} = 2^j\underline{k}, \tag{1}$$

with $v_{\underline{n}}^2 \propto \sigma_{\underline{n}}^2$ (a local variance) and where $h_{\underline{n}}$ quantifies the regularity of the texture locally around location $\underline{n}$. For homogeneous fractal textures, local variance and local regularities are constant over the texture: $h_{\underline{n}} \equiv H$ and $\sigma_{\underline{n}}^2 \equiv \Sigma^2$. For piecewise fractal textures,

Work supported by CBP (Blaise Pascal Center) with the use of SIDUS (Single Instance Distributing Universal System) implemented by E. Quemener.

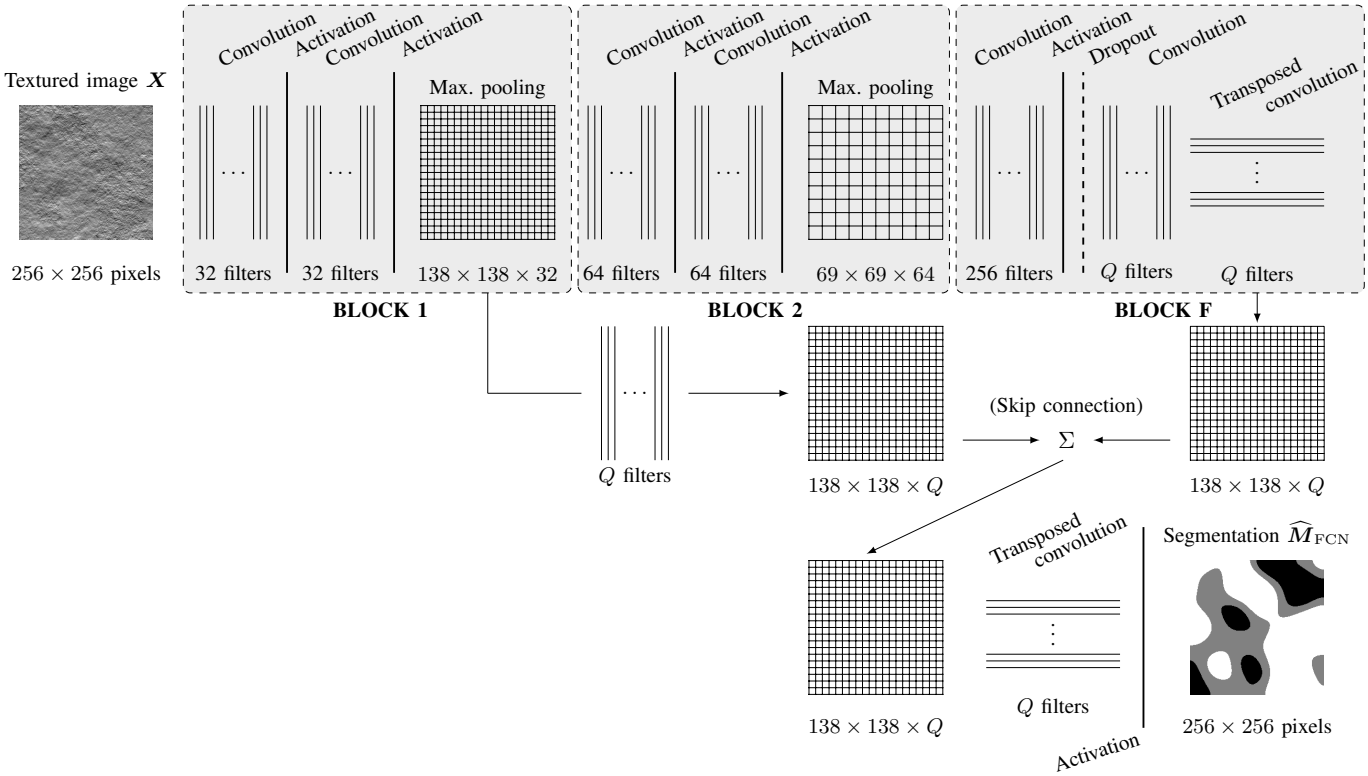


Fig. 1: **Fully convolutional neural network for texture segmentation.** Sketch of FCNN with skip connexions (simplest architecture, $4\,10^5$ weights).

$h_{\underline{n}}$ and $\sigma^2_{\underline{n}}$ are expected to form piecewise constant regions where $\forall \underline{n} \in \Omega_q$, $\sigma^2_{\underline{n}} \equiv \Sigma^2_q$ and $h_{\underline{n}} \equiv H_q$.

B. Joint TV segmentation

Goal. The aim is to recover the underlying partition $\Omega = \cup_{q=1}^{Q} \Omega_q$ (cf. bottom-right on Fig. 1) of the image domain Ω from estimations of $h_{\underline{n}}$ and $\sigma^2_{\underline{n}}$ performed locally (i.e. pixel-wise) using wavelet leaders.
Segmentation scheme. To achieve that goal, we designed in [10] an objective function combining a least square data fidelity term, fitting the mathematical model (1), and total variation penalization, enforcing piecewise constancy, leading to the optimization problem:

$$\left(\widehat{\boldsymbol{v}}, \widehat{\boldsymbol{h}}\right) := \arg\min_{\boldsymbol{v},\boldsymbol{h}} \sum_{j=j_1}^{j_2} \|\boldsymbol{v} + j\boldsymbol{h} - \boldsymbol{\ell}_j(\boldsymbol{X})\|_2^2 + \lambda \mathcal{P}_\alpha(\boldsymbol{v}, \boldsymbol{h}) \quad (2)$$

where $\boldsymbol{\ell}_j(\boldsymbol{X}) = \log_2(\boldsymbol{\mathcal{L}}_j) \in \mathbb{R}^{N_1 \times N_2}$, $j = \{j_1, \ldots, j_2\}$ are the non-decimated log-*leaders* and $\mathcal{P}_\alpha = \mathrm{TV}(\boldsymbol{v}) + \alpha \mathrm{TV}(\boldsymbol{h})$ (cf. [28] for TV definition). The regularization parameters, $\lambda > 0$ and $\alpha > 0$, have to be tuned to reach the best segmentation accuracy.

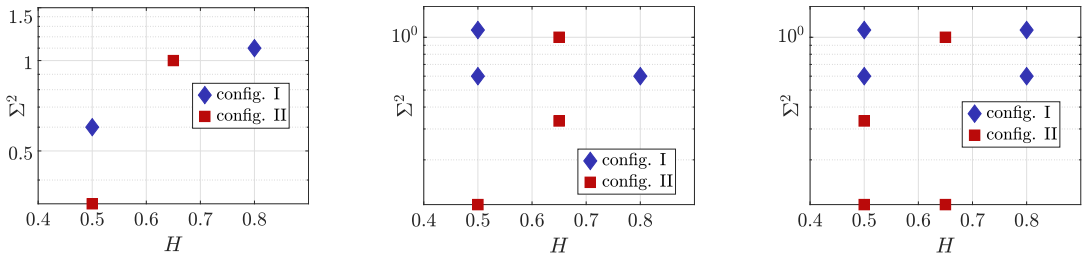
To perform the minimization (2), we have recourse to an accelerated proximal primal-dual algorithm with convergence guarantees to the global minimizer $\left(\widehat{\boldsymbol{v}}, \widehat{\boldsymbol{h}}\right)$ [29]. The segmentation $\widehat{\boldsymbol{M}}_{\mathrm{ROF}}$ (i.e. a label map of same size as the analyzed image) is then obtained by thresholding $\widehat{\boldsymbol{h}}$ as proposed in [30], i.e. $\widehat{\boldsymbol{M}}_{\mathrm{ROF}} = \mathcal{T}_Q(\widehat{\boldsymbol{h}})$.

III. Supervised fully CNN

Fully convolutional neural networks (FCNN). - We followed the work of Andrearczyk et al. [7] in which the state-of-the-art semantic segmentation FCN-8s Network [13], is tailored to texture segmentation. Mathematically, the FCNN takes entry $\boldsymbol{X}^{(\ell)}$ (256×256 image) and outputs 256×256 label map $\boldsymbol{M}^{(\ell)} \equiv \mathcal{R}_\theta(\boldsymbol{X}^{(\ell)})$, where θ denotes the weights of the network and $\mathcal{R}_\theta$ encapsulates the successive (convolution, activation, ...) operations. The two major ingredients tuned to texture segmentation are, first, the skip connections with shallow layers (enabling to keep precise localization), and second, making use of a large number of filters at each layer to capture as much as can be of the rich properties of textures. The state-of-the-art FCNN in [7] reaches a size of $8 \cdot 10^7$ weights for textured images of size $N_1 = N_2 = 256$.

Networks designed here. Starting from the state-of-the-art FCNN described above, we further propose two successively simplified versions, whose sizes reduce respectively to $2 \cdot 10^6$ and $4 \cdot 10^5$ weights. The global architecture of the three networks is thus similar and they only differ in complexity (number of weights). Hence, in the following, they are referred to by their number of weights to emphasize the complexity hierarchy. The proposed networks are composed of

(i) Several successive convolutional blocks, or *layers*, (BLOCKS 1, 2 on Fig. 1) combining a sequence of convolutions with ReLU activations. Layers are terminated by a polling operation, inducing resolution decrease enabling to deal with more global features as one gets deeper into the network.
(ii) These layers are followed by a fully convolutional block composed of one convolutional layer having a large number of filters and wider kernels of size 5, one convolution getting to targeted final depth Q and one transposed convolution, upsampling the feature maps (BLOCK F on Fig. 1).
(iii) This is followed by skip connections between the output of shallow blocks (e.g. BLOCK 1 on Fig. 1) and the output of BLOCK F, performing successive appropriate upsampling to produce a label map with same resolution $N_1 \times N_2$ as input images.
(iv) Finally, a softmax activation is applied to produce decision, that is one class label per pixel.



(a) Two classes ($Q = 2$) (b) Three classes ($Q = 3$) (c) Four classes ($Q = 4$)

Fig. 2: Texture configurations $\left\{\left(\Sigma_q^2, H_q\right),\, q = 1, \ldots, Q\right\}$.

These four steps are summarized in Fig. 1 for the $4 \cdot 10^5$ w. network. For all networks, the default filter size is $p = 3$, the default stride is $s = 1$ and a $(2, 2)$ pooling is applied at the end of each block.

The three explored architectures are

• $4 \cdot 10^5$ w. net: This proposed network is composed of 2 convolutional blocks. More specifically, BLOCK 1 is composed of two layers, each having 32 filters and BLOCK 2 has two layers, each having 64 filters. The fully convolutional layer in BLOCK F contains 256 filters. One skip connection combine information from BLOCK 1 and BLOCK F using Q filters.

• $2 \cdot 10^6$ w. net: This proposed network is composed of 3 blocks, BLOCK 1 with two layers, each having 32 filters, BLOCK 2 with two layers, each having 64 filters and BLOCK 3 with three layers each having 128 filters. The fully convolutional layer in BLOCK F contains 512 filters. Two skip connections connect BLOCK 1 and BLOCK 2 to the final block, each having Q filters.

• $8 \cdot 10^7$ w. net: Andrearczyk et al. [7] network is composed of 4 convolutional blocks. BLOCK 1 is composed of two layers, with 64 filters, BLOCK 2 is composed of two layers, with 128 filters, BLOCK 3 is composed of three layers with 256 filters and BLOCK 4 has three layers, each having 512 filters. The fully convolutional layer in BLOCK F contains 4096 filters. Three skip connections connect BLOCK 1, BLOCK 2 and BLOCK 3 to final block.

Supervised learning – From a database $\left\{\boldsymbol{X}^{(\ell)}, \boldsymbol{M}^{(\ell)}\right\}_{\ell=0}^{L}$ an optimization strategy estimates the optimal weights $\widehat{\theta}$ of the network as:

$$\widehat{\theta} = \text{argmin}_\theta \; d(\widehat{\boldsymbol{M}}^{(\ell)}, \mathcal{R}_\theta(\boldsymbol{X}^{(\ell)})) \tag{3}$$

where $\mathcal{R}_\theta$ summarized the successive steps **(i)-(iv)**, composing the neural network. Finally the supervised segmentation is obtained as $\widehat{\boldsymbol{M}}_{\text{FCN}}^{(\ell)} = \mathcal{R}_{\widehat{\theta}}(X)$.

IV. Experiments on synthetic textures

A. Experimental settings

Learning dataset. As sketched in Fig. 2, we investigate Q-class segmentation with $Q \in \{2, 3, 4\}$. Each training image $\boldsymbol{X}^{(\ell)}$ is built from a partition

$$\Omega^{(\ell)} = \cup_{q=1}^{Q} \Omega_q^{(\ell)}, \quad \text{with } q \neq q' \Rightarrow \Omega_q^{(\ell)} \cap \Omega_{q'}^{(\ell)} = \emptyset \tag{4}$$

randomly generated, and from the fractal descriptors of the Q textures $\left(\Sigma_q^2, H_q\right)_{q=1,\ldots,Q}$. For each Q, we consider two configurations, i.e., two different sets of characteristic descriptors $\left\{\left(\Sigma_q^2, H_q\right),\, q = 1, \ldots, Q\right\}$. In Config. I (in blue on Fig.1) textures to be segmented have large $\Delta H \equiv H_q - H_{q'}$ and small $\Delta\Sigma^2 \equiv \Sigma_q^2 - \Sigma_{q'}^2$ and conversely in Config. II (in red in Fig.1).

To train neural networks, for Config. I, II and $Q = \{2, 3, 4\}$, we generated a data set of 2000 $\boldsymbol{X}^{(\ell)}$, associated with ground truth segmentation $\boldsymbol{M}^{(\ell)}$ (label map equivalent to the partition $\Omega^{(\ell)} = \cup_{q=1}^{Q} \Omega_q^{(\ell)}$). Each texture $\boldsymbol{X}^{(\ell)}$ contains samples of each Q textures (i.e. $\Omega_q^{(\ell)} \neq \emptyset$, $\forall q$, $\forall \ell$).

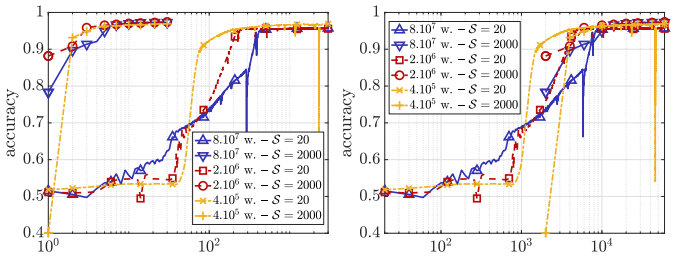
TV-based texture segmentation. Following [10], the wavelet transform of textured images is performed using Daubechies wavelets

	2 classes	3 classes	4 classes
Trained on Config. I, tested on Config. I			
Trained on 2000 images			
Joint TV	$93.2 \pm 0.8\%$	$69.3 \pm 2.8\%$	$58.6 \pm 1.5\%$
FCNN $8 \cdot 10^7$ / $\mathcal{S} = 2000$	$97.3 \pm 0.6\%$	$97.8 \pm 0.3\%$	$97.1 \pm 0.4\%$
FCNN $2 \cdot 10^6$ / $\mathcal{S} = 2000$	$97.4 \pm 0.6\%$	$98.1 \pm 0.3\%$	$96.8 \pm 0.5\%$
FCNN $4 \cdot 10^5$ / $\mathcal{S} = 2000$	$96.9 \pm 0.7\%$	$98.0 \pm 0.3\%$	$96.5 \pm 0.5\%$
Trained on 20 images			
FCNN $8 \cdot 10^7$ / $\mathcal{S} = 20$	$95.5 \pm 0.9\%$	$97.5 \pm 0.4\%$	$95.4 \pm 0.8\%$
FCNN $2 \cdot 10^6$ / $\mathcal{S} = 20$	$95.4 \pm 1.1\%$	$97.4 \pm 0.5\%$	$95.9 \pm 0.7\%$
FCNN $4 \cdot 10^5$ / $\mathcal{S} = 20$	$96.6 \pm 0.7\%$	$98.0 \pm 0.4\%$	$96.5 \pm 0.5\%$
Trained on Config. II, tested on Config. II			
Joint TV	$97.8 \pm 0.2\%$	$95.2 \pm 3.1\%$	$64.9 \pm 1.4\%$
Trained on 2000 images			
FCNN $8 \cdot 10^7$ / $\mathcal{S} = 2000$	$99.1 \pm 0.2\%$	$98.3 \pm 0.3\%$	$95.7 \pm 0.5\%$
FCNN $2 \cdot 10^6$ / $\mathcal{S} = 2000$	$99.0 \pm 0.2\%$	$98.5 \pm 0.3\%$	$95.6 \pm 0.5\%$
FCNN $4 \cdot 10^5$ / $\mathcal{S} = 2000$	$99.1 \pm 0.2\%$	$98.4 \pm 0.3\%$	$95.2 \pm 0.6\%$
Trained on 20 images			
FCNN $8 \cdot 10^7$ / $\mathcal{S} = 20$	$98.8 \pm 0.2\%$	$97.9 \pm 0.3\%$	$94.5 \pm 0.7\%$
FCNN $2 \cdot 10^6$ / $\mathcal{S} = 20$	$98.6 \pm 0.3\%$	$97.4 \pm 0.4\%$	$93.0 \pm 0.9\%$
FCNN $4 \cdot 10^5$ / $\mathcal{S} = 20$	$98.8 \pm 0.3\%$	$98.3 \pm 0.3\%$	$94.8 \pm 0.6\%$
Trained on Config. I, tested on Config. II			
Joint TV	$79.2 \pm 2.9\%$	$95.2 \pm 1.2\%$	$66.3 \pm 1.1\%$
Trained on 2000 images			
FCNN $8 \cdot 10^7$ / $\mathcal{S} = 2000$	$91.2 \pm 2.1\%$	$65.7 \pm 7.2\%$	$55.6 \pm 3.4\%$
FCNN $2 \cdot 10^6$ / $\mathcal{S} = 2000$	$87.9 \pm 2.5\%$	$69.0 \pm 7.6\%$	$50.8 \pm 4.0\%$
FCNN $4 \cdot 10^5$ / $\mathcal{S} = 2000$	$81.8 \pm 3.8\%$	$65.2 \pm 7.2\%$	$46.4 \pm 3.7\%$
Trained on 20 images			
FCNN $8 \cdot 10^7$ / $\mathcal{S} = 20$	$91.4 \pm 1.6\%$	$63.3 \pm 7.1\%$	$54.7 \pm 3.3\%$
FCNN $2 \cdot 10^6$ / $\mathcal{S} = 20$	$92.4 \pm 1.6\%$	$65.6 \pm 7.4\%$	$44.4 \pm 3.4\%$
FCNN $4 \cdot 10^5$ / $\mathcal{S} = 20$	$86.3 \pm 2.6\%$	$64.9 \pm 7.2\%$	$48.4 \pm 3.8\%$
Trained on Config. II, tested on Config. I			
Joint TV	$90.9 \pm 2.8\%$	$66.7 \pm 2.5\%$	$52.0 \pm 1.5\%$
Trained on 2000 images			
FCNN $8 \cdot 10^7$ / $\mathcal{S} = 2000$	$56.2 \pm 13.5\%$	$73.5 \pm 8.2\%$	$50.9 \pm 3.9\%$
FCNN $2 \cdot 10^6$ / $\mathcal{S} = 2000$	$55.1 \pm 14.0\%$	$74.9 \pm 8.2\%$	$51.3 \pm 4.3\%$
FCNN $5\,10^5$ / $\mathcal{S} = 2000$	$55.5 \pm 13.8\%$	$72.6 \pm 8.1\%$	$50.2 \pm 3.8\%$
Trained on 20 images			
FCNN $8 \cdot 10^7$ / $\mathcal{S} = 20$	$57.1 \pm 13.3\%$	$71.1 \pm 8.2\%$	$52.6 \pm 3.8\%$
FCNN $2 \cdot 10^6$ / $\mathcal{S} = 20$	$55.3 \pm 14.0\%$	$71.7 \pm 8.4\%$	$49.6 \pm 4.2\%$
FCNN $5\,10^5$ / $\mathcal{S} = 20$	$62.3 \pm 11.5\%$	$71.0 \pm 8.2\%$	$54.1 \pm 3.7\%$

TABLE I: **Segmentation accuracy (% of well-classified pixels).**

with two vanishing moments [31]. The considered scales range from $2^{j_1} = 2^1$ to $2^{j_2} = 2^3$. For each Q and each configuration, the Joint algorithm is run on one training image $\boldsymbol{X}^{(1)}$ over a 20×20 grid of regularization parameters (λ, α), so as to select optimal parameters (in term of segmentation accuracy). Parameters λ and α are then frozen to optimal values and used to segment a set of one hundred test images (built in the same way as training images).

FCNN supervised training. The training for supervised learning consists in minimizing the loss d, computed over the entire training dataset (cf. Eq. (3)) with respect to the weights θ of the network. Two-class segmentation is trained minimizing binary cross entropy, while three and four-class use categorical cross entropy. The minimization is performed using ADAM optimizer with AMSGrad strategy [32], learning rate $2\,10^{-4}$ and batch size of 20 images. By construction of ADAM algorithm, all the textures in the training set are used once in the implementation of each epoch of Stochastic Gradient Descent. Thus, we defined the *number of images processed* along the training of FCNN, as the number of epochs times the size of training set. Performance for FCNN are evaluated by applying temporarily frozen FCNN on the same test set of one hundred textures as the one used for Joint TV optimization Algorithm.



(a) w.r.t. epochs (b) w.r.t. images processed

Fig. 3: **Evolution of accuracy along training phase.** Accuracy of different FCNN trained on Config. I, for two-classes segmentation.

B. Results and Discussion

Convergence. Fig. 3a reports, for Config. I and with $Q = 2$, the evolution of accuracy for the three networks, trained on the training dataset of 2000 textures. Fig. 3a clearly shows that FCNN trained on the full dataset of 2000 textures do converge to a stable maximal accuracy after the use of 30 epochs. Equivalent plots and conclusions are obtained for other configurations.

Performance. Tab. I reports segmentation performance after convergence, defined as segmentation accuracies (percentage of well-classified pixels). For $Q = 2$, Tab. I shows that both FCNN and Joint TV optimization Algorithm achieve comparably high accuracy on Config I and II. For $Q = 3$, FCNN perform as well as for $Q = 2$ for both configurations, while Joint TV optimization Algorithm only maintains competitive accuracy on Config II. For $Q = 4$, FCNN performance suffer only from a mild decrease, while those of Joint TV optimization Algorithm significantly degrade. FCNN with different complexities (different depth and width) show similar performance despite a decrease by 200 of the number of parameters from the largest state-of-the-art FCN-8s to the smallest FCNN tested here.

Robustness w.r.t. training set size $\mathcal{S}$. To investigate the impact of the training set size, learning was conducted a second time using a small subset of the learning dataset described in Sec IV-A of only $\mathcal{S} = 20$ textures. Fig. 3a shows that convergence requires a significantly larger number of epochs. Yet, Fig. 3b shows, that convergence occurs for both the large and small data sets, for a comparable number of training samples actually used and reused, referred to as the number of images processed. Altogether it shows that the number of epochs required is inversely proportional to the database size.

Further, Tab. I indicates only a slight decrease in performance when the small training set is used (despite its significantly smaller size). The use of a smaller training set goes along with a significant reduction in memory cost during training.

(a) Truth (b) Joint (c) FCNN $8 \cdot 10^7$

Fig. 4: $Q = 2$**-class segmentation** (trained and tested on Config. I), showing irregular contours for FCNN segmentation.

Robustness w.r.t. training configuration. To investigate the robustness of the segmentation procedures with respect to a mismatch between training and testing sets, Tab. I reports performance achieved when FCNN training and Joint TV parameter tuning are performed using one configuration, and testing is conducted using the other configuration. When trained on Config. I (large ΔH, small $\Delta\Sigma^2$) and tested on Config. II (small ΔH, large $\Delta\Sigma^2$), FCNN perform accurate segmentation for $Q = 2$ above Joint TV optimization performance. However, FCNN suffer from severe performance degradation as the number of classes Q increases, while Joint TV optimization degrades significantly less. When trained on Config. II and tested on Config. I, Joint TV optimization turns out to be significantly more robust than FCNN for $Q = 2$. All procedures show a lack of robustness for $Q = 4$. All together, Joint TV optimization shows equivalent, if not better (see Training on Config. I, tested on Config II with $Q = 3,\ 4$ classes in Tab. I), robustness compared to FCNN. Also, the FCNN with larger complexity does not show more robustness than the one with the lowest complexity.

Computational load. We now evaluate computational costs. For FCNN, $\mathcal{W}$ weights are learned, on $\mathcal{S}$ (either 2000 or 20) training segmented images, running ADAM for $\mathcal{I}$ (either 30 or 3000) epochs, resulting in a cost of

$$\mathcal{C} = \mathcal{W} \times \mathcal{S} \times \mathcal{I}. \tag{5}$$

For Joint TV optimization, there are $\mathcal{W} = 2$ hyperparameters (λ, α) to tune and a single image ($\mathcal{S} = 1$) is processed 400 times (20×20 couples (λ, α)), each performing $2.5\,10^4$ iterations, for a total of $\mathcal{I} = 10^7$ iterations. Resulting computational costs are compared in Tab. II showing drastically smaller costs for Joint TV optimization compared to FCNN.

	$\mathcal{W}$	$\mathcal{S}$	$\mathcal{I}$	$\mathcal{C}$
Joint TV	2	1	10^7	$2\,10^7$
FCNN $8 \cdot 10^7$ / $\mathcal{S} = \{20, 2000\}$	$8 \cdot 10^7$	$\{20, 2000\}$	$\{3000, 30\}$	$4.8\,10^{12}$
FCNN $2 \cdot 10^6$ / $\mathcal{S} = \{20, 2000\}$	$2 \cdot 10^6$	$\{20, 2000\}$	$\{3000, 30\}$	$1.2\,10^{11}$
FCNN $4 \cdot 10^5$ / $\mathcal{S} = \{20, 2000\}$	$4 \cdot 10^5$	$\{20, 2000\}$	$\{3000, 30\}$	$2.4\,10^{10}$

TABLE II: **Computational cost $\mathcal{C}$ for training phase.** The estimation is performed using the formula proposed in Eq. (5).

Interface accuracy. FCNN lead to irregular interfaces (as emphasized in [7] and illustrated on Fig. 4). Yet, for applications requiring precise interface length measurements, such as [10], it is crucial to recover accurate and smooth contours. Tab. III compares relative errors on total interface length (i.e., the number of pixels at the border of two different regions) for $Q = 2$-class segmentation, and shows that joint TV optimization systematically and significantly outperforms FCNN. Moreover, when trained in one configuration and tested on another, FCNN perform particularly bad at measuring interfaces.

Train/Test	Config. I/I	Config. II/II	Config. I/II	Config. II/I
Joint TV	14 ± 4	13 ± 2	48 ± 22	16 ± 4
FCNN $8 \cdot 10^7$	33 ± 3	36 ± 2	112 ± 24	68 ± 18
FCNN $2 \cdot 10^6$	37 ± 3	41 ± 3	213 ± 72	77 ± 4
FCNN $4 \cdot 10^5$	24 ± 3	18 ± 2	337 ± 131	62 ± 9

TABLE III: **Relative error on interface length for two-class segmentation.** FCNN are trained on $\mathcal{S} = 2000$ images. (Similar results are obtained with $\mathcal{S} = 20$.)

V. Conclusion

This work showed that supervised FCNN compares favorably against unsupervised hand-crafted feature Joint TV optimization for fractal texture segmentation, even with reduced training data sets, and reduced complexities (compared to state-of-the-art architectures), at the cost though of much larger computation and memory resources. However, FCNN did not show robustness in training/testing datasets mismatch (even those with largest complexities) and yield poor estimation of interface lengths, thus still making their use debatable in real-world applications. Following recent works in deep learning literature a natural option would be to combine both approaches so as to exploit the advantages of each, as LISTA [33] did.

References

[1] R. Trullo, C. Petitjean, D. Nie, D. Shen, and S. Ruan, "Joint segmentation of multiple thoracic organs in ct images with two collaborative deep architectures," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pp. 21–29. Springer, 2017.

[2] K.-T. Song and H.-Y. Chen, "Lateral driving assistance using optical flow and scene analysis," in *2007 IEEE Intelligent Vehicles Symposium*. IEEE, 2007, pp. 624–629.

[3] R. Jennane, W. J. Ohley, S. Majumdar, and G. Lemineur, "Fractal analysis of bone x-ray tomographic microscopy projections," *IEEE Trans. Med. Imag.*, vol. 20, no. 5, pp. 443–449, 2001.

[4] Z. Marin, K. A. Batchelder, B. C. Toner, L. Guimond, E. Gerasimova-Chechkina, A. R. Harrow, A. Arneodo, and A. Khalil, "Mammographic evidence of microenvironment changes in tumorous breasts," *Medical physics*, vol. 44, no. 4, pp. 1324–1336, 2017.

[5] S. M. Angelo, M. Matos, and K. J. Marfurt, "Integrated seismic texture segmentation and clustering analysis to improved delineation of reservoir geometry," in *SEG Technical Program Expanded Abstracts 2009*, pp. 1107–1111. Society of Exploration Geophysicists, 2009.

[6] S.G. Roux, A. Arneodo, and N. Decoster, "A wavelet-based method for multifractal image analysis. III. applications to high-resolution satellite images of cloud structure," *The European Physical Journal B-Condensed Matter and Complex Systems*, vol. 15, no. 4, pp. 765–786, 2000.

[7] V. Andrearczyk and P.F. Whelan, "Texture segmentation with fully convolutional networks," *preprint arXiv:1703.05230*, 2017.

[8] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Match. Int.*, vol. 33, no. 5, pp. 898–916, 2011.

[9] J. Yuan, D. Wang, and A. M. Cheriyadat, "Factorization-based texture segmentation," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 3488–3497, Nov. 2015.

[10] B. Pascal, N. Pustelnik, P. Abry, M. Serres, and V. Vidal, "Joint estimation of local variance and local regularity for texture segmentation. application to multiphase flow characterization," in *Proc. IEEE Int. Conf. Image Proc. (ICIP)*, Athens, Greece, 2018, IEEE, pp. 2092–2096.

[11] J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Y. Jia, and E. Shelhamer, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. IEEE Int. Conf. Comput. Vis. and Patt. Rec.*, Columbus, Ohio, USA, 2014, Citeseer.

[12] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *ICLR*, 2014.

[13] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. and Patt. Rec.*, Boston, Massachusetts, 2015, pp. 3431–3440.

[14] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," in *Proc. IEEE Int. Conf. Comput. Vis. and Patt. Rec.*, Boston, Massachusetts, USA, 2015, pp. 447–456.

[15] P. Luc, C. Couprie, S. Chintala, and J. Verbeek, "Semantic segmentation using adversarial networks," in *NIPS Workshop on Adversarial Training*, Barcelona, Spain, 2016.

[16] M. Cimpoi, S. Maji, I. Kokkinos, and A. Vedaldi, "Deep filter banks for texture recognition, description, and segmentation," *Int. J. Comp. Vis.*, vol. 118, no. 1, pp. 65–94, 2016.

[17] I. Ustyuzhaninov, C. Michaelis, W. Brendel, and M. Bethge, "One-shot texture segmentation," *preprint arXiv:1807.02654*, 2018.

[18] A. P. Pentland, "Fractal-based description of natural scenes," *IEEE Trans. Pattern Anal. Match. Int.*, vol. 6, pp. 661–674, 1984.

[19] B. Hnat, S. C. Chapman, G. Gogoberidze, and R.T. Wicks, "Scale-free texture of the fast solar wind," *Physical Review E*, vol. 84, no. 6, pp. 065401, 2011.

[20] H. Wendt, P. Abry, S. Jaffard, H.Ji, and Z. Shen, "Wavelet leader multifractal analysis for texture classification," in *Proc. IEEE Int. Conf. Image Proc. (ICIP)*, Cairo, Egypt, 2009.

[21] J. M. Keller, S. Chen, and R. M. Crownover, "Texture description and segmentation through fractal geometry," *Computer Vision, Graphics, and Image Processing*, vol. 45, no. 2, pp. 150–166, 1989.

[22] B. B. Chaudhuri and N. Sarkar, "Texture segmentation using fractal dimension," *IEEE Trans. Pattern Anal. Match. Int.*, vol. 17, no. 1, pp. 72–77, 1995.

[23] P. Abry, S. Jaffard, and H. Wendt, "When Van Gogh meets Mandelbrot: Multifractal classification of painting's texture," *Signal Process.*, vol. 93, no. 3, pp. 554–572, 2013.

[24] L.M. Kaplan, "Extended fractal analysis for texture classification and segmentation," *IEEE Trans. Image Process.*, vol. 8, no. 11, pp. 1572–1585, 1999.

[25] F. Riaz, A. Hassan, S. Rehman, and U. Qamar, "Texture classification using rotation-and scale-invariant gabor texture features," *Signal Process. Lett.*, vol. 20, no. 6, pp. 607–610, 2013.

[26] H. Xu, J. Yan, N. Persson, W. Lin, and H. Zha, "Fractal dimension invariant filtering and its CNN-based implementation," in *Proc. IEEE Int. Conf. Comput. Vis. and Patt. Rec.*, Honolulu, Hawaii, USA, 2017, pp. 3491–3499.

[27] H. Wendt, S. G. Roux, P. Abry, and S. Jaffard, "Wavelet leaders and bootstrap for multifractal analysis of images," *Signal Process.*, vol. 89, no. 6, pp. 1100–1114, 2009.

[28] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: nonlinear phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.

[29] B. Pascal, N. Pustelnik, and P. Abry, "Nonsmooth convex joint estimation of local regularity and local variance for fractal texture segmentation," *preprint arXiv :1910.05246*, 2019.

[30] X. Cai and G. Steidl, "Multiclass segmentation by iterated rof thresholding," in *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, Lund, Sweden, 2013, Springer, pp. 237–250.

[31] S. Mallat, *A wavelet tour of signal processing*, Elsevier, 1999.

[32] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," *preprint arXiv:1904.09237*, 2019.

[33] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *International Conference on Machine Learning*, 2010, pp. 399–406.