

# LGN-CNN: A BIOLOGICALLY INSPIRED CNN ARCHITECTURE.

FEDERICO BERTONI<sup>\*,†</sup>, GIOVANNA CITTI<sup>†</sup>, AND ALESSANDRO SARTI<sup>\*</sup>

## ABSTRACT

In this paper we introduce a biologically inspired CNN architecture that has a first convolutional layer that mimics the role of the LGN. The first layer of the net shows a rotationally symmetric pattern justified by the structure of the net itself that turns up to be an approximation of a Laplacian of Gaussian. The latter function is in turn a good approximation of the receptive profiles of the cells in the LGN. The analogy with respect to the visual system structure is established, emerging directly from the architecture of the net.

## 1. INTRODUCTION

Convolutional Neural Networks (CNNs) have been inspired by the modular structure of the visual cortices leading to an architecture that contains many layers that try to mimic some behavior of the visual system. In particular, each convolutional layer obtains its input from the previous convolutional layer. The connections and similarities between CNNs and visual system have been widely studied in the last years; however, CNNs have been developed independently reaching state-of-the-art performances in many areas.

In [9] the authors introduce new smoothness functionals based on the regularization theory on CNNs. The authors in [21] develop a CNN architecture based on the structure of the visual cortices, justifying each layer of the net by well known behavior of the visual system, as for example introducing a first layer composed by Gabor filters with different orientations and scales. In [26] it has been studied the strict connections between each cortical layer and convolutional layer based on the encoding and decoding ability of the visual system, using goal-driven hierarchical convolutional neural networks. Furthermore, in [27], the authors model the ventral

---

<sup>\*</sup>CAMS, CNRS - EHESS, PARIS, FRANCE.

<sup>†</sup>DIPARTIMENTO DI MATEMATICA, UNIVERSITÀ DI BOLOGNA, ITALY.

## 2 LGN-CNN: A BIOLOGICALLY INSPIRED CNN ARCHITECTURE.

---

stream (the series of cortical areas thought to subserve object recognition) comparing it with fMRI voxel responses. Moreover, in [17] the authors obtain Gabor shape filters by training an unsupervised learning algorithm on natural images.

Recurrent Neural networks (see e.g. [22]) have been introduced to implement the horizontal connectivity typical of the layers of the natural visual cortex. A modification of these nets, more geometric and more similar to the structure of the brain, have been recently proposed in [16].

Despite these similarities, the structure of an artificial neural network and the human brain is still very different. In particular in most cases the net is able to reconstruct a bank of Gabor shape filters in the first convolutional layer which are approximations of the Receptive Profiles (RPs) of the cells in V1. In [21] where the authors describe the connections between each CNN layer and the visual cortices, the role of the Lateral Geniculate Nucleus (LGN) has not been investigated. In our paper we aim to introduce a CNN architecture that tries to mimic this behavior. In particular, since the RPs of LGN cells are rotationally symmetric, we focus on the relation between the architecture of the CNNs and the invariance properties of their filters, in order to find an architecture that gives rotationally symmetric filters in the first convolutional layer.

As regards CNNs, their invariance properties are one of the main theoretical problem about them. Convolutional layers in classical CNNs are translation-invariant since all filters are applied all over the input. This allows the net in the case of image classification to recognize the image despite the position of the object.

A lot of invariances and symmetries are present at the level of the filters obtained through the backpropagation algorithm. One of the most powerful and most studied is the **rotation invariance**. Indeed, it often happens that at a certain layer the same filter appears several times rotated by different angles. This behavior has suggested in some cases to modify the structure of a CNN imposing the rotation invariance directly to the filters. In [15] and [25] the authors rotate the filters in each layer by  $R$  different angles and apply them to the input. This allows to reduce the number of filters at each layer and, as a consequence, the number of parameters and the complexity of the net decrease giving results comparable to state-of-the-art CNNs.

Another method for introducing invariance under rotations is **data augmentation**, which consists in incrementing the number of training images by flipping, rotating, rescaling, cropping, adding gaussian noise to them. In this way the net is trained on a larger set of images and can learn different kinds of invariances. In [6] and [5] the authors rotate the same image by  $S$  different angles in order to achieve rotation invariance.

In [4], [12] and [7] the authors introduce one or more novel layers to face the rotation invariance problem. In [4] they propose four different layers that work on the input rotated by 0, 90, 180 and 270 degrees. In [12] a different kind of pooling is used before the fully connected layers in order to obtain invariance with respect to different features. A kernel that finds out symmetries in the input connecting consecutive layers is introduced in [7].

In the present work, we propose to build a CNN architecture inspired by the structure of visual cortices called LGN-CNN. In particular, the idea is to introduce a first layer that behaves similarly to the LGN that prefilters the visual stimulus obtained by the retina, highlighting the contours of the objects present in the stimulus. The RPs of cells in the LGN can be approximated by a Laplacian of Gaussian (LoG) which is rotationally symmetric (for a review see for example [18]). Thus, our first layer contains a single filter: this enforces the development of rotation symmetry during the training phase, so that the filter eventually approximates a LoG.

As regards the filters of our CNN obtained after training, we expect that the second layer of the net mimics the behavior of V1 which is the first visual cortex that analyzes the visual stimulus after the LGN. Simple cells in V1 have been studied for a long time and their RPs can be approximated by a Gabor function (see e.g. [2], [11], [13], [18]). Thus, we have studied their shapes by approximating the filters obtained after the training phase with a Gabor function. We have compared the results obtained on a LGN-CNN and on a classical CNN with the RPs of a macaque's V1 from the work of Ringach ([19]). We have observed that the filters of the second layer obtained with our CNN are more similar to the neuroscience data of macaque's RPs with respect to classical CNN. This enforces the link between our architecture and the visual system structure, at least as regards the LGN cells and simple cells in V1.

## 2. THE VISUAL SYSTEM

The visual system is one of the most studied and most understood area of the brain. We will describe only the parts that interested the most our studies. For a complete overview see for example [18], [23], [10].

The retina is a light-sensitive layer of tissue of the eye which receives the visual stimulus and translates it into electrical impulses. These impulses reach firstly the LGN which is a part of the thalamus whose cells preprocess the visual stimulus.

#### 4 LGN-CNN: A BIOLOGICALLY INSPIRED CNN ARCHITECTURE.

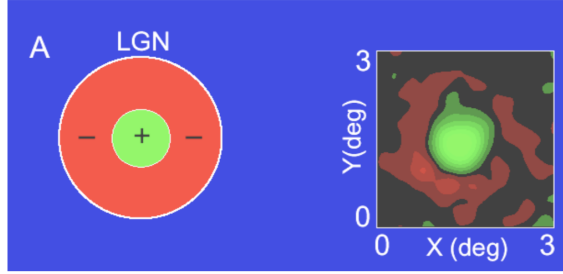


FIGURE 1. On the right: RP of a LGN cell where the excitatory area is in green and the inhibitory one is in red. On the left: Its approximation by a LoG. *From: [3].*

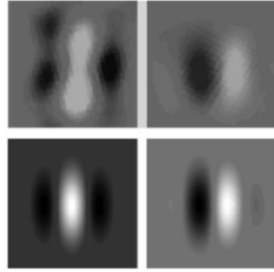


FIGURE 2. First row: RPs of two simple cells of V1 where the excitatory area is in white and the inhibitory one is in black. Second row: their approximations by Gabor functions. *From: [20].*

Then the impulse is processed by the cells of V1, whose output is taken in input to all the other layers of the visual cortex.

We are mainly interested in the cells of the LGN and in the simple cells of V1. Each cell receives the electrical impulse from a little portion of the retina  $\Omega$  called **receptive field** (RF). The RF of each cell is divided in excitatory and inhibitory areas which are activated by the light and that can be modeled as a function  $\Psi : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}$  called **receptive profile** (RP). Thus, if the excitatory areas are activated the cell will fire whereas it will be silenced in the case of inhibitory areas activation. Figure 1 shows the RP of a LGN cell that can be modeled by a **laplacian of gaussian** (LoG). Indeed it will highlight the contours of the objects in the image whereas it will set to zero all the uniform areas. On the other hand figure 2 shows the RPs of two simple cells of V1 modeled by Gabor functions. These cells will fire if the contour of the object passing in their RF has a similar orientation w.r.t. their RPs.

### 3. INTRODUCING LGN-CNN ARCHITECTURE

In this section we have introduced the main novelty of this paper, a CNN architecture inspired by the structure of the visual system. In particular, we have investigated the role of the LGN in the visual system whose receptive profile cells can be approximated by a LoG. Since LoG is a **rotationally symmetric function**, we aim to find a CNN architecture that achieves this property in the first layer and eventually attains a LoG shape.

#### 3.1. Relations between a CNN and the visual system.

Many authors have studied the connections between a CNN and the RPs of simple cells in V1 (see e.g. [26], [27], [17], [9], [21]). In particular, the first convolutional layer of a CNN is usually composed by a bank of Gabor filters obtained after the training phase. This suggests that the first convolutional layer analyzes the contours of the objects present in the image similarly to what happens in V1. Also the architecture of a CNN has many points in common with the structure of the visual system. Just to recall some of them:

- It is composed by many convolutional layers that sequentially analyzed the image, similarly to the visual cortices of the visual system;
- After each convolutional layer it applies a non linearity that sets to zero all the negative values, similarly to what happens to the neurons that fire only if they reach a certain amount of voltage;
- The spatial dimensions decrease after each convolutional layer giving the opportunity to the filters of the next convolutional layer to analyze the information from a larger area of the starting image, similarly to what happens to neurons of following visual cortices that receive information from a group of neurons of the previous visual cortex.

There are several differences between CNNs and the visual system, as for example the fact that the RPs of cells are contextual and vary according to the visual stimulus. Also the structure of the visual cortices is more complex, since the neurons in the same cortex communicates each other and following layers could communicate with previous ones giving feedback information. However the studies of the similarities between CNNs and the visual system could improve the comprehension of both of them. In the next subsections we are going to describe which problem we have faced and how.

### 3.2. Introducing LGN in a CNN.

As far as we know, the action of the LGN has been ignored and it has not been implemented in a CNN. As we have already discussed in subsection 2 the RP of a LGN cell can be modeled by a LoG that acts directly on the visual stimulus and highlights the contours. Our aim is to build a CNN architecture that mimics this behavior in order to strengthen the links between CNNs and the visual system.

Since the LGN preprocesses the visual stimulus before it reaches V1, we should add a first layer at the beginning of the CNN that reproduces the role of the LGN. It should apply to the image a LoG highlighting the contours of the objects, indeed without modifying in any way the dimensions of the input. Thus, the idea is to introduce a first convolutional layer containing a single filter that eventually should obtain a LoG shape.

The theoretical idea behind this structure can be found in a simple result on rotationally symmetric functionals. In particular, if we have a rotationally symmetric functional  $F$  that has a unique minimum  $\omega$  then  $\omega$  is also rotationally symmetric. Indeed, since  $F$  is rotationally symmetric,  $F(\omega \circ g) = F(\omega)$  for a rotation  $g$ . Thus, since the minimum is unique,  $\omega = \omega \circ g$  and this implies the rotation symmetry of the solution. There are several results on symmetries of minimum for functionals as for example in [14], [8]. Our aim is to extend these results in the case of CNNs in particular on our architecture that we will call as Lateral Geniculate Nucleus Convolutional Neural Network (LGN-CNN).

We have also studied the modifications that occurs to the filters in the second convolutional layer in our architecture. In particular, we have tried to compare their shape with respect to the RPs of a macaque's V1 from the work of Ringach ([19]).

In his work, Ringach has recorded the RPs of a macaque's V1 and has fitted them with a Gabor function defined as follows:

$$(1) \quad h(x', y') = A \exp(-(x'/\sqrt{2}\sigma_x)^2 - (y'/\sqrt{2}\sigma_y)^2) \cos(2\pi f x' + \phi)$$

where  $(x', y')$  is translated and rotated from the original coordinate system  $(x_0, y_0)$

$$x' = (x - x_0) \cos \theta + (y - y_0) \sin \theta \quad y' = -(x - x_0) \sin \theta + (y - y_0) \cos \theta.$$

Then he has compared his results with respect to the filters obtained using the Independent Component Analysis and the Sparse Coding. These are the main steps he followed:

- Recording the RPs from several simple cells in V1;
- Fitting a Gabor function defined in equation (1) to the RPs;
- Comparing the results on  $(n_x, n_y) = (\sigma_x \cdot f, \sigma_y \cdot f)$  plane.

Thus, we have followed the same steps by approximating the filters obtained after the training phase with a Gabor function (1) and plotting them on the  $(n_x, n_y)$  plane. Indeed, we can compare the elongation in the  $x$  and  $y$  direction which characterizes the RPs of simple cells in V1. This analysis should enforce the link between our architecture and the visual system structure, at least as regards simple cells in V1.

### 3.3. The layers that compose LGN-CNN.

Let us formally describe the architecture of LGN-CNN. This net takes as input an image  $I$  modeled as a function  $I : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}$ , where  $\Omega$  is a square of size  $M \times M$ , and tries to classify it between  $d$  different categories. During the entire description we assume that the input of each layer is a tensor  $T^i$  of size  $M^i \times M^i \times N^i$  where  $M^i$  is the spatial dimension and  $N^i$  matches the number of filters of the last convolutional layer.

The  $i$ -th convolutional layer is a functional  $\ell^i$  that acts on its input by applying  $n^i$  linear operators defined as follows

$$\Psi_j^i : \Omega^i \subset \mathbb{R}^3 \rightarrow \mathbb{R} \quad , \quad j = 1 \dots n^i.$$

where  $\Omega^i$  is a cube of size  $s^i \times s^i \times n^{i-1}$ . Indeed each filter is a tensor of size  $s^i \times s^i \times n^{i-1}$ , obtained after training. Thus the convolutional layer acts in the following way on its input  $T^i$ :

$$\ell^i(T^i(x))_j = \sum_{y \in b^x} \Psi_j^i(y) \cdot T^i(y) \quad , \quad j = 1 \dots n^i.$$

where  $b^x$  is the cubic neighborhood of  $x$  of the same size of the filter. Let us note that the first convolutional layer  $\ell^0$  is composed of only one filter  $\Psi^0$  of size  $s^0 \times s^0$ . After each convolutional layer we will apply a RELU  $R$  which is a function that applies elementwise to the input  $T^i$  and let all negative numbers be zero:

$$(R(T^i)) = \max(T^i, 0)$$

After each convolutional layer and the RELU, with the exception of the first layer  $\ell^0$ , we apply a pooling that splits the output of the RELU in squares of size  $s_p \times s_p$  (usually  $s_p = 2$  or  $s_p = 4$ ) in the spatial dimensions and shrinks each of them to a single value, usually the maximum or the average of elements in the square.

## 8 LGN-CNN: A BIOLOGICALLY INSPIRED CNN ARCHITECTURE.

Formally, if denote  $b^i$  the  $i$ -th the set  $s_p \times s_p$  square of the decomposition of the input  $T^i$ , then the maximum pooling  $p_m^c$  and the average pooling  $p_a^c$  are defined as

$$p_m^c(T^i)^k = \max_{c \in b^k} x_c^k, \quad p_a^c(T^i)^k = \sum_{c \in b^k} x_c^k$$

Note that after the first layer  $\ell^0$  we will not apply any pooling. In this way taking a classical CNN and adding  $\ell^0$  will not modify the structure of the net; this is a great advantage since we can compare easily the two architectures. Indeed we will not add complexity to the problem since the number of parameters will only increase of  $s^0 \times s^0$ . Furthermore,  $\Psi^0$  will prefilter the input image without modifying in any way its dimension; this behavior mimics the behavior of the LGN which let the net to be closer to the visual system structure.

Thus, at the end of the net, after all convolutional layers, RELUs and poolings, there is a fully-connected ( $FC$ ) layer which modifies the size of the input eventually matching the number of categories  $d$ . In particular, the  $FC^i$  layer is a functional that acts in the same way as the convolutional layer, indeed it is composed by  $n^i$  linear operators. The main difference is that each filter approximated by a tensor has the same dimension of the input. Thus the output of the  $FC^i$  layer is a vector of length  $n^i$ ; in this way, if we set  $n^i = d$  we obtain a vector that is long as the number of categories  $d$ .

Let us note that however there are no constraints about the values that the  $FC$  layer's output  $I^*$  can obtain. Since we would like to classify an image, the output of the net should represent a probability distribution. Thus we will apply a softmax  $\sigma$  that modifies the values in the following way:

$$(2) \quad \sigma(I^*)_z = \frac{e^{I_z^*}}{\sum_k e^{I_k^*}}.$$

Then we obtain a probability distribution over the set of categories since  $0 \leq \sigma(I^*)_z \leq 1$ ,  $\forall z = 1 \dots d$  and  $\sum_z \sigma(I^*)_z = 1$ . Thus we have built a function that takes an input image  $I$  and returns a probability distribution over a set of  $d$  categories

$$F : I \in L^1(\mathbb{R}^2) \mapsto \mathbb{R}^d,$$

The net selects the category that most likely is associated to  $I$ , indeed

$$\tilde{z} = \arg \max_z F(I)_z.$$



A good classifier should have good performance on a test set. But how can we determine all the parameters in the net?

### 3.4. The loss function.

So far we have described the layers that compose the structure of a net but we have not said anything about how we can obtain all the parameters. First of all let us outline that all the parameters of the net come from the convolutional layers and the  $FC$  layer since the pooling and RELU layers do not have any parameter to fit. Then if  $y : \Gamma \subseteq L^1(\mathbb{R}^2) \rightarrow \{1, \dots, d\}$  is the label functional defined on a set of images  $\Gamma \subseteq L^1(\mathbb{R}^2)$  that well classifies all the image in the training set  $\Gamma$ , we could define a loss function in order to find a functional  $F$  that well classifies the images in  $\Gamma$  and eventually also other images (say for example the images in a test set). The loss function we have used is the following:

$$(3) \quad L(F(I), y(I)) = F_{\bar{z}}(I) + \log\left(\sum_z e^{(F_z(I) - F_{\bar{z}}(I))}\right) - F_{y(I)}$$

If the loss function is small over a training set it means that the net is able to perform the classification task. Then our objective is to minimize the loss function on a training set and, applying a backpropagation algorithm, determine the parameters of the net. This procedure is repeated several times in order to obtain a better approximation of the parameters and having better performances on a test set.

## 4. APPLICATIONS OF LGN-CNN

### 4.1. Settings.

In this subsection we are going to describe the settings in which we have tested our architecture. We have used MATLAB2019a for academic use and the software MatConvNet (see [24]).

Since we would like to obtain a rotationally symmetric filter we have decided to train our LGN-CNN on a dataset of natural images called STL-10 (see [1]) that contains 5000 training images divided in 10 different classes. Indeed using a dataset that contains for the most part few directions (say for example most vertical lines) could have been leading to a directional filter in the first layer of our architecture. We have modified the images in the following way: we have converted them from RGB color to grayscale color using the built-in function *rgb2gray* of MATLAB; we have rotated them randomly by 0, 90, 180 and 270 degrees. This was due to the

## 10 LGN-CNN: A BIOLOGICALLY INSPIRED CNN ARCHITECTURE.

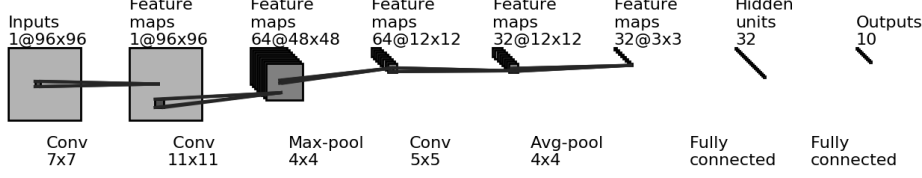


FIGURE 3. Architecture of LGN-CNN.

fact that we would like to prevent any possible shift of the center of the filter in the first layer. Indeed, most of images contains the sky which is on the top of the image and could possibly shift the center of the filter towards the bottom. Furthermore we have chosen this dataset since the images' size is  $96 \times 96$ ; in fact we decided to use quite large filters in the first and second layer ( $7 \times 7$  and  $11 \times 11$  respectively) in order to obtain more information about their shapes.

Figure 3 shows the architecture of our CNN. The input is an image of size  $96 \times 96 \times 1$ . Then there is the first convolutional layer  $\ell^0$  composed by only  $\Psi^0$  of size  $7 \times 7$  followed by a RELU. Thus the second layer  $\ell^1$  composed by 64 filters of size  $11 \times 11$  receives as input a matrix of the same size of the image. Note that the stride is 2 and this is why the spatial dimensions half. After the net applies a RELU and a max POOLING with squares of size  $4 \times 4$ . The third and last convolutional layer  $\ell^2$  is composed by 32 filters of size  $5 \times 5 \times 64$  and it is followed by a RELU and an avg POOLING with squares of size  $4 \times 4$ . Eventually two fully-connected layers are applied giving as output a vector of length 10. Here the net applies a softmax  $\sigma$  defined in equation (2) in order to obtain a probability distribution over the 10 classes. The functional that models this net is the following

$$(4) \quad F(I) := (\sigma \circ FC^2 \circ FC^1 \circ p_a^4 \circ R \circ \ell^2 \circ p_m^4 \circ R \circ \ell^1 \circ R \circ \ell^0)(I)$$

A loss function as the one defined in equation (3) is applied to the functional (4).

### 4.2. The first layer of the net.

After the training phase we can analyze the net we have obtained. In this subsection we focus on the first layer. Figure 4 shows the filter  $\Psi^0$  and the comparison with respect to minus the LoG. Let us note that figure 4a is the actual result of the net after training which has positive value in the center and negative ones around it.

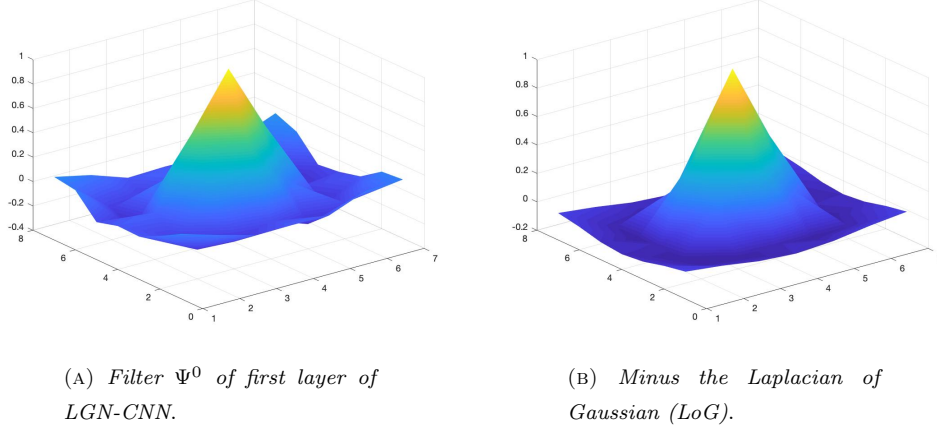


FIGURE 4. Comparison between the filter  $\Psi^0$  and minus the LoG.  
We can see that  $\Psi^0$  is really close to a discrete approximation of a LoG.

We can see that figure 4b which shows minus the Laplacian of Gaussian has a shape really close to  $\Psi^0$ . Moreover figure 5a shows the actual filter  $\Psi^0$  obtained after the training phase. Then in figures 5b and 5c we compare a  $2D$  approximation of  $\Psi^0$  and minus the LoG in which the rotationally symmetric pattern is clearer. Let us note that the filter attains this shape without any constraint except the structure of the net itself. This suggests that choosing the net architecture influences directly the filters we will obtain, letting us to link the net architecture to the cortical layers structure.

#### 4.3. The second layer of the net.

Since we would like to enforce the link between our architecture and the structure of the visual system we have decided to study the filters in the second layer comparing them with some real data obtained on monkey in [19]. Therefore, we have trained two different CNNs, a LGN-CNN defined by the functional (4) and a classical CNN defined by the functional (5) in which we have eliminated the first convolutional layer  $\ell^0$  and its following RELU  $R$ , characteristic of our architecture.

$$(5) \quad F(I) := (\sigma \circ FC^2 \circ FC^1 \circ p_a^4 \circ R \circ \ell^2 \circ p_m^4 \circ R \circ \ell^1) (I)$$

Let us note that in both architectures  $\ell^1$  contains filters with Gabor shapes after training. This is a well known results on the filters of the first convolutional layer

## 12 LGN-CNN: A BIOLOGICALLY INSPIRED CNN ARCHITECTURE.

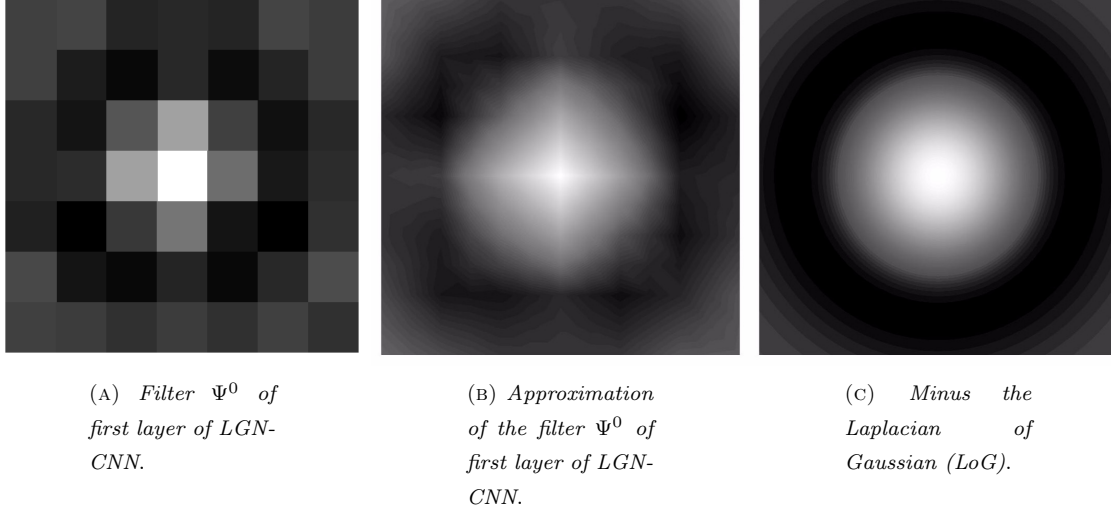


FIGURE 5. The first figure shows the  $7 \times 7$  filter  $\Psi^0$  of the net. Then it shows the approximation of  $\Psi^0$  and minus the LoG. Comparing the last two figures it is possible to see that  $\Psi^0$  is close to a LoG.

of CNNs (e.g. in [21], [26]); however, the introduction of a first layer composed by a single filter does not change this behavior, enforcing the link of our architecture and the visual system structure. Indeed, we have studied the statistical distribution of these banks of filters confronting the results with the real data of Ringach.

We have approximated the filters in the banks using the function (1); figure 6 shows some of the filters of LGN-CNN and their approximation. We can define  $(n_x, n_y) = (\sigma_x \cdot f, \sigma_y \cdot f)$  where  $n_x$  and  $n_y$  estimate the elongation in  $x$  and  $y$  directions respectively thanks to  $\sigma_x$  and  $\sigma_y$ . They are rescaled by  $f$  which indicates how far the shape of the filter is with respect to a Gaussian; in particular, if  $f = 0$  the function  $h$  in (1) simplifies to a Gaussian since the cosine becomes a constant. In order to better compare the plots we looked for the distribution that best fits the neural data. In particular, it approximates the points closer to the origin with a line  $y = \alpha x$  and then it approximates the rest of the points with a line starting from the previous one.

Figure 7 shows the three plot we would like to compare. Let us note that introducing LGN-CNN modifies the elongation of Gabor filters in  $\ell^1$ . In particular, in classical CNN the filters are often more elongated in the  $x$  direction as we can see from the slope of the interpolating line in figure 7a. In figure 7b we can see that the slope changes greatly and that the filters become much more elongated in

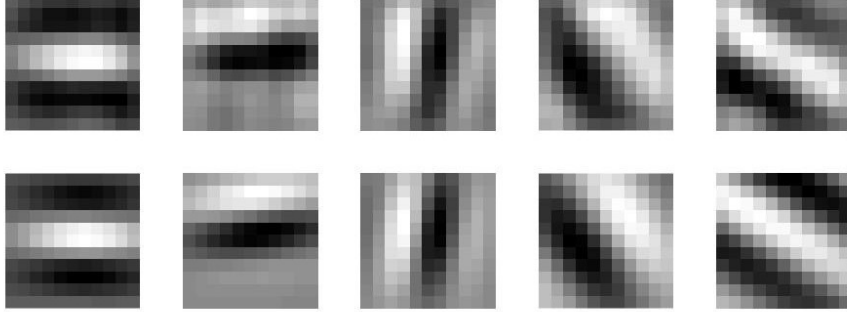


FIGURE 6. First row: filters from LGN-CNN. Second row: their approximation with the function (1).

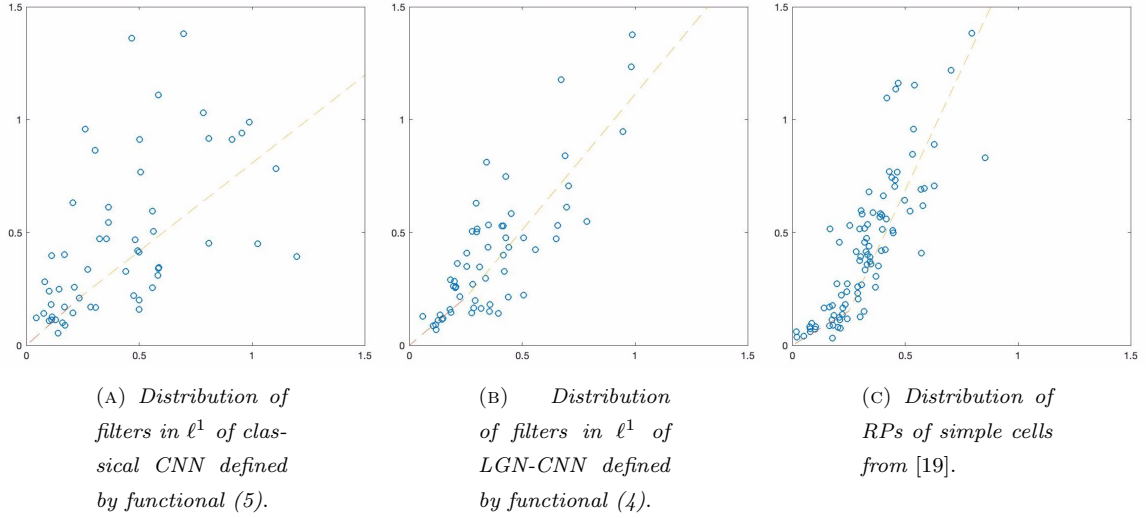


FIGURE 7. Comparison between the statistical distribution on  $(n_x, n_y)$  plane of filters of a classical CNN, of our architecture and of RPs of real data.

the  $y$  direction. This behavior is the same in the case of RPs (figure 7c) in which the distribution has a similar slope of LGN-CNN. This enforces more the link of LGN-CNN with the structure of the visual system motivating us to pursue in this direction.

## 14 LGN-CNN: A BIOLOGICALLY INSPIRED CNN ARCHITECTURE.

### 5. ROTATION SYMMETRY OF $\Psi^0$ : A POSSIBLE SETTING

In this section we define a setting in which it should be possible to study the rotation symmetry of  $\Psi^0$ . The idea is to extend some results on rotation invariant functionals (see [14]).

Let us consider the architecture of a LGN-CNN in which we can split the first convolutional layer composed by only one filter from the rest of the net which will be fixed. Thus this first layer can be approximated by a function  $\Psi^0 : \mathbb{R}^2 \rightarrow \mathbb{R}$ , assuming  $\Psi^0 \in L^1(\mathbb{R}^2)$ . A general image can be defined as a function  $I : \mathbb{R}^2 \rightarrow \mathbb{R}$  where we assume  $I \in L^1(\mathbb{R}^2)$ . Thus the rest of the net will be defined by a functional

$$C : L^1(\mathbb{R}^2) \rightarrow L^1(\mathbb{R}^3)$$

And then we can define

$$(6) \quad F : L^1(\mathbb{R}^2) \times L^1(\mathbb{R}^2) \rightarrow L^1(\mathbb{R}) \quad , \quad F(I, \Psi^0)(x) := \int_{\mathbb{R}^2} C((I * \Psi^0))(z, x) dz.$$

Then  $F(I, \Psi^0)(x)$  is a probability distribution over a set of categories. Thus our aim is to find a function  $\Psi^0$  in such a way that  $F$  approximates well the known functional  $y$  which is defined on a subset  $\Gamma \subset L^1(\mathbb{R}^2)$  of all the images.

Since  $I, \Psi^0 \in L^1(\mathbb{R}^2)$  we can impose that the output of  $F$  has the following properties:

$$F(I, \Psi^0)(x) \geq 0 \quad , \quad \int_{\mathbb{R}} F(I, \Psi^0)(x) dx = 1$$

The idea is to adapt the proof the authors obtained in [14] for a certain class of functionals to our case.

### 6. CONCLUSIONS

The study of the role of the LGN in the visual system and the rotation invariance properties of the RPs of its cells has leaded our research to the introduction of a CNN architecture that mimics this structure. In particular, we have added to a CNN a first convolutional layer composed by a single filter which attains a rotationally symmetric pattern. The filter has inherited this property from the modified architecture of the net.

We have also shown that it is not only rotationally symmetric but it also obtains a LoG shape that approximates the RPs of the LGN cells. This behavior enforces the link between the visual system structure and the architecture of CNNs. Furthermore, we have analyzed the statistical distribution of the filters of the second convolutional layer that attain a Gabor shape even with the introduction of the first layer. We have shown that the statistical distribution becomes closer to the

real data of RPs of simple cells in V1 (from [19]) enriching the connections with the neural structure.

In the future we will face the theoretical problem regarding the rotation symmetry of the first convolutional layer. Furthermore, we will analyze the modifications that in a LGN-CNN occur to the bank of filters of other convolutional layers of deeper architecture, comparing them with neural data.

#### REFERENCES

- [1] A. Coates, H. Lee, A. Y. Ng. An Analysis of Single Layer Networks in Unsupervised Feature Learning. AISTATS, 2011.
- [2] J. G. Daugman. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters, J. Opt. Soc. Am. A2, 1160-1169, 1985.
- [3] DeAngelis, G.C., Ozhawa, I., Freeman, R.D., Receptive-field dynamics in the central visual pathways, Trends in Neuroscience 18, 10 (1995) 451-458.
- [4] S. Dieleman, J. De Fauw, and K. Kavukcuoglu. Exploiting cyclic symmetry in convolutional neural networks. arXiv preprint arXiv:1602.02660, 2016.
- [5] S. Dieleman, K. W. Willett, and J. Dambre. Rotation-invariant convolutional neural networks for galaxy morphology prediction. Monthly Notices of the Royal Astronomical Society, vol. 450, no. 2, pp. 1441- 1459, 2015.
- [6] B. Fasel and D. Gatica-Perez. Rotation-invariant neoperceptron. In Proc. International Conference on Pattern Recognition (ICPR), vol. 3. IEEE, pp. 336-339, 2006.
- [7] R. Gens and P. M. Domingos. Deep symmetry networks. in Advances in Neural Information Processing Systems, pp. 2537-2545, 2014
- [8] B. Gidas, W. N. Ni, L. Nirenberg. Symmetry of positive solutions of nonlinear elliptic equations in RN. Math. Anal. Appl., Part I (L. Nachbin, Ed.), Academic Press, San Diego, 1981.
- [9] F. Girosi and M. Jones and T. Poggio. Regularization Theory and Neural Networks Architectures. Neural Computation, vol. 7, pp. 219-269, 1995.
- [10] Jessell, Thomas M.; Kandel, Eric R.; Schwartz, James H. "27. Central visual pathways". Principles of neural science. New York: McGraw-Hill. pp. 533-540. ISBN 978-0-8385-7701-1. OCLC 42073108, 2000.
- [11] J. P. Jones, L. A. Palmer. An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex. J. Neurophysiol. 58, 1233-1258, 1987.
- [12] D. Laptev, N. Savinov, J. M. Buhmann, and M. Pollefeys. TI-pooling: transformation-invariant pooling for feature learning in convolutional neural networks. arXiv preprint arXiv:1604.06318, 2016.
- [13] T. S. Lee. Image Representation Using 2D Gabor Wavelets. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 18, No. 10, 1996.
- [14] O. Lopes. Radial symmetry of minimizers for some translation and rotation invariant functionals. Journal of differential equations 124, 378388, 1996.
- [15] D. Marcos, M. Volpi, D. Tuia. Learning rotation invariant convolutional filters for texture classification. CoRR, <http://arxiv.org/abs/1604.06720>, 2016.
- [16] N. Montobbio, L. Bonnasse-Gahot, G. Citti, A. Sarti. KerCNNs: biologically inspired lateral connections for classification of corrupted images.

## 16 LGN-CNN: A BIOLOGICALLY INSPIRED CNN ARCHITECTURE.

---

- [17] B.A. Olshausen and D.J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature* 381, 607-609, 1996.
- [18] J. Petitot. *Neurogéométrie de la vision*. Les édition du École Polytechnique, 2009.
- [19] D.L. Ringach. Spatial structure and symmetry of simple-cell receptive fields in macaque primary visual cortex. *J. Neurophysiol.* 88(1):455-63, 2002.
- [20] A. Sarti and G. Citti. On the origin and nature of neurogeometry. *La Nuova Critica*, 2011.
- [21] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, T. Poggio. Robust object recognition with cortex-like mechanisms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 411-426, 2007.
- [22] A. Sherstinsky. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network. *CoRR*, 2018.
- [23] Sundsten, John W.; Nolte, John. *The human brain: an introduction to its functional anatomy*. St. Louis: Mosby. pp. 410-447. ISBN 978-0-323-01320-8. OCLC 47892833, 2001.
- [24] A. Vedaldi and K. Lenc. MatConvNet – Convolutional Neural Networks for MATLAB. *Proceeding of the ACM Int. Conf. on Multimedia*, 2015.
- [25] F. Wu, P. Hu, and D. Kong. Flip-rotate-pooling convolution and split dropout on convolution neural networks for image classification. *arXiv preprint arXiv:1507.08754*, 2015.
- [26] D. Yamins and J. DiCarlo. Using goal-driven deep learning models to understand sensory cortex. *Nature Neuroscience*, Vol 19, 356 EP, 2016.
- [27] D. Yamins, H. Hong, C. Cadieu and J. Dicarlo. Hierarchical modular optimization of convolutional networks achieves representations similar to macaque it and human ventral stream. *Adv. Neural Inf. Process. Syst.* 26, 3093-3101, 2013.