Integration of Morphological Operators in Neural Networks

Romain Hermary
(supervisor: Guillaume Tochon and Élodie Puybareau)

Technical Report n^o2122, July 2021
revision 4a429397

Image processing is a very broad field of study that encompasses a multitude of operations, each of them with different purposes and circumstances of use, complexities and results. Nowadays, the bests results for automatic image study (image segmentation, image classification, object detection, etc.) are obtained using deep learning, and more specifically convolutional neural networks. We explore and conduct experiments on a specific part of image processing, mathematical morphology, investigating on the best way of circumventing operations' complexities regarding their integration in a supervised learning pipeline.

Le traitement d'image est un domaine d'étude très vaste qui englobe une multitude d'opérations, chacune ayant des spécificités, des circonstances d'utilisation, des complexités et des résultats différents. Aujourd'hui, les meilleurs résultats pour l'étude automatique des images (segmentation d'images, classification d'images, détection des objets, etc.) sont obtenus en utilisant l'apprentissage profond, et plus spécifiquement les réseaux neuronaux convolutifs. Nous explorons et menons des expériences sur une partie spécifique du traitement d'image, la morphologie mathématique, en recherchant la meilleure façon de contourner les complexités des opérations concernant leur intégration dans un pipeline d'apprentissage supervisé.

Keywords
mathematical morphology, morphological layer, morphological neural network, deep learning, grayscale mathematical morphology, image processing

Laboratoire de Recherche et Développement de l'EPITA
14-16, rue Voltaire – FR-94276 Le Kremlin-Bicêtre CEDEX – France
Tél. +33 1 53 14 59 22 – Fax. +33 1 53 14 59 13
rhermary@lrde.epita.fr – http://www.lrde.epita.fr/

Copying this document

Contents

Chapter 1

Introduction

Mathematical Morphology (MM) is a part of digital image processing that is concerned with image filtering and geometric analysis by structuring elements —which can be assimilated to other images in the gray value morphology domain. Originated from the early work of Minkowski and Hadwiger on geometric measure theory and integral geometry (Hadwiger, 1957; Hilbert et al., 1911; Minkowski, 1903), and modernly formulated by Matheron and Serra (Matheron, 1975; Serra, 1983), mathematical morphology operations are very powerful tools, with a success coming from their ability to manage the finiteness of the digital image range (Angulo et al., 2017).

The idea behind mathematical morphology is to locally compare structures within the image with a reference shape called the Structuring Element (SE). MM can provide, among other practical interests, boundaries of objects, their skeletons, and their convex hulls. It is also useful for many pre- and post-processing techniques, especially in edge thinning and pruning. Generally speaking, most morphological operations are based on simple expanding and shrinking operations.

Nonetheless, for specific objectives and precise results, image manipulation with MM operators can be hard: finding the right combinations of operations and structuring elements is a very tedious part of the process; furthermore, attaining optimal solution is near-impossible when analyzing complex and heterogeneous data with sophisticated objectives. With this problematic in mind and looking at the recent breakthrough of deep learning in image processing initiated by Krizhevsky et al. (2012), one appropriate thought could be to take advantage of ever-expending data sets and the generalization power that supervised learning provides to help us in the task of finding those required combinations of operators and structuring elements.

In this paper we will first go through a mathematical morphology and a supervised learning introductions in Chapter 2 and Chapter 3 respectively, followed by Chapter 4 in which we will summarize prior knowledge and experiments —as this paper is in the continuum of a previous one (Kirszenberg et al., 2021)—, and Chapter 5 where newly conduct experiments and freshly formed problematic will be detailed, ending with a conclusion in Chapter 6 where we will also express further objectives.

Chapter 2

Mathematical Morphology

In this chapter, I will introduce the basics of mathematical morphology, starting with the original set approach and definitions, continuing towards a more adapted and practical theory for conventional image analysis.

2.1 Binary Mathematical Morphology

Mathematical morphology describes two operations at the root of everything, that are *erosion* and *dilation*. The base definitions of those transformations are binary oriented, which means it is just about space comparisons (intersection and inclusion) between two sets (Matheron, 1975; Serra, 1983). For the following formulas we define an Euclidean space E, X a subset of it, which can be identified as the manipulated image, and B another set, which can be seen as a neighborhood delimiter, called the structuring element:

$$\varepsilon_B(X) = \{z \in E, B_z \subseteq X\} \qquad \text{(erosion)} \tag{2.1}$$

$$\delta_B(X) = \{z \in E, B_z \cap X \neq \emptyset\} \qquad \text{(dilation)} \tag{2.2}$$

Where B_z is the structuring element B centered on the studied z element (delimiting a neighborhood around z):

$$B_z = \{b + z \mid b \in B\} \tag{2.3}$$

Essentially, for the erosion, the element z of E will be in the resulting set $\varepsilon_B(X)$ if the set B_z is entirely included in X (if all the selected neighborhood of z is in X); for a dilation, z will be in the resulting set $\delta_B(X)$ if B_z and X intersect (see Figure 2.1).

Together, they can be combined to create higher order transformations: *opening* and *closing*. The opening $\gamma = \delta \circ \varepsilon$ can be useful to remove small objects (usually bright pixels) of an image, whereas the closing $\phi = \varepsilon \circ \delta$ removes (or fills) small holes (Ritter and Wilson, 1996).

2.2 Grayscale Mathematical Morphology

A downside of the previously defined binary operations is that it does not take into account the elements values. Conventional images are not binary and represented as sets, but grayscaled

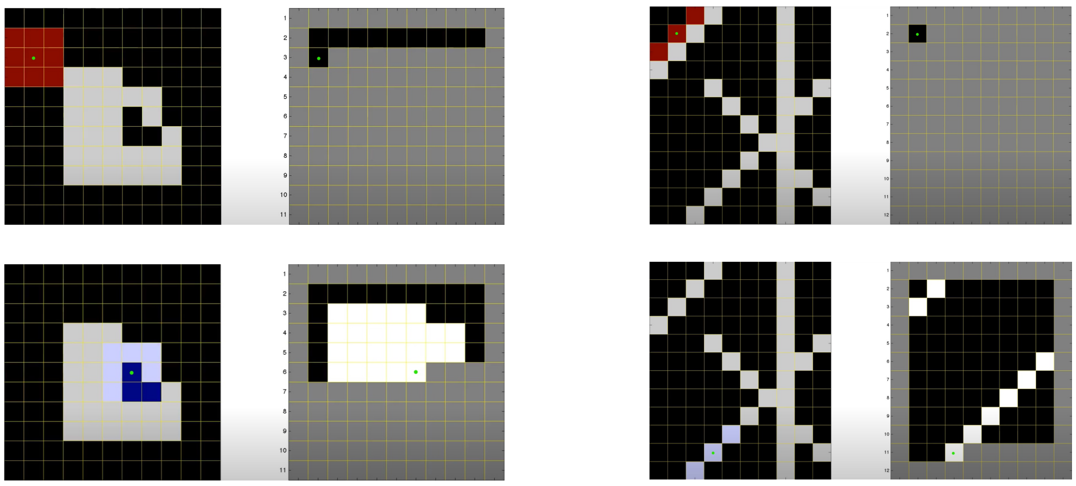


Figure 2.1: Each sub-image is presented as structuring element (colored) and shape (X) on the left side, and resulting set on the right. On the left figures an example of binary dilation, on the right figures an example of binary erosion. (Top Left) No intersection between the shape and the structuring element - (Bottom Left) Intersection - (Top Right) The structuring element is not included in the shape - (Bottom Right) Included. Pictures originally from Peter Corke (YouTube)

and could be seen as $3D$ landscapes (see Figure 2.2) —and so does a structuring element. With this in mind, grayscale morphological operators (see Figure 2.3) can be determined (Serra, 1983; Sternberg, 1986), with the image and structuring element now defined as functions, $f : E \subseteq \mathbb{Z}^2 \to \mathbb{R}$ and $b : B \subseteq E \to \mathbb{R}$, respectively. Operations obviously have to be adapted:

$$(f \ominus b)(x) = \inf_{y \in E} \{f(y) - b(x-y)\} \qquad \text{(erosion)} \tag{2.4}$$

$$(f \oplus b)(x) = \sup_{y \in E} \{f(y) + b(x-y)\} \qquad \text{(dilation)} \tag{2.5}$$

Those definitions also allow for a binary structuring element to be specified:

$$b(x) = \begin{cases} 0 & \text{if } x \in B \\ -\infty & \text{otherwise} \end{cases} \tag{2.6}$$

Certainly, opening and closing can also be defined:

$$f \circ b = (f \ominus b) \oplus b \qquad \text{(opening)} \tag{2.7}$$

$$f \bullet b = (f \oplus b) \ominus b \qquad \text{(closing)} \tag{2.8}$$

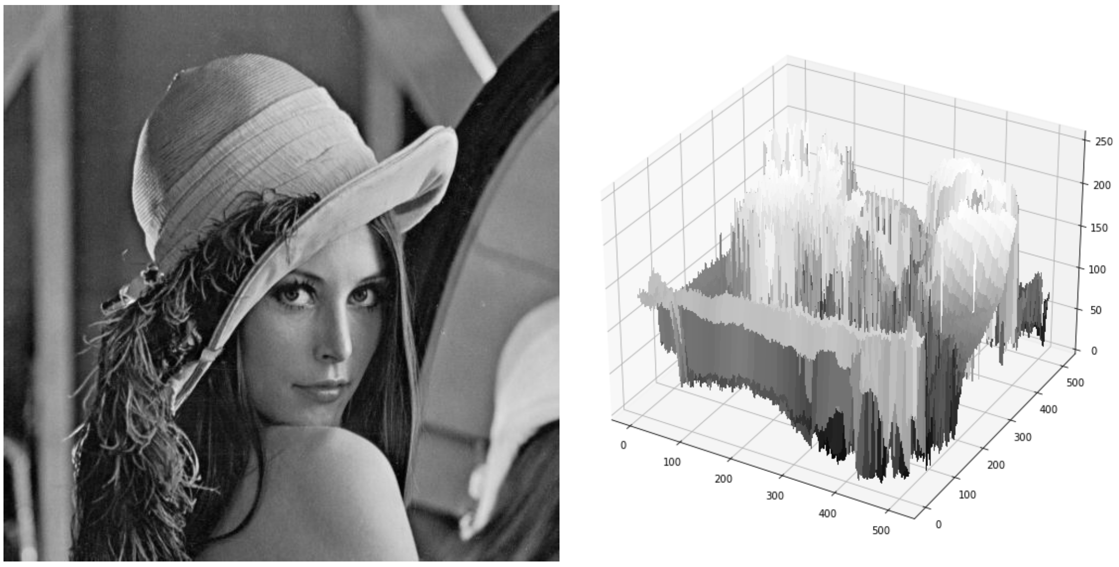


Figure 2.2: Grayscale image (left) and its 3D representation (right)

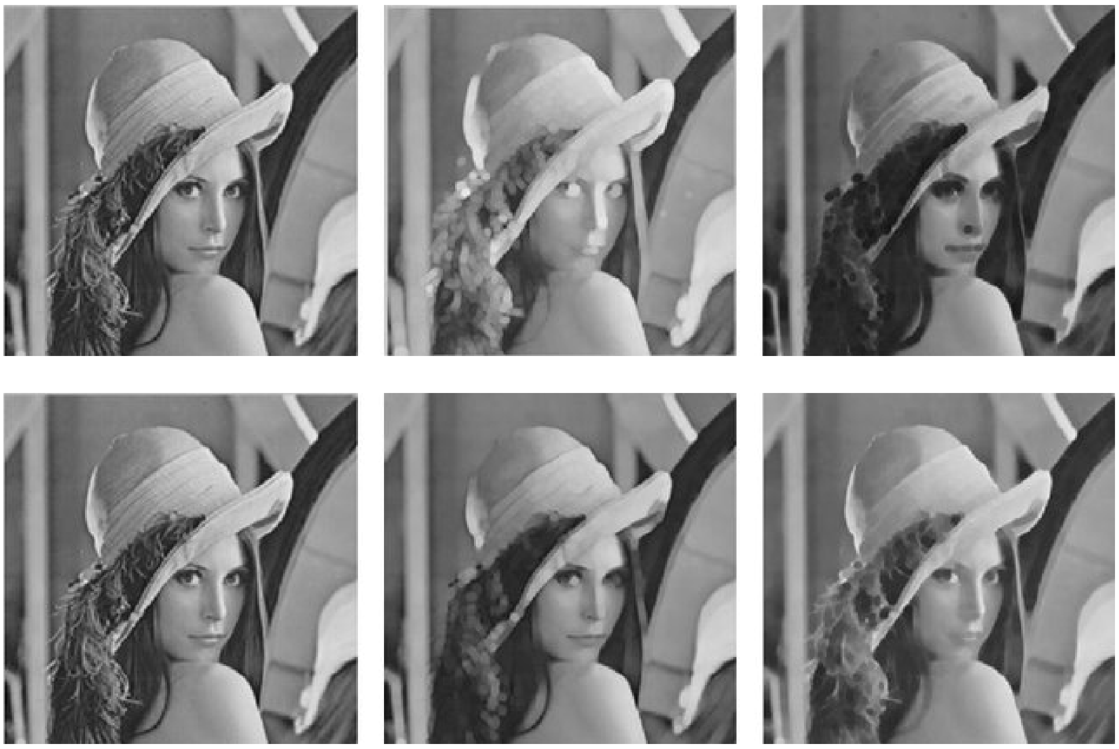


Figure 2.3: Results examples of grayscale dilation (Top Middle), erosion (Top Right), opening (Bottom Middle), closing (Bottom Right). Original image on the left (Top and Bottom). Images from Yu (2004)

Chapter 3

Supervised Learning

Supervised learning has been around for decades with the objective of automating complex processes like deduction, decision, distinction, etc. This is the second half of this subject, and some specificity will be depicted with the objective of giving more details about how a neural network is actually learning and why it is problematic when trying to integrate morphological filters within.

3.1 Preliminaries

Given an input, a neural network (an actual program with a lot of interconnected parameters) will compute an output. From this output, an error can be calculated by comparing it to the expected output; the main idea will be to minimize this error by adjusting the program's parameters. This error is a function and minimizing it is in fact finding the global minimum of it, which corresponds to finding the ideal parameters. To modify the parameters in the correct way, a method called *Gradient Descent* is used.

The idea behind this method is to compute and use the derivative of the error function to have an indication about its local curve and be able to follow a descending slope (with the expectation of going towards the minimum).

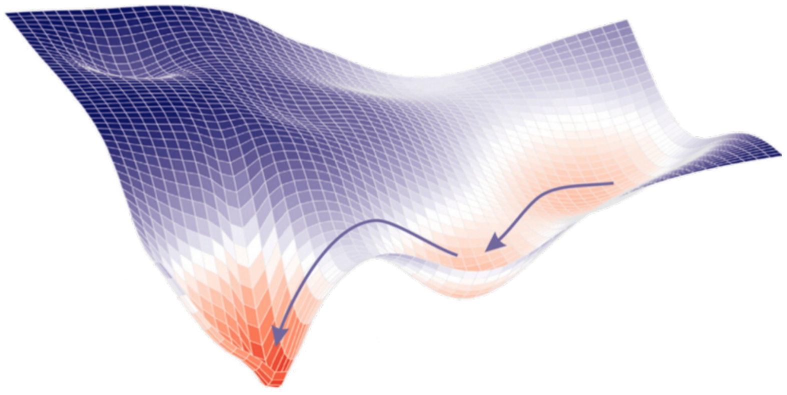


Figure 3.1: Illustration of the gradient descent method from PRIMO.ai

3.2 Convolutional Neural Networks (CNNs)

When talking about object detection, image analysis, image classification and essentially every algorithm that tries to apply the conveniences of deep learning and back-propagation in signal related fields, Convolutional Neural Networks (CNNs) appear to be the best contenders. Throughout the years, many variants were proposed with architectures and layer arrangements very peculiar and different, but the principle stays the same with the application of a convolution kernel (or filter, see Figure 3.2), which is updated and learned throughout the training process.

Convolution based networks were at the root of the deep learning breakthrough in image processing techniques (Russakovsky et al., 2015), in which the convolution layer plays an important role as an automatic feature extraction process. However, the convolution, which is a weighted summation within the sliding window, merely captures the linear features, resulting in losing nonlinear information regarding image content, such as object position, meaningful region, shape, and size (Shih et al., 2019).

Morphological operators are likewise very efficient at extracting features (topological and geometrical, such as shape, size, connectivity, and distance), but also useful in image enhancing processes like denoising. As said in the introduction, the biggest problem of using morphological operators is finding (in a trial-and-error fashion) the right operations, optimal sequence of them, and arranging the shape of their corresponding structuring element, since everything depends on the data and the objectives; thus, having a network learn everything instead would also solve these problems, presumably increase morphological operations power. In this way, morphological operators could be great contenders to convolutions in the supervised learning field.

Nicely enough, if the filters are thought as structuring elements, only the convolution operation needs to be replaced by a morphological operator to get a Deep Morphological Neural Network (DMNN). It would also directly integrate non-linearity in the network, which is essential for great generalization. However, one problem stands out: remembering the base equations of grayscale erosion and dilation (2.4, 2.5), they use *min* and *max* functions which are ***non-derivable***, but as said in the previous section the derivatives of the functions used in the network are needed to benefit from the gradient descent method.

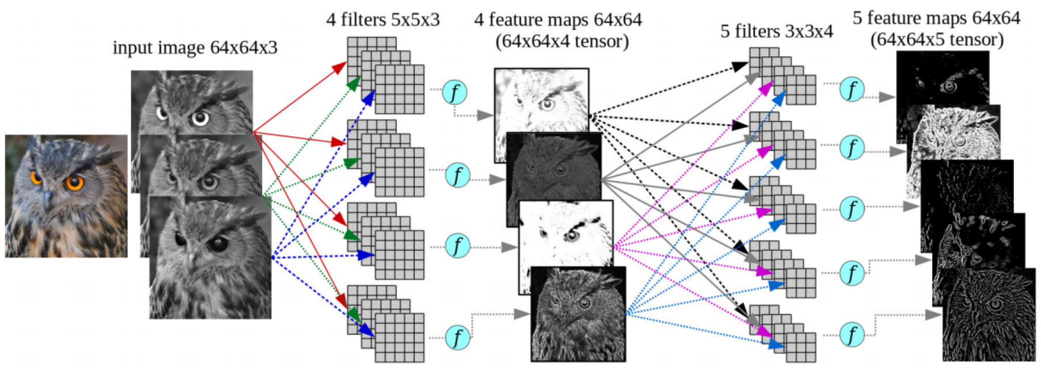


Figure 3.2: Convolution layers and the associated feature maps concept illustration, from Ponti et al. (2017)

3.3 Problematic and Environment

The biggest problem with the objective of replacing convolution layers by morphological layers is therefore the non-derivability of the *min* and *max* functions which makes it impossible not to disturb the gradient descent flow; that is why here we are trying to find the best way to approximate those.

To find the best approximation function, the best suited for its integration in a deep neural network, the fundamental step is to test the encountered possibilities. After choosing an approximation function, like the counter-harmonic mean (CHM) and the α-softmax which will be presented afterwards, their potency is being tested within a network (see Figures 3.3 and 3.4) which was structured the simplest way possible: the main idea is to apply on various images, from a database like MNIST (LeCun et al., 1989), morphological operations with known structuring elements to create target images for the network, and the network is asked to find a way to generate those target images from original images. In this sense, the network is evaluated on its capacity to learn the correct operation (erosion, dilation, opening and closing for now) and structuring elements which were used to get the target images. The size of the network and the number of operations and filters to learn depend on the order of the operation used to create the target images.

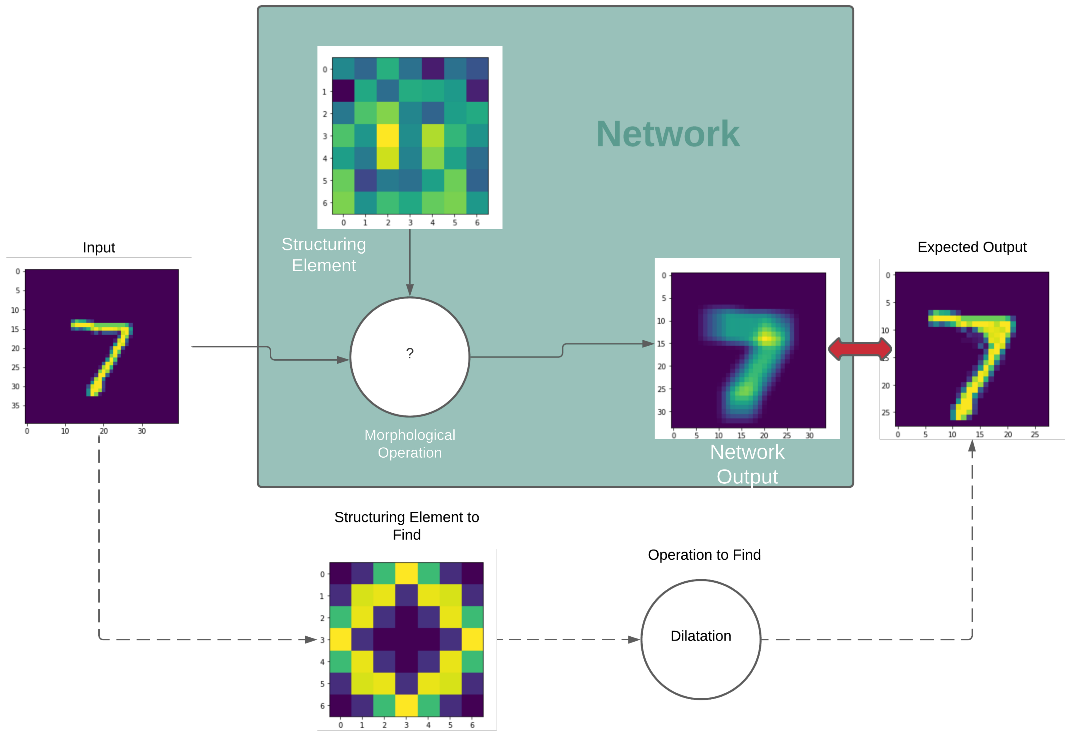


Figure 3.3: Schema of the training technique and the used network for simple operations (erosion and dilation). Example for a dilation operation.

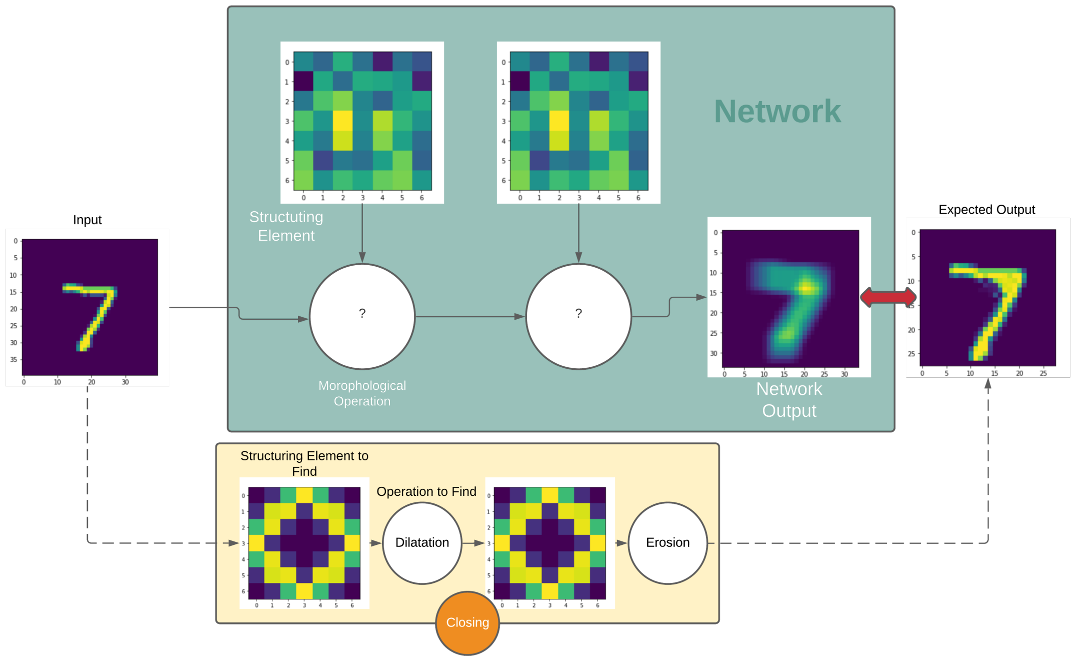


Figure 3.4: Schema of the training technique and the used network for composed operations (opening and closing). Example for a closing operation.

Chapter 4

Prior Results

This subject is based on previous work and experiences, in this section we will expose some background knowledge. We will go over multiple propositions and results of approximating functions, essentially a reference and the two proposed in the paper my work is based on. Furthermore, results exposed in this previous paper will be recapped and examined.

4.1 First Approximation With Counter Harmonic Mean (CHM)

The p-convolution layer ($PConv$) was proposed by Masci et al. in 2013. It is a morphological layer that uses the Counter Harmonic Mean (CHM) (Equation 4.1), also known as Leamer Mean, to approximate *min* and *max* functions. It serves as a reference in this work as it pioneered the use of morphological operators in neural networks (Hu et al., 2021), and is a great result reference as other papers (Shen et al., 2019; Shih et al., 2019) also trying to incorporate and exploit *min* and *max* approximation functions in a Deep Morphological Neural Networks (DMNN) will very likely be inspired by it and/or compare to it as well.

The CHM has a parameter, p in the following definitions, which as it tends to one infinity (positive or negative) approximates the maximum or minimum of a set of values (equations 4.2 and 4.3) which can be weighted.

For $n \in \mathbb{N}$, $\forall i \in \{1, 2, ..., n\}$ and $y_i, w_i \in \mathbb{R}^+$, we can define $Y := \{y_1, y_2, ..., y_n\}$; $\forall p \in \mathbb{R}$ the Lehmer Mean is defined as:

$$L_{p,w}(Y) := \frac{\sum_{k=1}^{n} w_k . y_k^p}{\sum_{k=1}^{n} w_k . y_k^{p-1}} \tag{4.1}$$

With the resulting properties being:

$$\lim_{p \to +\infty} (L_{p,w}(Y)) = max(Y), \tag{4.2}$$

$$\lim_{p \to -\infty} (L_{p,w}(Y)) = min(Y) \tag{4.3}$$

In the case of p-convolutions, weights are the values of the convolution kernel (or structuring element). The p-convolution of an image f at pixel x for a given (positive) convolution kernel

$w : W \subseteq E \rightarrow \mathbb{R}^+$ (with $E \subseteq \mathbb{Z}^2$, the image definition space) and the parameter $p \in \mathbb{R}$ is defined as:

$$PConv(f, w, p)(x) = (f *_p w)(x) = \frac{(f^{p+1} * w)(x)}{(f^p * w)(x)} = \frac{\sum_{y \in W(x)} f^{p+1}(y) w(x-y)}{\sum_{y \in W(x)} f^p(y) w(x-y)} \tag{4.4}$$

And from that we can recover erosion and dilation approximations:

$$\lim_{p \to +\infty} PConv(f, w, p)(x) = \sup_{y \in W(x)} \{f(y) + \frac{1}{p} \log(w(x-y))\} = (f \oplus \frac{1}{p} \log(w))(x) \tag{4.5}$$

$$\lim_{p \to -\infty} PConv(f, w, p)(x) = \inf_{y \in W(x)} \{f(y) - \frac{1}{p} \log(w(x-y))\} = (f \ominus \frac{1}{p} \log(w))(x) \tag{4.6}$$

As we can see in equations 4.5 and 4.6, we can recover erosion and dilation functions (Equations 2.4 and 2.5) from the definitions (demonstrated in Angulo (2010)). As depicted in their paper, the integration of the $PConv$ layer in a learning framework was a successful attempt, especially for a denoising process. However, the $PConv$ layer have some pitfalls:

- $f(x) = 0$ and $p < 0 \Rightarrow f^p(x) \equiv NaN$
- $f(x) < 0$ and $p \in \mathbb{R} \Rightarrow f^p(x) \in \mathbb{C}$
- If $w(x) = 0$ or $f^p(x) = 0 \Rightarrow \frac{1}{(f^p * w)(x)} \equiv NaN$

To counter this downsides, a rescale of input images between $[1, 2]$ before feeding them to any $PConv$ layer is required. Also, from the results of the layer (analyzed in the original paper), a tendency to learn hollow structuring elements can be discerned.

4.2 Proposed Layers

As evoked in Chapter 1, this work is the sequel of Kirszenberg et al. paper, Going beyond p-convolutions to learn grayscale morphological operators, published in 2021. In this paper, two morphological layers are proposed, $\mathcal{L}Morph$ and $\mathcal{S}Morph$.

4.2.1 $\mathcal{L}Morph$

The $\mathcal{L}Morph$ layer is a simpler approach of the use of the Counter-harmonic mean: instead of only having $f(y)$, the pixel value, to the power p, the summation of the pixel value and the structuring element's corresponding value $w(x-y)$ is raised to the power of p. This is easier to define an equivalence with erosion and dilation in this way, as detailed in the paper.

$$\mathcal{L}Morph(f, w, p)(x) = \frac{\sum_{y \in W(x)} (f(y) + w(x-y))^{p+1}}{\sum_{y \in W(x)} (f(y) + w(x-y))^p} \tag{4.7}$$

$$\lim_{p \to +\infty} \mathcal{L}Morph(f, w, p)(x) = \sup_{y \in W(x)} \{f(y) + w(x-y)\} = (f \oplus w)(x) \tag{4.8}$$

$$\lim_{p\to-\infty} \mathcal{L}Morph(f,w,p)(x) = \inf_{y\in W(x)} \{f(y)+w(x-y)\} = (f\ominus -w)(x) \tag{4.9}$$

Using the same base formula of approximation, $\mathcal{L}Morph$ suffers from quite the same issues as the $PConv$ layer concerning domain definition, thus a rescale of input data is needed.

4.2.2 $\mathcal{S}Morph$

This time following another approximation function, the $\alpha-$softmax, the $\mathcal{S}Morph$ layer is presented as a more promising alternative as it does not suffer from any numerical issues, and yields better results most of the time (see section 4.3).

The $\alpha-$softmax function, taking back $Y := y_1, y_2, \ldots, y_n$ defined in section 4.1 and $\alpha \in \mathbb{R}$ is defined as:

$$\mathcal{S}_\alpha(\mathbf{x}) = \frac{\sum_{i=1}^n x_i e^{\alpha x_i}}{\sum_{i=1}^n e^{\alpha x_i}} \tag{4.10}$$

With the approximation properties, similarly characterized as:

$$\lim_{\alpha\to+\infty} \mathcal{S}_\alpha(Y) = \max_i y_i \tag{4.11}$$

$$\lim_{\alpha\to-\infty} \mathcal{S}_\alpha(Y) = \min_i y_i \tag{4.12}$$

The $\mathcal{S}Morph$ layer is accordingly defined as:

$$\mathcal{S}Morph(f,w,\alpha)(x) = \frac{\sum_{y\in W(x)}(f(y)+w(x-y))e^{\alpha(f(y)+w(x-y))}}{\sum_{y\in W(x)} e^{\alpha(f(y)+w(x-y))}} \tag{4.13}$$

With the resulting properties derived as follow:

$$\lim_{\alpha\to+\infty} \mathcal{S}Morph(f,w,\alpha)(x) = (f\oplus w)(x) \tag{4.14}$$

$$\lim_{\alpha\to-\infty} \mathcal{S}Morph(f,w,\alpha)(x) = (f\ominus -w)(x) \tag{4.15}$$

4.3 Results Analysis

The conducted experiments in this paper used the structuring elements of Figure 4.1 to create target images; therefore, they are the expected structuring elements to be learned through the experiences. Furthermore, convergence of parameters p ($PConv$ and $\mathcal{L}Morph$) and α ($\mathcal{S}Morph$) is also looked after, as they are expected to progress towards positive or negative infinity, for dilation and erosion respectively. The training steps were performed on the MNIST database LeCun et al. (1989) (composed of handwritten digits images).

From Kirszenberg et al.'s paper we can retrieve visual and quantitative results: Figure 4.2 shows that, for dilation and erosion, $\mathcal{L}Morph$ and $\mathcal{S}Morph$ work pretty well at learning desired structuring elements and transformations. Table 4.3 shows that the best results when looking

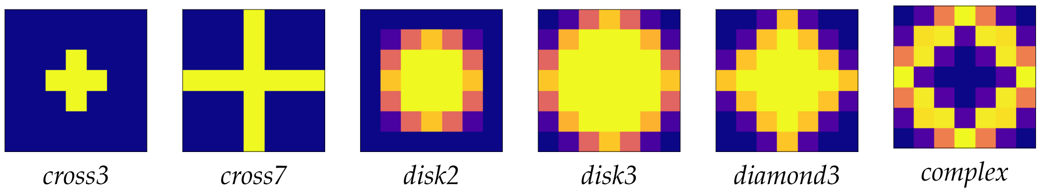


Figure 4.1: 7×7 target grayscale structuring elements. All values range between 0 (deep blue) and 1 (yellow).

at the *Loss* (output comparison) and the RMSE (learned structuring element compared to the expected one) are split between $\mathcal{L}Morph$ and $\mathcal{S}Morph$, both outperforming the $PConv$ layer.

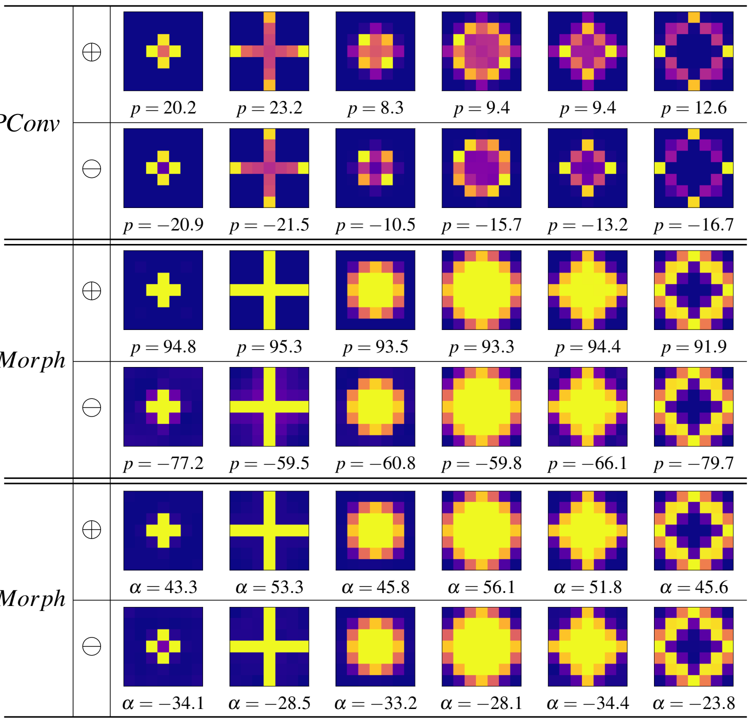


Figure 4.2: Learned structuring element (with corresponding p or α at convergence) for $PConv$, $\mathcal{L}Morph$ and $\mathcal{S}Morph$ layers on dilation $\oplus$ and erosion $\ominus$ tasks.

Now looking at opening and closing results (Figure 4.4 and Table 4.3), we can see that there is more trouble to reach desired convergence, especially for $\mathcal{L}Morph$ with the opening operation. Eventually, $\mathcal{L}Morph$ and $\mathcal{S}Morph$ have globally better statistics than $PConv$, although some experiences showed the layers difficulties to converge toward a great result, whether talking about the parameters or structuring elements.

We will consider those convergence defaults as edge cases, as it is neither a normal nor an expected behavior.

			cross3	*cross7*	*disk2*	*disk3*	*diamond3*	*complex*	
PConv	$\oplus$	LOSS	0.5	0.8	5.1	6.4	6.8	3.3	$\times 10^{-4}$
		RMSE	0.41	1.42	2.22	3.09	2.80	2.38	
	$\ominus$	LOSS	2.4	0.62	13	2.6	5.2	1.2	$\times 10^{-5}$
		RMSE	0.82	1.55	2.82	3.77	3.63	2.76	
$\mathcal{L}$*Morph*	$\oplus$	LOSS	0.84	1.2	0.78	1.2	1.2	2.1	$\times 10^{-5}$
		RMSE	**0.02**	**0.00**	**0.02**	**0.05**	**0.01**	**0.48**	
	$\ominus$	LOSS	1.1	**0.37**	1.3	**0.37**	0.58	**1.1**	$\times 10^{-6}$
		RMSE	**0.44**	0.63	0.37	0.29	0.38	**0.48**	
$\mathcal{S}$*Morph*	$\oplus$	LOSS	**1.8**	**4.0**	**2.5**	**4.9**	**3.9**	**1.7**	$\times \mathbf{10^{-7}}$
		RMSE	0.10	0.13	0.08	0.06	0.09	0.51	
	$\ominus$	LOSS	210	4.1	**0.8**	4.1	**4.3**	**9.5**	$\times \mathbf{10^{-7}}$
		RMSE	0.78	**0.15**	**0.09**	**0.05**	**0.09**	0.51	

Figure 4.3: MSE loss at convergence and RMSE between the learned structuring element displayed by Figure 4.2 and the target for $PConv$, $\mathcal{L}Morph$ and $\mathcal{S}Morph$ layers on dilation $\oplus$ and erosion $\ominus$ tasks. Best (lowest) results are in bold.

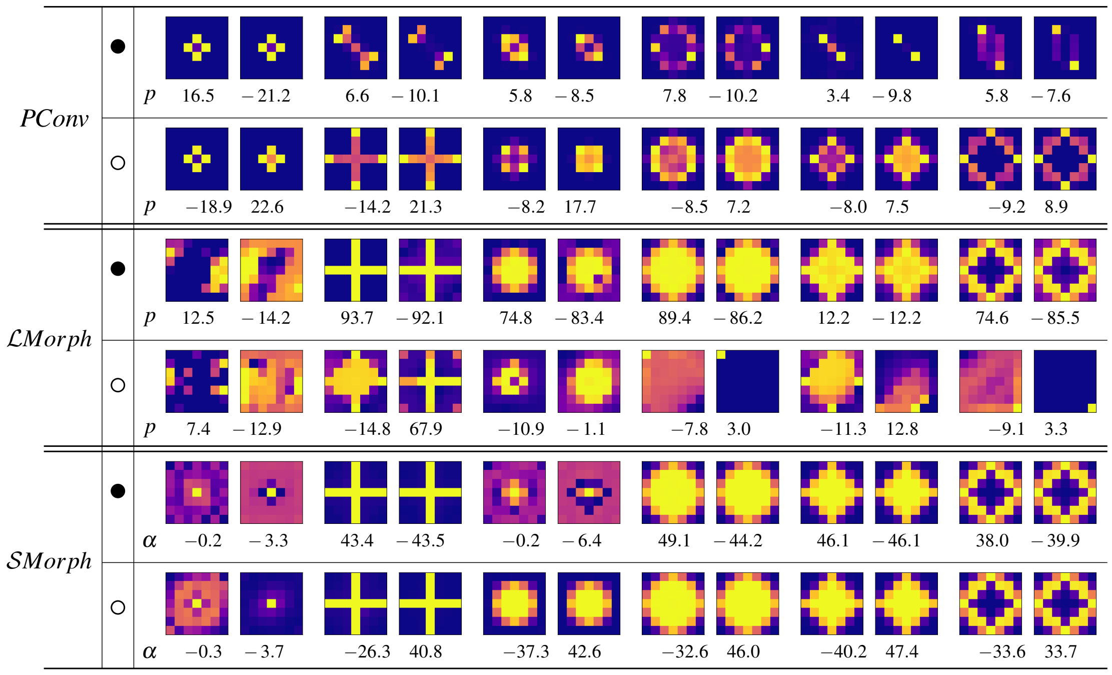


Figure 4.4: Learned structuring elements (with corresponding p or α value for each layer) for $PConv$, $\mathcal{L}Morph$ and $\mathcal{S}Morph$ layers on closing $\bullet$ and opening $\circ$ tasks.

			cross3	*cross7*	*disk2*	*disk3*	*diamond3*	*complex*	
PConv	●	LOSS	0.09	5.1	1.7	1.0	5.4	4.9	$\times 10^{-3}$
		RMSE	**0.83** | **0.81**	3.94 | 3.97	2.82 | 3.07	4.44 | 4.64	4.34 | 4.51	3.68 | 3.65	
	○	LOSS	**0.86**	0.76	4.8	3.9	4.4	2.3	$\times 10^{-4}$
		RMSE	**0.91** | **0.32**	1.45 | 0.87	2.70 | 2.49	3.10 | 2.52	3.15 | 2.43	2.10 | 2.08	
$\mathcal{L}$*Morph*	●	LOSS	**0.5**	0.13	**2.0**	0.18	6.2	0.14	$\times 10^{-4}$
		RMSE	3.58 | 5.12	**0.01** | 0.63	**0.14** | **1.16**	**0.07** | 0.40	0.75 | 0.95	**0.47** | 0.99	
	○	LOSS	8.7	36	2.0	2.7	4.6	2.5	$\times 10^{-3}$
		RMSE	3.17 | 4.74	2.90 | 1.20	1.99 | 1.61	2.81 | 5.56	1.66 | 3.54	2.72 | 3.98	
$\mathcal{S}$*Morph*	●	LOSS	1700	**0.31**	4400	**0.42**	**0.37**	**0.17**	**$\times 10^{-6}$**
		RMSE	2.10 | 3.59	0.14 | **0.08**	2.78 | 3.68	0.08 | **0.01**	**0.08** | **0.10**	0.52 | **0.52**	
	○	LOSS	4700	**0.14**	**0.22**	**0.25**	**0.24**	**0.09**	**$\times 10^{-6}$**
		RMSE	3.40 | 1.63	**0.18** | **0.02**	**0.05** | **0.01**	**0.02** | **0.02**	**0.02** | **0.01**	**0.51** | **0.52**	

Figure 4.5: MSE loss at convergence and RMSE between the learned structuring elements displayed by Figure 4.4 and the target for $PConv$, $\mathcal{L}Morph$ and $\mathcal{S}Morph$ layers on closing $\bullet$ and opening $\circ$ tasks. Best (lowest) results are in bold.

Chapter 5

Work, Experiments and Results

One of the main purpose of my work this semester has been to investigate those edge cases, trying to find why there were problems in the learning process and potentially find solutions to remedy those complications. In this part I will detail my working process throughout this first half of the year, going from the assimilation of the subject to experiments field extend, passing by the investigation regarding those corner cases.

5.1 Prerequisites

First of all, this subject comes with two big imperatives, that are being aware of (grayscale) mathematical morphology and supervised learning. I discovered mathematical morphology through definitions, examples and experiences in `Python`, but also when searching for references on the subject. For the supervised learning part, I already had the basics and only re-documented myself to strengthen the theory.

With enough theory, documentation, knowledge and practice, I would now start looking at different libraries such as `scikit-learn`, `SciPy`, `PyTorch` and `CUDA` which are essential for the implementation leading the theory of the subject.

Also, the code used to produce the results of Kirszenberg et al.'s paper was given to me; I had to fully understand it, which was a heavy task (but rather simple compared to re-coding everything) as it is roughly about 1.3k lines of code in `Python` (use of `PyTorch`, `SciPy`, etc.), and 650 lines of `C++` (GPU programming with `CUDA` and a `Python` bridge part with `PyTorch`). Still, the code was not working so I had to debug it to make it run.

5.2 Experiences Reproduction

What had not been done with the original experiments was to repeat the learning procedures to have a more accurate vision on the actual results and more rigorous conclusions over the proposed layers.

With the success of running the given code, I have been able to write scripts to automate runs of learning to have a broader database and have realistic results. I firstly did ten runs with the three different layers for the four operations and the six different structuring elements, which translate to $10 * 3 * 4 * 6 = 720$ base runs. Considering the time of each run (it hugely depends on the operation and the layer) being between twenty minutes and one hour, I considered it to be enough to make general statistics about the learning behaviors and processes.

In figures 5.1 and 5.2 we can see that the results I had for erosion and dilation through those runs are sensibly similar to the ones exposed in the original paper, which is what I was expecting, and very stable; those results were comforting in the idea of having a solid base for further work.

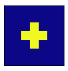
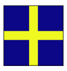
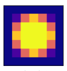
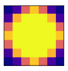
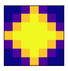
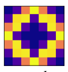
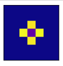
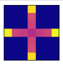
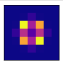
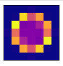
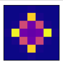
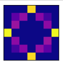
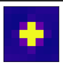
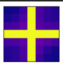
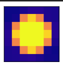
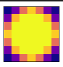
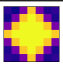
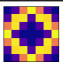
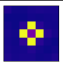
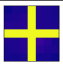
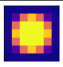
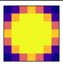
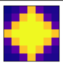
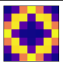
		 *cross3*	 *cross7*	 *disk2*	 *disk3*	 *diamond3*	 *complex*
PConv	*w*						
	p	-20.86 ± 0.45	-20.35 ± 0.20	-10.52 ± 0.004	-15.64 ± 0.06	-13.21 ± 0.02	-16.45 ± 0.07
	RMSE	$0.82 \pm 7\times10^{-4}$	$1.55 \pm 1\times10^{-3}$	$2.82 \pm 1\times10^{-4}$	$3.77 \pm 6\times10^{-4}$	$3.64 \pm 1\times10^{-4}$	$3.19 \pm 6\times10^{-3}$
	LOSS	$2.4\times10^{-5} \pm 1\times10^{-6}$	$6.2\times10^{-6} \pm 1\times10^{-7}$	$1.3\times10^{-4} \pm 1\times10^{-8}$	$2.6\times10^{-5} \pm 5\times10^{-8}$	$5.2\times10^{-5} \pm 2\times10^{-8}$	$1.2\times10^{-5} \pm 4\times10^{-8}$
	EPOCHS	209 ± 45	169 ± 22	49 ± 3	92 ± 7	66 ± 8	130 ± 18
$\mathscr{L}$*Morph*	*w*						
	p	-77.09 ± 0.15	-59.58 ± 0.20	-67.78 ± 0.30	-59.46 ± 0.18	-66.25 ± 0.17	-79.36 ± 0.93
	RMSE	$\mathbf{0.44} \pm 3\times10^{-4}$	$0.60 \pm 1\times10^{-2}$	$0.37 \pm 4\times10^{-2}$	$0.30 \pm 7\times10^{-3}$	$0.38 \pm 4\times10^{-2}$	$\mathbf{0.04} \pm 1\times10^{-3}$
	LOSS	$\mathbf{1.1\times10^{-6}} \pm 5\times10^{-9}$	$\mathbf{3.6\times10^{-7}} \pm 4\times10^{-9}$	$1.3\times10^{-6} \pm 6\times10^{-8}$	$\mathbf{3.7\times10^{-7}} \pm 4\times10^{-9}$	$5.8\times10^{-7} \pm 3\times10^{-9}$	$1.1\times10^{-6} \pm 5\times10^{-8}$
	EPOCHS	158 ± 4	164 ± 20	146 ± 16	256 ± 3	231 ± 9	876 ± 139
$\mathscr{S}$*Morph*	*w*						
	α	-33.95 ± 0.03	-28.52 ± 0.013	-30.61 ± 0.016	-28.10 ± 0.006	-34.38 ± 0.003	-23.79 ± 0.011
	RMSE	$0.78 \pm 2\times10^{-2}$	$\mathbf{0.18} \pm 3\times10^{-2}$	$\mathbf{0.10} \pm 2\times10^{-3}$	$\mathbf{0.04} \pm 2\times10^{-2}$	$\mathbf{0.13} \pm 4\times10^{-2}$	$0.08 \pm 2\times10^{-2}$
	LOSS	$2.1\times10^{-5} \pm 5\times10^{-9}$	$4.0\times10^{-7} \pm 8\times10^{-9}$	$\mathbf{4.6\times10^{-7}} \pm 8\times10^{-9}$	$4.2\times10^{-7} \pm 6\times10^{-9}$	$\mathbf{4.3\times10^{-7}} \pm 3\times10^{-9}$	$\mathbf{9.5\times10^{-7}} \pm 7\times10^{-9}$
	EPOCHS	40 ± 4	45 ± 4	38 ± 8	38 ± 7	42 ± 5	55 ± 9

Figure 5.1: Learned filter w, corresponding parameter p/α, RMSE between the learned filter and the target structuring element, MSE training loss at convergence and number of training epochs for $PConv$, $\mathcal{L}Morph$ and $\mathcal{S}Morph$ layers on an erosion $\ominus$ task. Reported values correspond to the average $\pm$ standard deviation over the 5 runs. Best (lowest) results are in bold.

Now focusing on opening and closing operations results which can be seen in figures 5.4 and 5.3, the output is also similar and even better than the one in the paper, which is very reassuring especially for $\mathcal{L}Morph$ and the opening operation.

With those runs, I could not reproduce the errors for the $\mathcal{L}Morph$ layer and the opening operation introduced in Kirszenberg et al. (2021). While I was investigating on edge cases (see section 5.3), I found similarities between the aspects of the structuring elements half through the learning process (see Figure 5.6) and the learning failures, which led me to the conclusion that those edge cases were probably caused by a problem during the runs, such as a premature stop of the learning process. As a consequence, I was able to concentrate my investigation process on permanent edge cases, which still are a problem to solve.

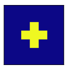
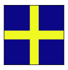
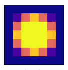
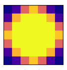
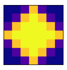
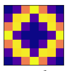
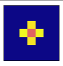
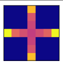
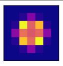
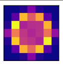
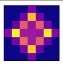
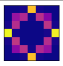
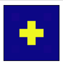
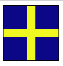
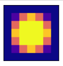
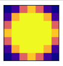
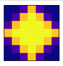
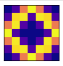
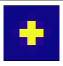
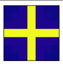
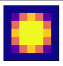
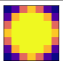
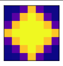
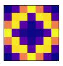
		 *cross3*	 *cross7*	 *disk2*	 *disk3*	 *diamond3*	 *complex*
PConv	w						
	p	19.87 ± 0.23	22.58 ± 0.44	8.27 ± 0.002	9.43 ± 0.002	9.39 ± 0.001	12.57 ± 0.005
	RMSE	$0.41 \pm 2\times10^{-3}$	$1.42 \pm 6\times10^{-4}$	$2.22 \pm 3\times10^{-4}$	$3.05 \pm 3\times10^{-4}$	$2.79 \pm 3\times10^{-4}$	$2.81 \pm 7\times10^{-5}$
	LOSS	$4.8\times10^{-5} \pm 1\times10^{-6}$	$9.0\times10^{-5} \pm 4\times10^{-6}$	$5.1\times10^{-4} \pm 5\times10^{-8}$	$6.3\times10^{-4} \pm 6\times10^{-9}$	$6.8\times10^{-4} \pm 5\times10^{-9}$	$3.3\times10^{-4} \pm 1\times10^{-8}$
	EPOCHS	194 ± 17	222 ± 38	46 ± 8	41 ± 3	38 ± 4	70 ± 7
$\mathscr{L}$*Morph*	w						
	p	94.92 ± 0.18	95.89 ± 0.29	94.16 ± 0.17	94.25 ± 0.96	94.67 ± 0.31	91.27 ± 0.64
	RMSE	$\mathbf{0.02} \pm 9\times10^{-5}$	$\mathbf{0.003} \pm 8\times10^{-6}$	$\mathbf{0.01} \pm 8\times10^{-5}$	$\mathbf{0.05} \pm 1\times10^{-4}$	$\mathbf{0.01} \pm 2\times10^{-4}$	$\mathbf{0.04} \pm 3\times10^{-4}$
	LOSS	$8.4\times10^{-6} \pm 4\times10^{-8}$	$1.1\times10^{-5} \pm 9\times10^{-8}$	$7.6\times10^{-6} \pm 4\times10^{-8}$	$1.2\times10^{-5} \pm 3\times10^{-7}$	$1.1\times10^{-5} \pm 1\times10^{-7}$	$2.1\times10^{-5} \pm 2\times10^{-7}$
	EPOCHS	119 ± 12	159 ± 22	206 ± 11	193 ± 27	206 ± 9	295 ± 35
$\mathscr{S}$*Morph*	w						
	α	41.97 ± 0.016	52.01 ± 0.027	40.75 ± 0.014	50.06 ± 0.044	49.47 ± 0.04	32.79 ± 0.021
	RMSE	$0.1 \pm 3\times10^{-3}$	$0.14 \pm 2\times10^{-3}$	$0.09 \pm 4\times10^{-4}$	$0.08 \pm 1\times10^{-3}$	$0.09 \pm 2\times10^{-3}$	$0.05 \pm 1\times10^{-2}$
	LOSS	$\mathbf{8.5\times10^{-7}} \pm 3\times10^{-8}$	$\mathbf{1.5\times10^{-6}} \pm 2\times10^{-8}$	$\mathbf{1.2\times10^{-6}} \pm 1\times10^{-8}$	$\mathbf{1.8\times10^{-6}} \pm 4\times10^{-8}$	$\mathbf{1.5\times10^{-6}} \pm 9\times10^{-9}$	$\mathbf{1.6\times10^{-6}} \pm 2\times10^{-8}$
	EPOCHS	39 ± 8	39 ± 9	44 ± 9	45 ± 4	44 ± 4	49 ± 7

Figure 5.2: Learned filter w, corresponding parameter p/α, RMSE between the learned filter and the target structuring element, MSE training loss at convergence and number of training epochs for $PConv$, $\mathcal{L}Morph$ and $\mathcal{S}Morph$ layers on a dilation $\oplus$ task. Reported values correspond to the average $\pm$ standard deviation over the 5 runs. Best (lowest) results are in bold.

5.3 Edge Cases Study

5.3.1 Structuring Elements Learning Troubles

To find the causes of learning failures, I had to better understand the learning process; along these lines I started to look at the evolution of the learned structuring element during the runs. It appeared (see Figure 5.6) that the structuring elements are learned from the outside toward the inside. To be more specific, the convergence of the learned structuring elements seems to start at the edges of the filter and then continues at the center. This observation leads to the hypothesis of the structuring elements being harder to learn when they are not *touching* the borders of the filter.

To test this assumption, I tried to add margins to structuring elements that *touched* the edges, and remove them for those not touching. The result of these experiments can be seen in Figure 5.6.

The outcome of these experiments seems to show that adding margins to structuring elements or removing them influences the learning process. The more there are margins, the harder it seems for it to be learned. Although the opening with a *cross* 3×3 and $\mathcal{L}Morph$, closing with *cross* 3×3 and *disk2* 5×5 for $\mathcal{S}Morph$ are failures, the others show improvements in the learning process, which shows that removing the borders helps the structuring elements to be learned.

Nevertheless, it would need more tests to have a more rigorous conclusion, such as testing the learning process with larger *touching* structuring element, because enlarging the structuring elements may cause troubles for the learning process as the input images are small (25×25 without padding), or testing with other smaller structuring elements, as smaller than the usual

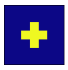
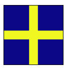
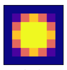
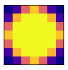
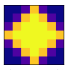
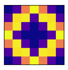
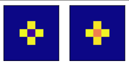
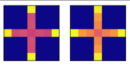
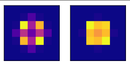
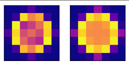
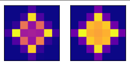
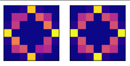
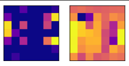
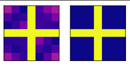
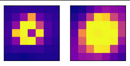
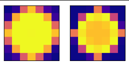
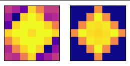
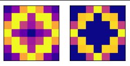
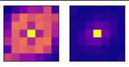
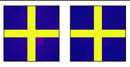
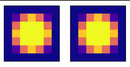
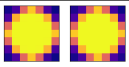
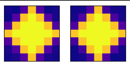
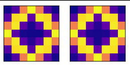
		 *cross3*	 *cross7*	 *disk2*	 *disk3*	 *diamond3*	 *complex*
PConv	w_1/w_2						
	p_1	-18.65 ± 0.29	-14.17 ± 0.32	-8.13 ± 0.04	-8.59 ± 0.06	-8.05 ± 0.01	-9.26 ± 0.15
	p_2	22.11 ± 0.35	21.21 ± 0.47	17.72 ± 0.15	7.03 ± 0.10	7.49 ± 0.01	8.87 ± 0.10
	RMSE$_1$	$\mathbf{0.91} \pm 4\times10^{-4}$	$1.45 \pm 1\times10^{-3}$	$2.69 \pm 7\times10^{-3}$	$3.13 \pm 3\times10^{-2}$	$3.17 \pm 5\times10^{-3}$	$2.53 \pm 3\times10^{-2}$
	RMSE$_2$	$\mathbf{0.32} \pm 1\times10^{-3}$	$0.87 \pm 2\times10^{-3}$	$2.49 \pm 2\times10^{-3}$	$2.51 \pm 1\times10^{-2}$	$2.43 \pm 4\times10^{-3}$	$2.49 \pm 1\times10^{-2}$
	LOSS	$\mathbf{8.9\times10^{-5}} \pm 3\times10^{-6}$	$7.5\times10^{-5} \pm 1\times10^{-6}$	$4.9\times10^{-4} \pm 2\times10^{-6}$	$3.9\times10^{-4} \pm 2\times10^{-5}$	$4.4\times10^{-4} \pm 4\times10^{-6}$	$2.2\times10^{-4} \pm 6\times10^{-6}$
	EPOCHS	167 ± 43	114 ± 8	61 ± 8	47 ± 13	50 ± 5	27 ± 12
$\mathscr{L}$Morph	w_1/w_2						
	p_1	$\mathbf{7.39} \pm 0.14$	-22.68 ± 0.57	-10.74 ± 0.14	-10.88 ± 0.11	-12.82 ± 0.08	-9.54 ± 0.03
	p_2	$\mathbf{-12.89} \pm 0.06$	66.71 ± 0.91	$\mathbf{-1.07} \pm 0.003$	8.18 ± 0.08	12.55 ± 0.07	9.79 ± 0.10
	RMSE$_1$	$3.17 \pm 2\times10^{-3}$	$1.02 \pm 6\times10^{-2}$	$1.99 \pm 7\times10^{-4}$	$0.17 \pm 8\times10^{-3}$	$1.92 \pm 5\times10^{-2}$	$1.15 \pm 8\times10^{-2}$
	RMSE$_2$	$4.72 \pm 8\times10^{-3}$	$\mathbf{0.02} \pm 3\times10^{-4}$	$1.61 \pm 8\times10^{-3}$	$1.80 \pm 4\times10^{-2}$	$0.70 \pm 4\times10^{-3}$	$0.85 \pm 8\times10^{-3}$
	LOSS	$8.7\times10^{-3} \pm 9\times10^{-6}$	$4.3\times10^{-5} \pm 2\times10^{-6}$	$2.0\times10^{-3} \pm 1\times10^{-5}$	$3.8\times10^{-4} \pm 1\times10^{-5}$	$3.2\times10^{-4} \pm 4\times10^{-6}$	$2.3\times10^{-4} \pm 2\times10^{-6}$
	EPOCHS	51 ± 6	111 ± 8	165 ± 42	21 ± 5	46 ± 6	44 ± 4
$\mathscr{S}$Morph	w_1/w_2						
	α_1	$\mathbf{-0.36} \pm 0.0009$	-24.71 ± 0.16	-34.71 ± 0.01	-29.15 ± 0.07	-38.36 ± 0.08	-18.69 ± 0.02
	α_2	$\mathbf{-3.72} \pm 0.05$	40.62 ± 0.01	41.38 ± 0.008	45.41 ± 0.02	47.13 ± 0.02	16.08 ± 0.02
	RMSE$_1$	$3.30 \pm 1\times10^{-2}$	$\mathbf{0.22} \pm 1\times10^{-2}$	$\mathbf{0.06} \pm 8\times10^{-3}$	$\mathbf{0.04} \pm 8\times10^{-3}$	$\mathbf{0.06} \pm 5\times10^{-2}$	$\mathbf{0.11} \pm 3\times10^{-2}$
	RMSE$_2$	$1.63 \pm 5\times10^{-3}$	$0.06 \pm 3\times10^{-2}$	$\mathbf{0.03} \pm 1\times10^{-2}$	$\mathbf{0.05} \pm 1\times10^{-2}$	$\mathbf{0.06} \pm 2\times10^{-2}$	$\mathbf{0.11} \pm 3\times10^{-2}$
	LOSS	$4.8\times10^{-3} \pm 4\times10^{-6}$	$\mathbf{4.6\times10^{-7}} \pm 9\times10^{-8}$	$\mathbf{5.5\times10^{-7}} \pm 4\times10^{-8}$	$\mathbf{6.2\times10^{-7}} \pm 8\times10^{-8}$	$\mathbf{5.5\times10^{-7}} \pm 4\times10^{-8}$	$\mathbf{1.8\times10^{-6}} \pm 4\times10^{-8}$
	EPOCHS	22 ± 3	41 ± 5	35 ± 3	42 ± 4	42 ± 6	42 ± 4

Figure 5.3: Learned filters w_i, corresponding parameter p_i/α_i and RMSE$_i$ between the learned filter and the target structuring element for both layers ($i \in \{1,2\}$), MSE training loss at convergence and number of training epochs for $PConv$, $\mathcal{L}Morph$ and $\mathcal{S}Morph$ layers on an opening $\circ$ task. Best (lowest) results are in bold. Abnormal results are in red.

7×7 could just be easier to learn. Also, a special structuring element composed of two disks, one touching the edges and one inside the first touching neither the edges nor the first disk could lead to interesting results.

5.3.2 Architecture Inherent Range Problem

One problem which does not clearly appear in previous shown results is the actual learned values of the structuring elements. The ones used (Figure 4.1) have values between 0 and 1, but even if the visual aspect of the learned structuring elements (usually displayed with the color gradient depending on the minimum and the maximum values of the structuring element) is very closed to the expected one are very similar, the values are not between 0 and 1. This caused the problem of not fully understanding how the networks could produce a correct output and not knowing how to truly interpret the resulting structuring elements and compare them to the ground truth.

The first thing to notice, taking back equations (4.9) and (4.15), is the minus sign in the formula of the erosion $(f \ominus -w)(x)$, with w being the structuring element. Because of this condition, when learning an erosion the network has to learn $-w$ to compensate everything and have an

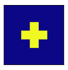
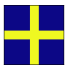
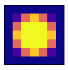
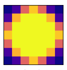
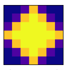
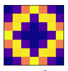
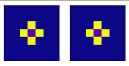
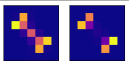
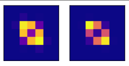
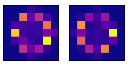
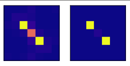
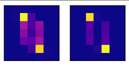
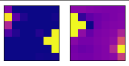
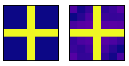
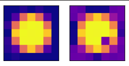
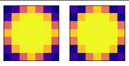
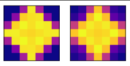
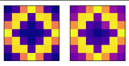
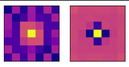
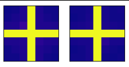
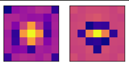
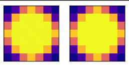
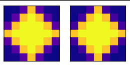
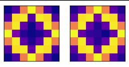
		 *cross3*	 *cross7*	 *disk2*	 *disk3*	 *diamond3*	 *complex*
PConv	w_1/w_2						
	p_1	16.36 ± 0.12	5.99 ± 0.5	5.78 ± 0.004	7.85 ± 0.21	3.43 ± 0.002	5.84 ± 0.003
	p_2	-20.95 ± 0.14	$-8.07 \pm \mathbf{1.65}$	-8.54 ± 0.002	-10.22 ± 0.36	-9.73 ± 0.04	7.64 ± 0.002
	RMSE$_1$	$\mathbf{0.82} \pm 6{\times}10^{-4}$	$3.82 \pm 1{\times}10^{-1}$	$2.82 \pm 2{\times}10^{-4}$	$4.47 \pm 3{\times}10^{-3}$	$4.34 \pm 5{\times}10^{-5}$	$4.16 \pm 3{\times}10^{-5}$
	RMSE$_2$	$\mathbf{0.81} \pm 1{\times}10^{-3}$	$3.87 \pm 9{\times}10^{-2}$	$3.07 \pm 3{\times}10^{-4}$	$4.63 \pm 2{\times}10^{-1}$	$4.51 \pm 6{\times}10^{-5}$	$4.13 \pm 4{\times}10^{-6}$
	LOSS	$\mathbf{8.9{\times}10^{-5}} \pm 1{\times}10^{-6}$	$6.5{\times}10^{-3} \pm 7{\times}10^{-4}$	$1.7{\times}10^{-3} \pm 6{\times}10^{-8}$	$1.0{\times}10^{-3} \pm 6{\times}10^{-5}$	$5.4{\times}10^{-3} \pm 5{\times}10^{-7}$	$5.0{\times}10^{-3} \pm 1{\times}10^{-7}$
	EPOCHS	194 ± 21	56 ± 8	50 ± 4	60 ± 7	35 ± 8	49 ± 6
$\mathcal{L}$*Morph*	w_1/w_2						
	p_1	$46.55 \pm \mathbf{29.7}$	93.67 ± 0.43	$73.33 \pm \mathbf{2.49}$	89.50 ± 0.55	12.19 ± 0.009	$72.29 \pm \mathbf{1.84}$
	p_2	$-41.33 \pm \mathbf{22.2}$	-92.077 ± 0.38	$-82.37 \pm \mathbf{1.84}$	-86.29 ± 0.26	-12.24 ± 0.02	$-84.01 \pm \mathbf{1.32}$
	RMSE$_1$	$3.17 \pm 2{\times}10^{-1}$	$\mathbf{0.01} \pm 7{\times}10^{-5}$	$\mathbf{0.14} \pm 3{\times}10^{-3}$	$\mathbf{0.07} \pm 3{\times}10^{-4}$	$0.75 \pm 7{\times}10^{-4}$	$\mathbf{0.05} \pm 1{\times}10^{-3}$
	RMSE$_2$	$3.29 \pm 9{\times}10^{-1}$	$0.63 \pm 2{\times}10^{-3}$	$\mathbf{1.16} \pm 2{\times}10^{-2}$	$0.40 \pm 1{\times}10^{-3}$	$0.95 \pm {}''{\times}10^{-4}$	$0.75 \pm 5{\times}10^{-2}$
	LOSS	$3.7{\times}10^{-3} \pm 6{\times}10^{-4}$	$1.3{\times}10^{-5} \pm 2{\times}10^{-7}$	$\mathbf{2.0{\times}10^{-4}} \pm 1{\times}10^{-6}$	$1.7{\times}10^{-5} \pm 2{\times}10^{-7}$	$6.2{\times}10^{-4} \pm 1{\times}10^{-7}$	$1.5{\times}10^{-5} \pm 8{\times}10^{-7}$
	EPOCHS	72 ± 3	300 ± 22	157 ± 12	123 ± 17	61 ± 8	$329 \pm \mathbf{94}$
$\mathcal{S}$*Morph*	w_1/w_2						
	α_1	$\mathbf{-0.25} \pm 0.0003$	42.47 ± 0.02	$\mathbf{-0.28} \pm 0.002$	41.42 ± 0.26	43.02 ± 0.04	23.82 ± 0.04
	α_2	-3.44 ± 0.01	-41.60 ± 0.02	-6.48 ± 0.02	-39.96 ± 0.06	-42.94 ± 0.03	-29.73 ± 0.04
	RMSE$_1$	$1.99 \pm 1{\times}10^{-3}$	$0.16 \pm 1{\times}10^{-2}$	$2.77 \pm 2{\times}10^{-3}$	$0.09 \pm 1{\times}10^{-3}$	$\mathbf{0.09} \pm 3{\times}10^{-3}$	$0.15 \pm 4{\times}10^{-2}$
	RMSE$_2$	$3.77 \pm 3{\times}10^{-3}$	$\mathbf{0.09} \pm 8{\times}10^{-3}$	$3.81 \pm 9{\times}10^{-3}$	$\mathbf{0.01} \pm 3{\times}10^{-3}$	$\mathbf{0.10} \pm 5{\times}10^{-3}$	$\mathbf{0.11} \pm 6{\times}10^{-2}$
	LOSS	$1.9{\times}10^{-3} \pm 6{\times}10^{-7}$	$\mathbf{8.0{\times}10^{-7}} \pm 3{\times}10^{-8}$	$4.5{\times}10^{-3} \pm 4{\times}10^{-6}$	$\mathbf{1.1{\times}10^{-6}} \pm 3{\times}10^{-8}$	$\mathbf{8.7{\times}10^{-7}} \pm 3{\times}10^{-8}$	$\mathbf{1.2{\times}10^{-6}} \pm 5{\times}10^{-8}$
	EPOCHS	18 ± 2	39 ± 6	15 ± 1	44 ± 4	39 ± 4	47 ± 3

Figure 5.4: Learned filters w_i, corresponding parameter p_i/α_i and RMSE$_i$ between the learned filter and the target structuring element for both layers ($i \in \{1, 2\}$), MSE training loss at convergence and number of training epochs for $PConv$, $\mathcal{L}Morph$ and $\mathcal{S}Morph$ layers on a closing $\bullet$ task. Best (lowest) results are in bold. Abnormal results are in red.

actual erosion by w:

$$(f \ominus -(-w))(x) = (f \ominus w))(x)$$

Issuing from this, when talking about an erosion, the expected values of the structuring element should be between -1 and 0. Still, this minus sign is not the solution at everything as a range of values could be $[-0.3; 2.7]$ (taken from experiments with the $\mathcal{S}Morph$ layer), which even when taking the opposite is not even near $[0; 1]$; a thing I however noticed is that when changing the structuring elements from their ranges to $[0; 1]$, which can be resumed as applying $\tilde{w} = \frac{\hat{w} - min(\hat{w})}{max(\hat{w}) - min(\hat{w})}$, with $\hat{w}$ the learned filter and $\tilde{w}$ the result of this manipulation, the *RMSE* between $\tilde{w}$ and w was less than the one between $\hat{w}$ and w.

The first approach I tried is to find a way to mathematically explain this disparity, trying to prove that the formulas we used could produce same results for structuring elements in different ranges, more precisely that there was a way for $(f \oplus w)(x) = (f \oplus (\alpha w + \beta))(x)$ to be possible, with $\alpha, \beta \in \mathbb{R}$. Sadly, this led to nothing great besides the fact that I enjoyed toying with the formulas, and that it could prove to be useful for further experiments.

The other idea to solve this problem has been to look at the network architecture more globally and find a culprit for the compensation that is clearly happening. Even if nothing is proven or tested to the extend it should be to have a solid conclusion, some experiences I have done

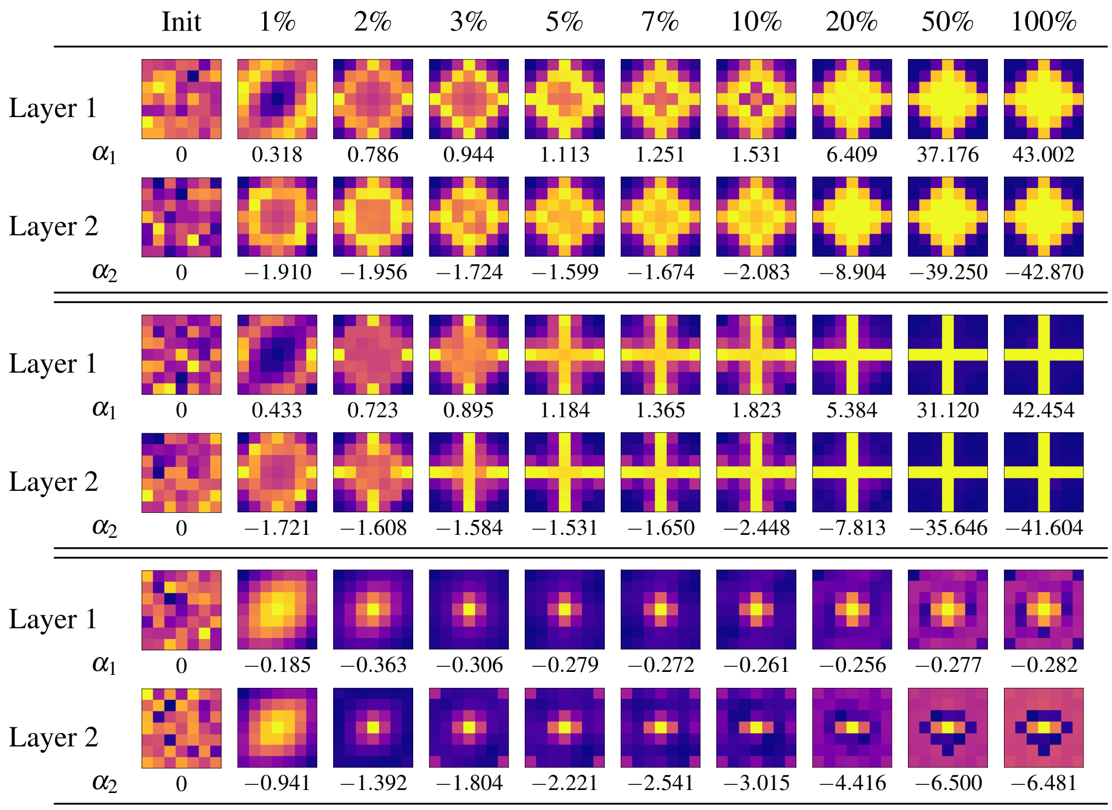


Figure 5.5: Convergence of two consecutive $\mathcal{S}Morph$ layers in a closing • scenario, with corresponding values of α_1 and α_2. Target structuring elements are *diamond3* (top row), *cross7* (middle row) and *disk2* (bottom row). Layers are shown at initialization, 1%, 2%, 3%, 5%, 7%, 10%, 20%, 50% and 100% of total number of training epochs.

clearly go in the sense of the scale bias layer ($1 \times 1 \times 1$ convolution layer, which can be resumed as a multiplication and an addition by/of learned parameters, see Figure 5.7) giving more flexibility when learning the structuring elements. It is worth mentioning that without it, $\mathcal{L}Morph$ and $PConv$ cannot reach convergence for an erosion as they cannot learn negative weights, and $\mathcal{S}Morph$ is *a priori* learning the filter in a more similar range as $[0; 1]$ than with it. This layer is thus necessary for CHM based layers (it also compensates for the rescale in range $[1; 2]$ of the input), but could be removed for runs with $\mathcal{S}Morph$.

5.4 Robustness Evaluation

To go further in the experiments and test if the proposed layers have the potential of bringing great results over more complicated but more realistic tasks, I led a bunch of new experiments which are just the beginning of a long list.

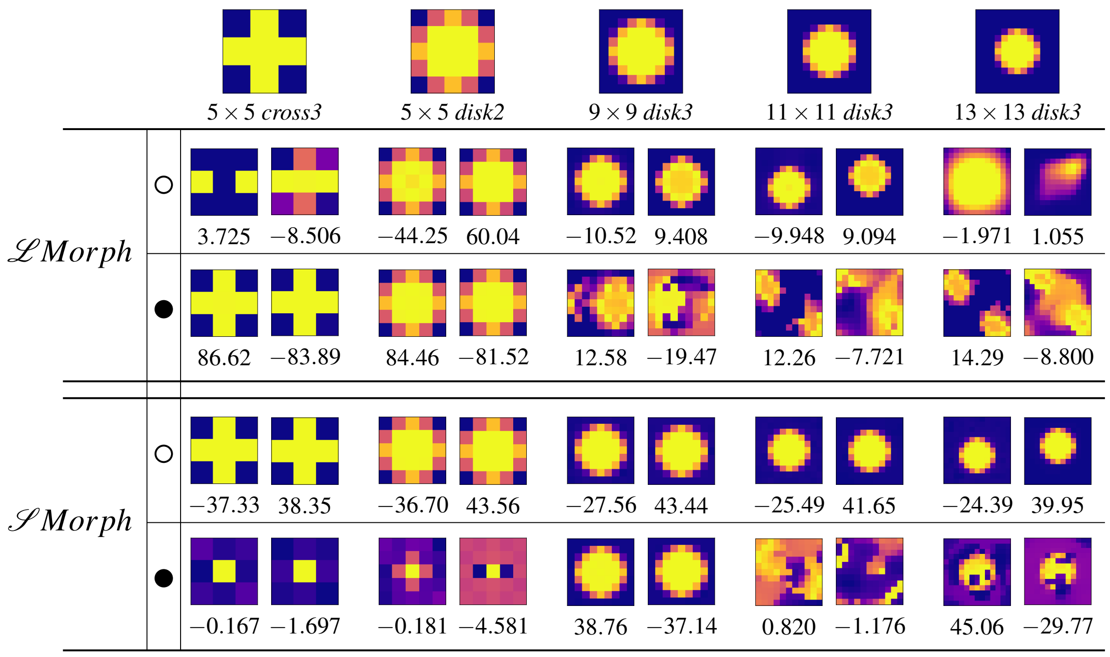


Figure 5.6: Learned filters for opening ○ and closing • operations for $\mathcal{L}Morph$ and $\mathcal{S}Morph$ for *cross3* and *disk2* in a 5×5 spatial support, as well as *disk3* in 9×9, 11×11 and 13×13 spatial supports.

5.4.1 Binary Morphology with $\mathcal{S}Morph$

As described in section 2.2, there exists an equivalent to binary mathematical morphology with grayscale formulas. To further test the capabilities of the proposed $\mathcal{S}Morph$ layer, experiments were conducted with the objective of learning binary structuring elements (Figure 5.8). Those studies could not be conducted on other layers as CHM based layers cannot by themselves (without scale/bias layer) learn things such as $-\infty$ (Equation 2.6).

The results of these experiments can bee seen in figures 5.9 and 5.10. What comes out of it is a total success for binary erosion and binary dilation, and mitigated but still very correct for opening and closing. Notice that, because the input image values are in practice bounded ($f(x) \in \{0, 1\}$ since the input images f are binary), the structuring function (2.6) is equivalent to

$$b : x \mapsto \begin{cases} 0 & \text{if } x \in B \\ \leq -1 & \text{otherwise} \end{cases} . \tag{5.1}$$

And, as $\mathcal{S}Morph(f, w, \alpha < 0)(x) \approx (f \ominus -w)(x)$ (from Equation 4.15), the expected filter to be learned are of the form

$$w(x) = \begin{cases} = 0 & \text{if } x \in B \\ \leq -1 & \text{otherwise} \end{cases} \qquad \text{for a dilation} \tag{5.2}$$

$$w(x) = \begin{cases} = 0 & \text{if } x \in B \\ \geq 1 & \text{otherwise} \end{cases} \qquad \text{for an erosion} \tag{5.3}$$

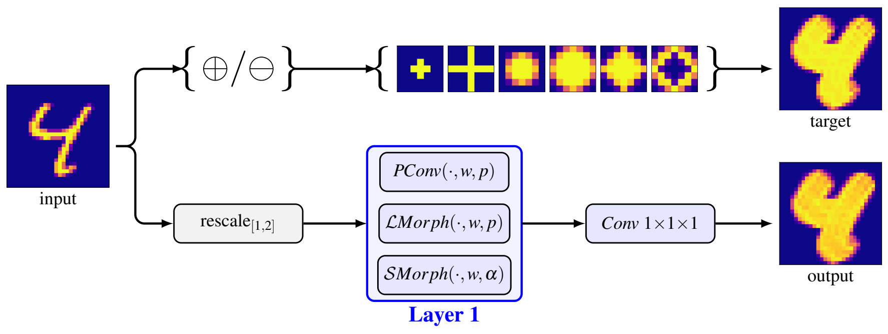


Figure 5.7: Network architecture used for the erosion/dilation scenarios. Blue blocks are trainable units. A scenario is defined as the choice of $\oplus$ or $\ominus$ and one of the 6 target structuring elements in the upper path, and the choice of one layer among $PConv$, $\mathcal{L}Morph$ and $\mathcal{S}Morph$ in the lower path.

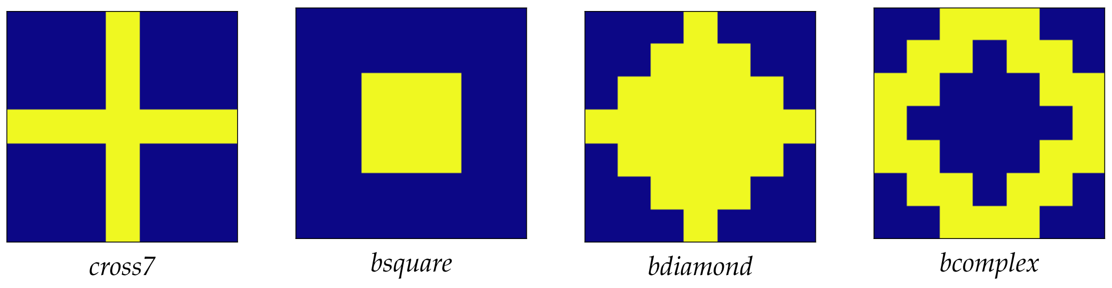


Figure 5.8: 7×7 target binary structuring elements. Yellow (resp. blue) corresponds to boolean TRUE (resp. FALSE).

In this sense, clipped versions of the results are calculated and displayed in the figures. They are used to calculate the $RMSE$ with the expected structuring element.

5.4.2 Denoising

As a second way of testing the layers and producing additional data, I tried the denoising power of $\mathcal{L}Morph$ and $\mathcal{S}Morph$. First, for a more realistic experience, I added the support of the *Fashion-MNIST* database[1]: it is a clothing pictures database which has the same structure as *MNIST* but acts as a bit more complex database. After assuring that classical operations as already evaluated with the *MNIST* database were also working on *Fashion-MNIST*, I implemented the noising and denoising process. I have in this way tested on *salt* noise (random pixels value turned to 1 as input pictures are ranging in $[0; 1]$) and *salt and pepper* noise (random pixel turned to 0 or 1), with two and four morphological layers respectively — *salt* noise needs an opening to be removed, so two operations (two layers) and *salt and pepper* needs an opening and a closing, so four layers. To have a more realistic *salt and pepper* noise, with pepper pixels not absorbed

[1]https://github.com/zalandoresearch/fashion-mnist

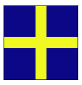
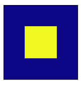
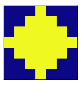
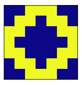
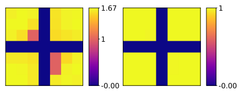
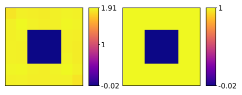
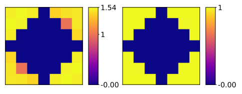
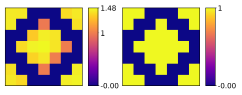
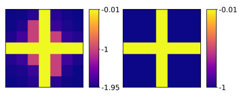
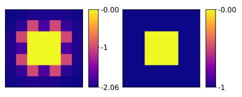
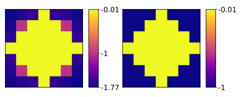
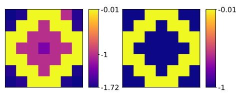
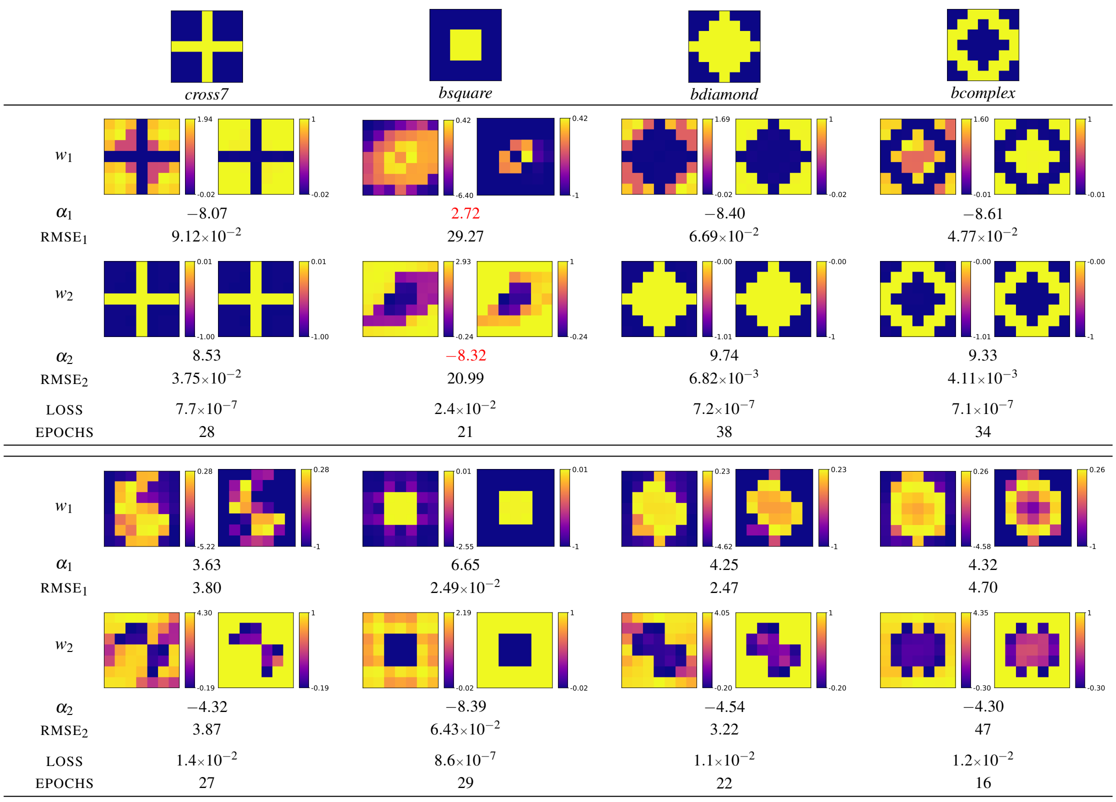
	 *cross7*	 *bsquare*	 *bdiamond*	 *bcomplex*
w				
α	-10.12	-9.80	-10.59	-10.43
RMSE	1.75×10^{-2}	4.65×10^{-2}	1.72×10^{-2}	1.08×10^{-2}
LOSS	2.9×10^{-7}	3.1×10^{-7}	2.9×10^{-7}	2.9×10^{-7}
EPOCHS	31	31	26	32
w				
α	9.10	8.88	9.96	10.11
RMSE	3.96×10^{-2}	1.78×10^{-2}	6.57×10^{-2}	5.45×10^{-2}
LOSS	4.5×10^{-7}	4.0×10^{-7}	4.6×10^{-7}	4.9×10^{-7}
EPOCHS	34	38	48	31

Figure 5.9: Learned filter w and its clipped version with associated colorbars, and quantitative metrics for a $\mathcal{S}Morph$ layer for binary erosion $\ominus$ (first row) and binary dilation $\oplus$ (second row) scenarios.

by the background, input images are a bit changed before being noised: if x is a standard input image, the actual input image for *salt and pepper* noise is equal to $\frac{1}{2}+\frac{x}{2}$, this results in the background being equal to 0.5 which is different from the 0 value of *pepper* noise.

The tests were conducted for multiple percentages of noise (10, 15, 20, 30, 40 and 50%). The salt-denoising power (Figures 5.11 and 5.11) of both layers is very impressive, especially for 40% and 50% corrupted pixels; Peak Signal-to-Noise Ratio (PSNR) still needs to be checked. The networks used for this part are, as said before, constituted of two morphological layers and we expect the network to learn an opening, which is an erosion followed by a dilation, and that is exactly what is happening as we can see in the figures 5.11 and 5.11 with the α and p parameters (for $\mathcal{S}Morph$ and $\mathcal{L}Morph$ respectively, shown above the learned structuring elements pictures) which are negative for the first layer and positive for the second.

For the *salt and paper* denoising part, networks with four layers were used for the training, and we expect the network to learn a combination of an opening and a closing operation. Results are a bit less satisfying with 30% noise (Figures 5.14 and 5.16), especially compared to the $PConv$ result (Figure 5.18). Nevertheless, for 10% *salt and pepper* noise (Figures 5.13, 5.15 and 5.17), the $\mathcal{S}Morph$ layer has a great visual result which still is promising for further tests on other tasks and more complex architectures. Surprisingly, for the 10% noise, $\mathcal{S}Morph$ (Figure 5.15) is succeeding on the denoising part but do not learn a succession of opening and closing operations but rather a way to fill the holes from pepper noise before applying a closing operation to remove the remaining salt noise.



	cross7	bsquare	bdiamond	bcomplex
α_1	-8.07	2.72	-8.40	-8.61
RMSE$_1$	9.12×10^{-2}	29.27	6.69×10^{-2}	4.77×10^{-2}
α_2	8.53	-8.32	9.74	9.33
RMSE$_2$	3.75×10^{-2}	20.99	6.82×10^{-3}	4.11×10^{-3}
LOSS	7.7×10^{-7}	2.4×10^{-2}	7.2×10^{-7}	7.1×10^{-7}
EPOCHS	28	21	38	34
α_1	3.63	6.65	4.25	4.32
RMSE$_1$	3.80	2.49×10^{-2}	2.47	4.70
α_2	-4.32	-8.39	-4.54	-4.30
RMSE$_2$	3.87	6.43×10^{-2}	3.22	47
LOSS	1.4×10^{-2}	8.6×10^{-7}	1.1×10^{-2}	1.2×10^{-2}
EPOCHS	27	29	22	16

Figure 5.10: Learned filter $w_{i\in\{1,2\}}$ and its clipped version, and quantitative metrics for two $\mathcal{S}Morph$ layers for binary opening $\circ$ (first row) and binary closing $\bullet$ (second row) scenarios. Abnormal results are in red.

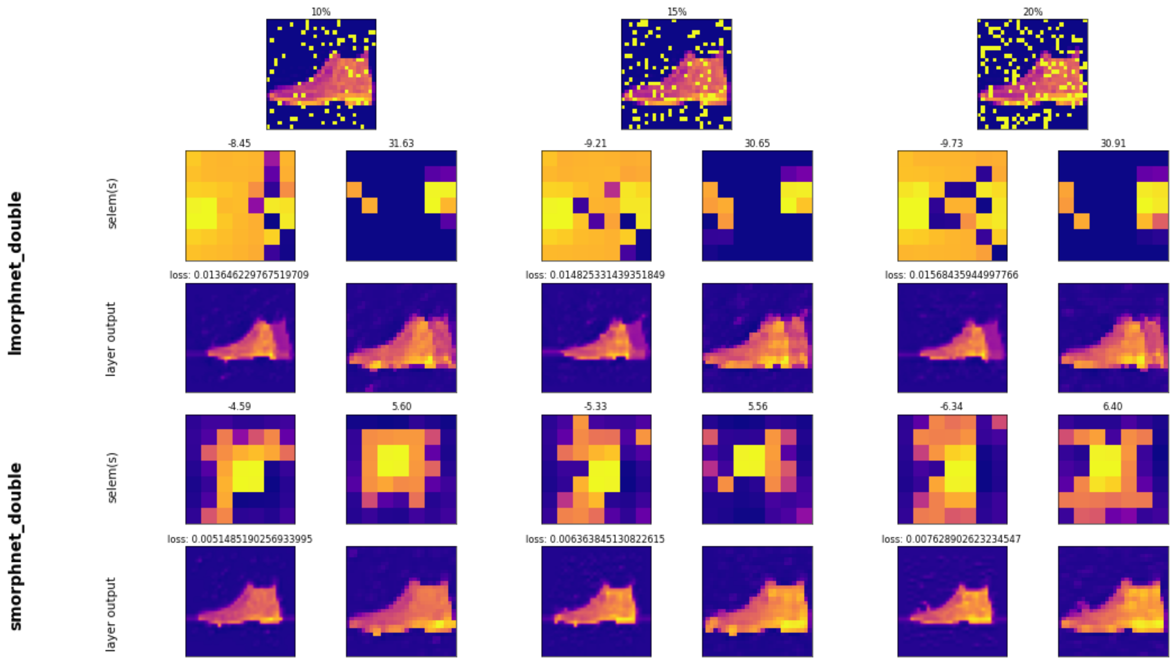


Figure 5.11: Results of learning denoising processes for different percentage (10, 15 and 20%) of *salt* noise

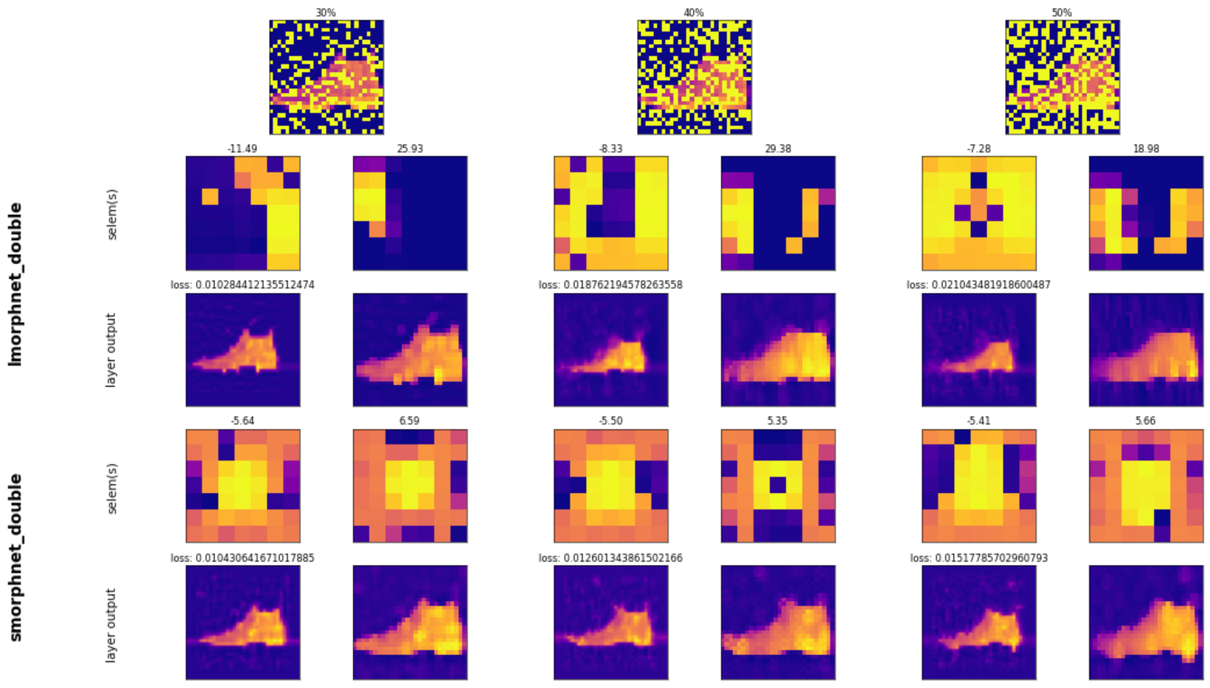


Figure 5.12: Results of learning denoising processes for different percentage (30, 40 and 50%) of *salt* noise

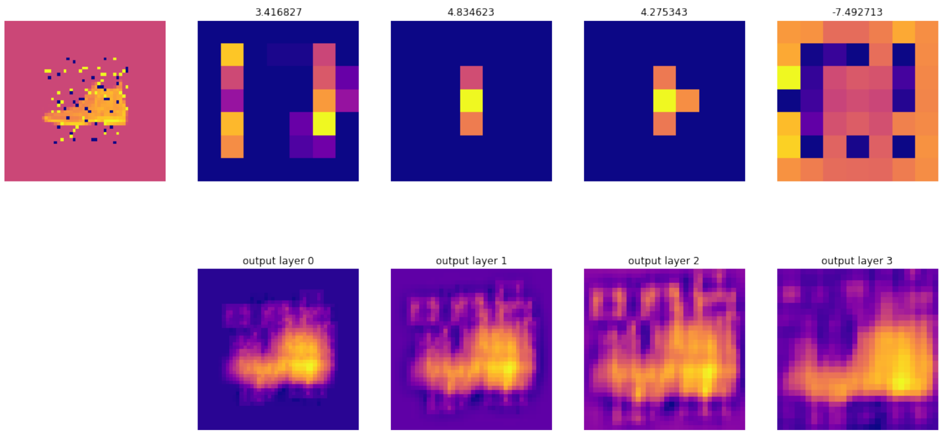


Figure 5.13: $\mathcal{L}Morph$ layer denoising learning for 10% *salt and pepper* noise

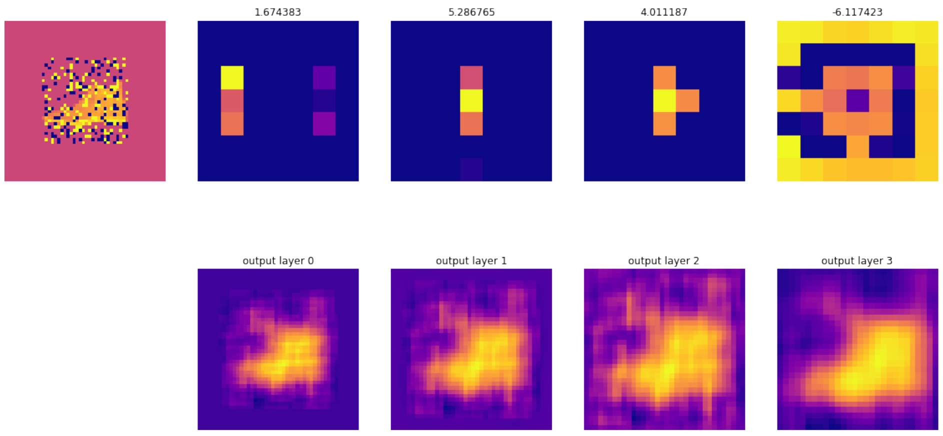


Figure 5.14: $\mathcal{L}Morph$ layer denoising learning for 30% *salt and pepper* noise

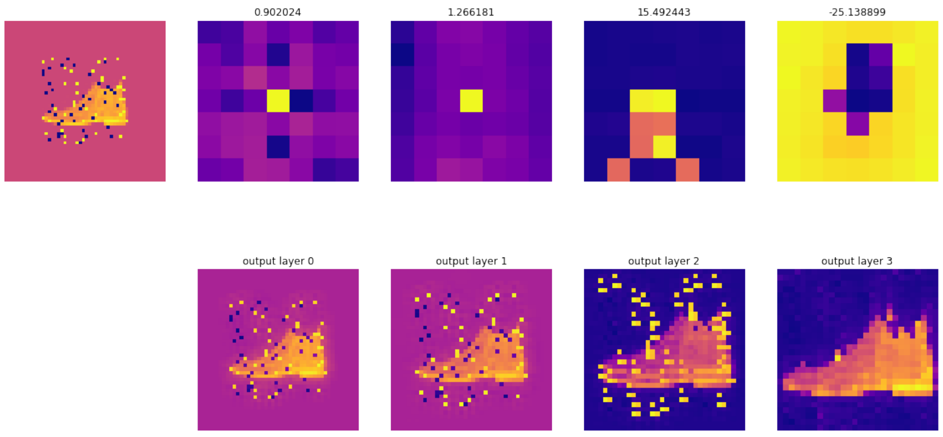


Figure 5.15: $\mathcal{S}Morph$ layer denoising learning for 10% *salt and pepper* noise

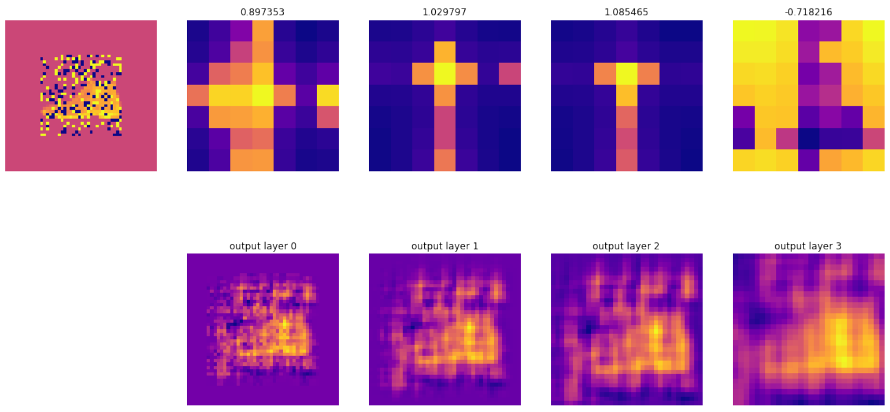


Figure 5.16: $\mathcal{S}Morph$ layer denoising learning for 30% *salt and pepper* noise

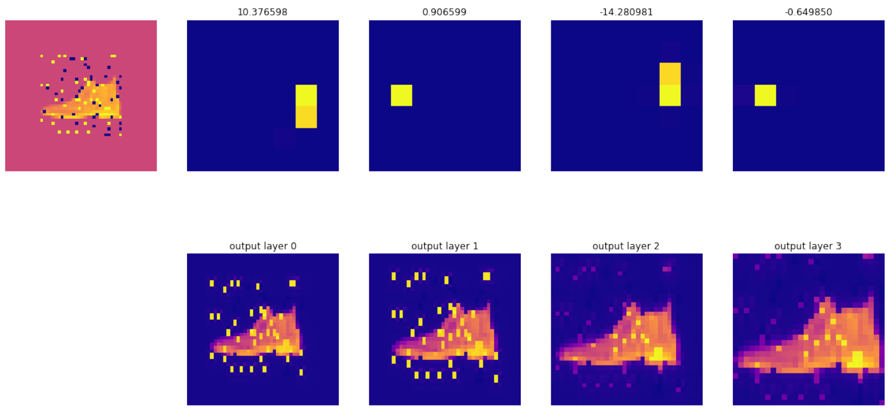


Figure 5.17: $PConv$ layer denoising learning for 10% *salt and pepper* noise

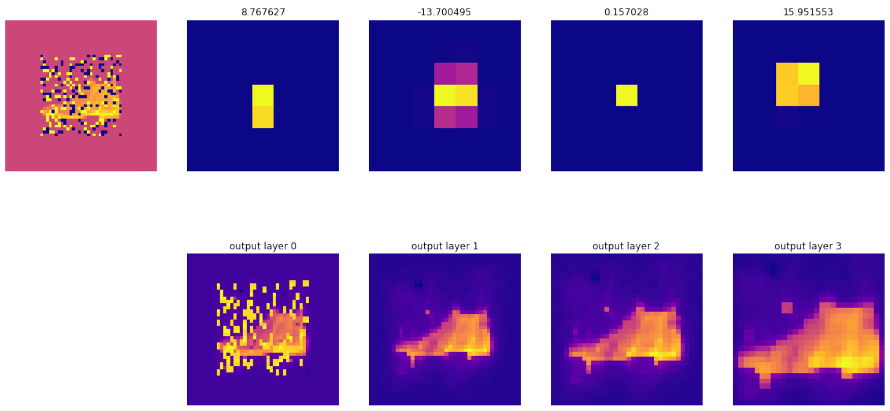


Figure 5.18: $PConv$ layer denoising learning for 30% *salt and pepper* noise

Chapter 6

Conclusion

In conclusion, during this semester I discovered new exciting fields of study, and learned a lot from my researches and experiments. I have taken back a subject which had already solid foundations and worked to pursue the effort done on trying to find an elegant way of learning structuring elements and morphological operators. I have started with a basic re-conduct of experiments, then dug out on the edge cases to fully understand what was or was not working, to finish with new experiments and new results. In six months with the huge help of my supervisors we have been able to submit a paper to the **Journal of Mathematical Imaging and Vision** (**JMIV**), which hopefully will be accepted.

Future Work

The next semester I will be more focused on trying new things: new morphological operations like the top-hat, modifying the formulas to see if I can counter a possible undesired side effect of needing to learn $-w$ for an erosion operation, adding layers to create a classifier and see if a morphological neural network can perform as well or better than a classical CNN, and as a last experiment, see if a morphological network can be of any help for hyper-intensities detection in brain images.

Acknowledgments

I would like to thank my supervisors Guillaume Tochon and Élodie Puybareau, for believing in me at first by giving me the opportunity to work on this subject, and then letting me work autonomously with the liberty to linger over the topics I wanted to go more in depth, but also for their support, their useful ideas and the amount of work they have put into my subject and the paper, to submit it (almost) in time. I would also like to thank all the teachers and assistants of my formation at EPITA and more specifically the RDI teachers who indirectly supported me on my subject by helping me becoming a more experience *baby*-researcher.

Chapter 7

Bibliography

Angulo, J. (2010). Pseudo-morphological image diffusion using the counter-harmonic paradigm. In *ACIVS*. (page 13)

Angulo, J., Velasco-Forero, S., and Meyer, F. (2017). Mathematical morphology and its applications to signal and image processing - 13th international symposium, ismm 2017, fontainebleau, france, may 15-17, 2017, proceedings. In *ISMM*. (page 4)

Hadwiger, H. (1957). Vorlesungen über inhalt, oberfläche und isoperimetrie. (page 4)

Hilbert, D., Speiser, A., and Weyl, H. (1911). *Gesammelte Abhandlungen von Hermann Minkowski*. Number v. 1. B. G. Teubner. (page 4)

Hu, Y., Belkhir, N., Angulo, J., Yao, A., and Franchi, G. (2021). Learning deep morphological networks with neural architecture search. *ArXiv*, abs/2106.07714. (page 12)

Kirszenberg, A., Tochon, G., Puybareau, É., and Angulo, J. (2021). Going beyond p-convolutions to learn grayscale morphological operators. In *DGMM*. (pages 4, 13, 14, 18, and 19)

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60:84 – 90. (page 4)

LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., and Jackel, L. (1989). Handwritten digit recognition with a back-propagation network. In *NIPS*. (pages 10 and 14)

Masci, J., Angulo, J., and Schmidhuber, J. (2013). A learning framework for morphological operators using counter-harmonic mean. In *ISMM*. (page 12)

Matheron, G. (1975). Random sets and integral geometry. (pages 4 and 5)

Minkowski, H. (1903). Volumen und oberfläche. *Mathematische Annalen*, 57:447–495. (page 4)

Ponti, M., Ribeiro, L. S. F., Nazaré, T. S., Bui, T., and Collomosse, J. (2017). Everything you wanted to know about deep learning for computer vision but were afraid to ask. *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*, pages 17–41. (page 9)

Ritter, G. and Wilson, J. N. (1996). Handbook of computer vision algorithms in image algebra. In *Handbook of computer vision algorithms in image algebra*. (page 5)

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. S., Berg, A., and Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252. (page 9)

Serra, J. (1983). Image analysis and mathematical morphology. (pages 4, 5, and 6)

Shen, Y., Zhong, X., and Shih, F. (2019). Deep morphological neural networks. *arXiv preprint arXiv:1909.01532*. (page 12)

Shih, F., Shen, Y., and Zhong, X. (2019). Development of deep learning framework for mathematical morphology. *Int. J. Pattern Recognit. Artif. Intell.*, 33:1954024:1–1954024:16. (pages 9 and 12)

Sternberg, S. (1986). Grayscale morphology. (page 6)

Yu, H. G. (2004). Morphological image segmentation for co-aligned multiple images using watersheds transformation. (page 7)