SORBONNE UNIVERSITÉ
UNIVERSITÉ DE BOLOGNE

École Doctorale 386

CAMS, EHESS (Paris) et Département de Mathématiques (Bologne)

This project has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 754362.

Thèse pour obtenir le grade de
DOCTEUR DE L'UNIVERSITÉ SORBONNE UNIVERSITÉ

Discipline: Mathématiques

Présentée et soutenue par:
Federico Bertoni

LIE SYMMETRIES IN CORTICAL INSPIRED CNNS

Sous la direction de :

Alessandro Sarti CAMS (EHESS-CNRS), Paris, France

Giovanna Citti DM, Bologne, Italie

Introduction

The visual system is one of the most studied part of the brain and, in particular, the LGN and V1, which are the first layers that analyze the visual stimulus, are some of the best understood visual areas. The first description of the early visual pathway and its corresponding geometry was proposed by D. H. Hubel and T. N. Wiesel in the '60s and further developed in the subsequent years [Hubel and Wiesel, 1962, 1977; Hubel, 1987]. The retinal input is first processed by the radially symmetric families of cells present in the LGN who are responsible of human contrast perception. Then it reaches V1, whose neurons are not only sensitive to the light intensity of the visual stimulus, but they also show a selectivity to other features, such as e.g. orientation, scale, velocity. Indeed, every cell presents in the visual cortices reacts to a local area of the visual stimulus called *receptive field* (RF), whereas the function that describes its activation in the presence of a visual stimulus is called *receptive profile* (RP) and it will be denoted as Ψ. Every retinal location is associated to an entire set of neurons of V1 called *hyper-columns*, sensitive to all possible instances of the corresponding variable, organized in the so called *orientation maps*.

From the mathematical point of view, several neuromathematical models based on classical differential geometry and Lie groups have been proposed. In the '80s Jan Koenderink in [Koenderink and van Doom, 1987] studied perceptual spaces using differential geometry and William Hoffman in [Hoffman, 1989] developed a model of the visual cortex as a fiber bundle equipped with a contact structure. In the same period Steven Zucker observed experimentally the relation between the measurement of Euclidean curvature and the role of end-stopping cells [Dobbins et al., 1987].

These first models raised the idea that the visual system was a "geometric machine" leading in the '90s to the development of several phenomenological models of vision based on calculus of variations and parabolic partial differential equations: to cite just some pioneering papers, the segmentation model of Mumford-Shah [Mumford and Shah, 1989] and the multiscale analysis of Alvarez, Lions, Morel [Alvarez et al., 1992]. Moreover, in the same years, some neuromathematical models presenting a double neuro-psycho nature have been proposed. David Mumford in [Mumford, 1994] modeled perceptual illusory contours through its elastica curves which are still at the center of contemporary research, but rethought in new mathematical setting. Thus, in '95 in [Williams and Jacobs, 1997] a first stochastic model of illusory contours in the space of position and orientation was proposed by Williams and Jacobs. At the end of the '90s a fundamental contribution of Jean Petitot and Yannick Tondut in [Petitot and Tondut, 1999] reconsidered the Hoffman model of the cortex as a contact bundle, computing the geodesic curves of the non-integrable structure and showing that the curves modeled the perceptual association fields, measured by Fields, Heyes and Hess in [Field et al., 1993]. In this way, they were able to predict the shape of an illusory contour, given its inducturs, and Petitot explicitly introduced the word Neurogeometry to denote the inner geometry of cortical connectivity. In the subsequent years Jean Petitot has further developed his model contributing massively to the progress of this field of research [Petitot, 2003, 2008, 2017].

In the new century, Govanna Citti and Alessandro Sarti in [Sarti et al., 2003; Citti and Sarti, 2006] observed that the cortical structure is actually defined in the SE(2) Euclidean group of rotation and translation equipped with a sub-Riemannian metric. In this fiber bundle structure, they expressed the cortico-cortical propagation in terms of sub-Riemannian diffusion PDE in order to model the image completion phenomenon. Bressloff and Cowan in [Bressloff and Cowan, 2003] studied the dynamics of neural population, modeling the cortex in the same SE(2) group. An impressive result was obtained when the activation map was computed in absence of an external input; indeed, they simulated the effects of chemical drugs obtaining an activation distribution corresponding to visual hallucinations,

as reported in classical literature. In [Duits and Franken, 2010a,b] Remco Duits shifted the focus by working in the SE(2) group from an image processing point of view. He lifted and propagated the visual signal in the $\mathbb{R}^2 \times S^1$ space and, by applying invertible kernels, was able to reconstruct the stimulus without loss of information in the 2D image plane. A phenomenological approach can be found in the entire work of Jean-Michel Morel and his computational Gestalt group [see e.g. Delsolneux et al., 2008; Morel et al., 2010; Limare et al., 2011]. Indeed, he found in the Helmoltz principle the basis of the classical theory of Gestalt obtaining deep theoretical and computational results. Daniel Bennequin approached several neuromathematical problems in terms of invariance, symmetry and ambiguity: in particular, he formalized these problems with principles of "information topology" theory, a new co-homology theory of information based on probability and entropy theory [see e.g. Pham and Bennequin, 2012; Bennequin, 2014]. James Bednar reproduced the geometric morphologies of the visual cortex through brain plasticity principles, recovering the main characteristics of the functional architecture of the visual cortex by a process of learning on a suitable set of stimuli [Stevens et al., 2013; Antolík et al., 2016]. In more recent years, further research has been carried out leading to several developments in this field: in [Citti and Sarti, 2013; Sarti and Citti, 2015; Favali et al., 2017; Boscain et al., 2018; Boscain, Ugo et al., 2018; Baspinar, 2021; Galyaev and Mashtakov, 2021; Jumakulyyev and Schultz, 2021] the authors faced the perceptual completion of figures task on SE(2) group; perception of contours in motion and trajectories has been studied in [Barbieri et al., 2014b]; applications in vessel segmentation have been analyzed in [Zhang et al., 2016; Bekkers et al., 2017; Abbasi-Sureshjani et al., 2018; Yu et al., 2021]; an orientation-dependent contrast perception model inspired by Wilson–Cowan-type equations has been proposed in [Bertalmío et al., 2019; Bertalmío et al., 2021] a model of the functional architecture of V1 from the RPs of simple cells have been proposed in [Montobbio et al., 2019b, 2020].

In particular, in these models the feature space $\mathcal{G}$ usually has the product form $\mathbb{R}^2 \times \mathcal{F}$, where the parameters $(x_0, y_0) \in \mathbb{R}^2$ indicate the retinal location where each RP is centered, while $f \in \mathcal{F}$ encodes the selectivity of the neurons to other

local features of the image.

A visual stimulus can be represented as a function $I = I(x, y)$, where (x, y) represents a retinal location and $I(x, y)$ the corresponding light. Its action z, associated to a kernel defined as $\Psi = \Psi(x, y)$, can be modeled as a linear integral operator as follows

$$z(I) := I * \Psi.$$

In the case of simple cells of V1 $\mathcal{F} = S^1$, i.e. each neuron responds maximally to a certain orientation θ at a specific retinal location (x, y).

Furthermore, in V1 there is experimental evidence of the existence of connections between simple cells of different hyper-columns with similar orientation, called *long-range horizontal connectivity* [see e.g. Ts'o et al., 1986; Bosking et al., 1997]. The connectivity has been described in [Citti and Sarti, 2006] as families of integral curves of two vector fields in the sub-Riemannian structure on $\mathbb{R}^2 \times S^1$. This model has been further developed in [Sanguinetti et al., 2010] where the authors proposed to model connectivity as suitable geometric kernels. In particular, in [Montobbio et al., 2019b] it is proposed to directly relate the shape of connectivity kernels in a family of cells to their RPs $\{\Psi_p\}_{p\in\mathcal{G}}$ as follows:

$$K(p, p_0) := Re\left(\int_{\mathbb{R}^2} \Psi_p(x, y)\overline{\Psi_{p_0}(x, y)}dxdy\right),$$

Many efforts have been made to relate cortical architectures with Convolutional Neural Networks (CNNs). Indeed, the first neural networks have been inspired by a simplification of the structure of the visual system, presenting a hierarchical structure, where each layer receives input from the previous one and provides output to the next one. Despite this simplification, they reached optimal performances in processes typical of the natural visual system, as for example object-detection [Redmon et al., 2016; Ren et al., 2017] or image classification [He et al., 2015; Simonyan and Zisserman, 2015].

More recently, relations between CNNs and human visual system have been widely studied, with the ultimate scope of making the CNN even more efficient in specific tasks. A model of the first cortical layers described as layers of a CNN has been studied in [Serre et al., 2007], whereas in [Yamins et al., 2015; Yamins

and DiCarlo, 2016] the authors were able to study higher areas by focusing on the encoding and decoding ability of the visual system. Recurrent Neural networks have been introduced to implement the horizontal connectivity [e.g. Sherstinsky, 2020], or feedback terms [e.g. Liang and Hu, 2015]. A modification of these nets, more geometric and more similar to the structure of the brain, has been recently proposed in [Montobbio et al., 2019a].

Furthermore, it is well known that both V1 RPs and the first convolutional layer of a CNN are mainly composed by Gabor filters. Biological based models of V1 in terms of Gabor filters have been made in [Serre et al., 2007; Zhang et al., 2019] and the statistic of the RFPs of a macaque's V1 was studied in [Ringach, 2002], but a comparison between these results and the statistics of learned filters is still missing.

Our scope in this thesis is to propose architectures of CNNs in such a way to model the early visual pathway, including the Lateral Geniculate Nucleus and the Horizontal Connectivity of the primary visual cortex. Moreover, we will show how cortically inspired architectures allow to perform contrast perceptual invariance as well as grouping and the emergence of visual percepts. Particularly, the LGN is modeled with a first layer ℓ^0 containing a single filter Ψ^0 that pre-filters the image I. Since the RPs of the LGN cells can be modeled as a LoG, we expect to obtain a radially symmetric filter with a similar shape; to this end, we prove the rotational invariance of Ψ^0 and we study the influence of this filter to the subsequent layer. Indeed, we compare the statistic distribution of the filters in the second layer ℓ^1 of our architecture with the statistic distribution of the RPs of V1 cells of a macaque studied in [Ringach, 2002].

Then, we model the horizontal connectivity of V1 implementing a transition kernel K^1 to the layer ℓ^1. In this setting, we study the vector fields and the association fields induced by the connectivity kernel K^1. To this end, we first approximate the filters bank in ℓ^1 with a Gabor function and use the parameters just found to re-parameterize the kernel. Thanks to this step, the kernel is now re-parameterized into a sub-Riemmanian space $\mathbb{R}^2 \times S^1$. Now we are able to compare the vector and association fields induced by K^1 with the models of the

horizontal connectivity.

The Retinex theory, proposed by Land [Land, 1964], describes the contrast perception phenomenon. Later, he developed this theory with J. McCann in [Land and McCann, 1971], proposing an algorithm and a computer program designed to simulate the Retinex processes taking place in the visual system : it associated to an image I the perceived $\widetilde{I}$. In the following years several works have contributed to the development of the Retinex theory and its application [among others Enroth-Cugell and Robson, 1966; Brainard and Wandell, 1986; Provenzi et al., 2005; Solomon et al., 2006; Lei et al., 2007; Valberg and Seim, 2013; Yeonan-Kim and Bertalmío, 2017]. Some variational approaches have been proposed for example by [Kimmel et al., 2003; Morel et al., 2010; Limare et al., 2011]. In the last two works the authors have formalized the Retinex theory as the solution of the following discrete Poisson PDE

$$-\Delta_d \widetilde{I} = M(I)$$

where Δ_d is the classical discrete Laplacian and M is a modified version of the discrete Laplacian. More recently, a geometrical model relating the architecture of the visual system and the invariances of RPs has been presented in [Citti and Sarti, 2013]. Our scope will be to adapt this approach by using kernels and structures learned by the CNN, instead of modelled kernels.

We will apply cortical inspired CNN to grouping, i.e. the ability of the visual system to perceive the elements in the image as distinct. The main principles that rule how we perceive the elements were described by Kanizsa [Kanizsa et al., 1979] through the Gestalt laws. Then, several models have been proposed to connect the cortical activity with these phenomena [see e.g. Hoffman, 1989; Weiss, 1999; Shi and Malik, 2000]. In [Sarti and Citti, 2015; Favali et al., 2017] the authors put in relation the meanfield equation of Ermentrout and Cowan [Ermentrout and Cowan, 1980] and Bressloff and Cowan [Bressloff et al., 2002; Bressloff and Cowan, 2003] which describes the evolution of the cortical activity depending on a connectivity kernel and the eigenvectors of the affinity matrix. In particular, they proved that the eigenvectors, ordered w.r.t. the corresponding eigenvalues in decreasing order, represent the most salient objects in the image.

The last part of our work is aimed to show that filters and kernels learned by the LGN-CNN with the connectivity layer are able to reproduce visual perceptual phenomena, namely contrast perception, whose main model is the Retinex theory and grouping, whose properties are governed by Gestalt laws. Our purpose is to substitute the action of the RP with the filter learned by the LGN-CNN. If the filter becomes a good good approximation of the associated ΔG_σ, then we shall expect that its inverse will allow to recover the perceived image $\widetilde{I}$ in problems of contrast perception. Indeed, we propose a Retinex algorithm for a learned kernel and apply it to the LGN inspired learned filter Ψ^0 comparing the results with the classical ones. We show that our model reaches similar Retinex abilities with respect to the classical theory and the LoG, the model of the LGN RPs.

On the other hand, we perform the grouping with learned kernels. Indeed, we propose an algorithm that allows to generate an affinity matrix from a bank of filters and, then, finding the first eigenvectors, to locate the perceptual units. We first test our algorithm with a modeled bank of Gabor filters, showing good performances in the case of position-oriented elements but bad ones in the case of clouds of points. Then, we test our algorithm on several learned CNN architecture, generating the affinity matrix from one or more convolutional layers. We show that, thanks to the presence of both Gaussian and Gabor like filters, the kernel generated from these banks of filters are able to detect both position-oriented elements and clouds of points. Furthermore, the use of more convolutional layers enhances the grouping ability of the kernels.

The thesis is organized as follows. The first three chapters contain an overview of the existing literature. Namely, the first chapter contains an overview of the structure of the visual pathway and the group symmetries presents there, as well as a review of the mathematical models of the functional architecture of the visual cortex. In the second chapter, we present the contrast perception phenomena and the grouping, whereas in the third one the main structures of the CNN architectures and the analogies with the visual system. Then, in the fourth and fifth chapters we describe the biologically inspired CNN architectures, one with the LGN layer

[proposed in Bertoni et al., 2022] and one with the transition kernel too [proposed in Bertoni et al., 2021]. Here we show the invariances arising in these models. In the last two chapters we test the ability of filters and kernels learned in CNNs to reproduce perceptual phenomena typical of brain architectures. In the sixth chapter we study the Retinex effects on the LGN inspired learned filter, whereas in the last one we show the grouping results on kernels learned in different CNNs.

Resumé

Le système visuel est l'une des parties du cerveau les plus étudiées et, en particulier, le corps géniculé latéral (CGL) et le cortex visuel primaire (V1), qui sont les premières couches qui analysent le stimulus visuel, sont parmi les zones visuelles les mieux comprises. La première description du premier parcours visuel et de sa géométrie correspondante a été proposée par D. H. Hubel et T. N. Wiesel dans les années 60 et les années suivantes [Hubel and Wiesel, 1962, 1977; Hubel, 1987]. L'entrée rétinienne est d'abord traitée par les familles radialement symétriques des cellules présentes dans le CGL qui sont responsables de la perception du contraste humain. Puis il atteint V1, dont les neurones ne sont pas seulement sensibles à l'intensité lumineuse du stimulus visuel, mais ils montrent également une sélectivité à d'autres caractéristiques, telles que l'orientation, l'échelle, la vitesse. En effet, chaque cellule présente dans les cortices visuels réagit à une zone locale du stimulus visuel appelé champ récepteur (CR), alors que la fonction qui décrit son activation en présence d'un stimulus visuel est appelée profil réceptif (PR) et sera désignée comme Ψ. Chaque emplacement rétinien est associé à un ensemble entier de neurones de V1 appelés *colonne corticale*, sensibles à toutes les instances possibles de la variable correspondante, organisés dans les soi-disant emphcartes d'orientation.

Du point de vue mathématique, plusieurs modèles neuromathématiques basés sur la géométrie différentielle classique et les groupes de Lie ont été proposés. Dans les années 80, Jan Koenderink dans [Koenderink and van Doom, 1987] a étudié les espaces perceptifs à l'aide de la géométrie différentielle et William Hoffman dans [Hoffman, 1989] a développé un modèle du cortex visuel sous la

forme d'un faisceau de fibres équipé d'une structure de contact. Dans la même période, Steven Zucker a observé expérimentalement la relation entre la mesure de la courbure euclidienne et le rôle des cellules terminales [Dobbins et al., 1987]. Ces premiers modèles ont soulevé l'idée que le système visuel était une "machine géométrique", ce qui a conduit dans les années 90 au développement de plusieurs modèles phénoménologiques de la vision basés sur le calcul des variations et les équations aux dérivées partielles paraboliques : pour ne citer que quelques articles pionniers, le modèle de segmentation de Mumford-Shah [Mumford and Shah, 1989] et l'analyse multi-échelle d'Alvarez, Lions, Morel [Alvarez et al., 1992]. De plus, dans les mêmes années, des modèles neuromathématiques aiant une double nature neuro-psychologique ont été proposés. David Mumford dans [Mumford, 1994] a modélisé des contours illusoires perceptifs à travers ses courbes élastiques qui sont toujours au centre de la recherche contemporaine, mais repensées dans un nouveau cadre mathématique. Ainsi, en 1995 dans [Williams and Jacobs, 1997] un premier modèle stochastique de contours illusoires dans l'espace de position et d'orientation a été proposé par Williams et Jacobs. A la fin des années 90, une contribution fondamentale de Jean Petitot et Yannick Tondut dans [Petitot and Tondut, 1999] a reconsidéré le modèle de Hoffman du cortex comme structure de contact, calculant les courbes géodésiques de la structure non intégrable et montrant que les courbes modélisées les champs d'association perceptifs, mesurés par Fields, Heyes et Hess dans [Field et al., 1993]. De cette façon, ils ont pu prédire la forme d'un contour illusoire, compte tenu de ses inductances, et Petitot a explicitement introduit le mot Neurogéométrie pour désigner la géométrie interne de la connectivité corticale. Au cours des années suivantes, Jean Petitot a fait évoluer son modèle en contribuant massivement aux progrès de ce domaine de recherche [Petitot, 2003, 2008, 2017].

Au nouveau siècle, Govanna Citti et Alessandro Sarti dans [Sarti et al., 2003; Citti and Sarti, 2006] ont observé que la structure corticale est en fait définie dans le groupe de rotation et de translation euclidien SE(2) doté d'une métrique sous-riemannienne. Dans cette structure de faisceaux de fibres, ils ont exprimé la propagation corticale en termes de diffusion sous-riemannienne d'EDP afin de modéliser le phénomène de complétion d'image. Bressloff et Cowan dans [Bressloff

and Cowan, 2003] ont étudié la dynamique de la population neuronale, modélisant le cortex dans le même groupe SE(2). Un résultat impressionnant a été obtenu lorsque la carte d'activation a été calculée en l'absence d'une entrée externe ; en effet, ils ont simulé les effets de drogues chimiques en obtenant une distribution d'activation correspondant à des hallucinations visuelles, comme on a rapporté dans la littérature classique. Dans [Duits and Franken, 2010a,b] Remco Duits a déplacé l'attention travaillant dans le groupe SE(2) du point de vue du traitement d'image. Il a soulevé et propagé le signal visuel dans l'espace $\mathbb{R}^2 \times S^1$ et, appliquant des noyaux inversibles, a pu reconstruire le stimulus sans perte d'information dans le plan image 2D. Une approche phénoménologique se retrouve dans l'ensemble de l'œuvre de Jean-Michel Morel et de son groupe computationnel de Gestalt [voir par exemple Delsolneux et al., 2008; Morel et al., 2010; Limare et al., 2011]. En effet, il a trouvé dans le principe de Helmoltz la base de la théorie classique de la Gestalt obtenant des résultats théoriques et informatiques approfondis. Daniel Bennequin a abordé plusieurs problèmes neuromathématiques en termes d'invariance, de symétrie et d'ambiguïté : il a notamment formalisé ces problèmes avec les principes de la théorie de la "topologie de l'information", une nouvelle théorie de la cohomologie de l'information basée sur la théorie des probabilités et de l'entropie [voir par exemple Pham and Bennequin, 2012; Bennequin, 2014]. James Bednar a reproduit les morphologies géométriques du cortex visuel à travers les principes de plasticité cérébrale, récupérant les principales caractéristiques de l'architecture fonctionnelle du cortex visuel par un processus d'apprentissage sur un approprié ensemble de stimuli [Stevens et al., 2013; Antolík et al., 2016]. Ces dernières années, d'autres recherches ont été menées aboutissant à plusieurs développements dans ce domaine : dans [Citti and Sarti, 2013; Sarti and Citti, 2015; Favali et al., 2017; Boscain et al., 2018; Boscain, Ugo et al., 2018; Baspinar, 2021; Galyaev and Mashtakov, 2021; Jumakulyyev and Schultz, 2021], les auteurs ont été confrontés à la tâche d'achèvement perceptif des figures sur le groupe SE(2) ; la perception des contours en mouvement et des trajectoires a été étudiée dans [Barbieri et al., 2014b] ; applications dans la segmentation des vaisseaux ont été analysées dans [Zhang et al., 2016; Bekkers et al., 2017; Abbasi-Sureshjani et al., 2018; Yu et al.,

2021] ; un modèle de perception de contraste dépendant de l'orientation inspiré des équations de type Wilson – Cowan a été proposé dans [Bertalmío et al., 2019; Bertalmío et al., 2021] un modèle de l'architecture fonctionnelle de V1 à partir des PR de cellules simples a été proposé dans [Montobbio et al., 2019b, 2020].

En conséquence, l'espace de fonction $\mathcal{G}$ a généralement la forme du produit $\mathbb{R}^2 \times \mathcal{F}$, où les paramètres $(x_0, y_0) \in \mathbb{R}^2$ indiquent l'emplacement rétinien où chaque PR est centré, tandis que $f \in \mathcal{F}$ encode la sélectivité des neurones vers d'autres caractéristiques locales de l'image.

Un stimulus visuel peut être représenté comme une fonction $I = I(x, y)$, où (x, y) représente un emplacement rétinien et $I(x, y)$ la lumière correspondante. Son action z, associée à un noyau défini comme $\Psi = \Psi(x, y)$, peut être modélisée en tant qu'un opérateur intégral linéaire comme suit

$$z(I) := I * \Psi.$$

Dans le cas des cellules simples de V1 $\mathcal{F} = S^1$, i.e. chaque neurone répond au maximum à une certaine orientation θ à un emplacement rétinien spécifique (x, y). En outre, dans V1 il y a des preuves expérimentales de l'existence des connexions entre des cellules simples qui appartiennent à différentes colonnes corticales avec une orientation similaire, appelée *connectivité horizontale à longue portée* [par example Ts'o et al., 1986; Bosking et al., 1997]. La connectivité a été décrite dans [Citti and Sarti, 2006] comme des familles de courbes intégrales des deux champs vectoriels dans la structure sous-riemannienne sur $\mathbb{R}^2 \times S^1$. Ce modèle a été développé dans [Sanguinetti et al., 2010] où les auteurs ont proposé de modéliser la connectivité comme noyaux géométriques appropriés. En particulier, dans [Montobbio et al., 2019b] il est proposé de relier directement la forme des noyaux de connectivité dans une famille de cellules à leur PRs $\{\Psi_p\}_{p\in\mathcal{G}}$ comme suit :

$$K(p, p_0) := Re\left(\int_{\mathbb{R}^2} \Psi_p(x, y)\overline{\Psi_{p_0}(x, y)}dxdy\right),$$

Des nombreux efforts ont été faits pour relier les architectures corticales avec les réseaux neuronaux convolutifs (CNNs). En effet, les premiers réseaux neuronaux ont été inspirés par une simplification de la structure du système visuel, présentant

une structure hiérarchique, où chaque couche reçoit l'entrée de la précédente et fournit la sortie à la suivante. Malgré cette simplification, ils ont atteint des valeurs optimales dans les processus typiques du système visuel naturel, comme par exemple la détection d'objets [Redmon et al., 2016; Ren et al., 2017] ou la classification d'images [He et al., 2015; Simonyan and Zisserman, 2015].

Plus récemment, les relations entre les CNNs et le système visuel humain ont été largement étudiées, dans le but ultime de rendre le CNN encore plus efficace dans des tâches spécifiques. Un modèle de premières couches corticales décrites comme couches d'un CNN a été étudié dans [Serre et al., 2007], alors que dans [Yamins et al., 2015; Yamins and DiCarlo, 2016] les auteurs ont pu étudier des zones plus élevées en se concentrant sur l'encodage et la capacité de décodage du système visuel. Des réseaux neuronaux récurrents ont été introduits pour mettre en œuvre la connectivité horizontale [par exemple Sherstinsky, 2020], ou termes de rétroaction [par exemple Liang and Hu, 2015]. Une modification de ces filtres, plus géométrique et plus similaire à la structure du cerveau, a été récemment proposée dans [Montobbio et al., 2019a]. De plus, il est bien connu que les V1 PRs et la première couche convolutionnelle d'un CNN sont principalement composées de filtres de Gabor. Des modèles biologiques de V1 en termes des filtres de Gabor ont été réalisés dans [Serre et al., 2007; Zhang et al., 2019] et la statistique des PRs de la V1 d'un macaque a été étudiée dans [Ringach, 2002], néanmoins une comparaison entre ces résultats et les statistiques des filtres appris est toujours manquante.

Notre but dans cette thèse est de proposer des architectures de CNN afin de modéliser la première voie visuelle, y compris le noyau géniculé latéral et la connectivité horizontale du cortex visuel primaire. En outre, nous allons montrer comment les architectures d'inspiration corticale permettent d'effectuer de l'invariance perceptuelle au contraste ainsi que le groupement et l'émergence des perceptions visuelles. En particulier, la CGL est modélisée avec une première couche ℓ^0 contenant un seul filtre Ψ^0 qui pré-filtre l'image I. Comme les PR des cellules CGL peuvent être modélisées en tant que LoG, nous nous attendons à obtenir un filtre

radialement symétrique de forme similaire ; à cette fin, nous prouvons l'invariance rotationnelle de Ψ^0 et nous étudions l'influence de ce filtre sur la couche suivante. En effet, nous comparons la distribution statistique des filtres de la deuxième couche ℓ^1 de notre architecture avec la distribution statistique des PR des cellules V1 d'un macaque étudiée dans [Ringach, 2002].

Ensuite, nous modélisons la connectivité horizontale de V1 implémentant un noyau de transition K^1 vers la couche ℓ^1. Dans ce cadre, nous étudions les champs vectoriels et les champs d'association induits par le noyau de connectivité K^1. À cette fin, nous rapprochons d'abord la banque de filtres en ℓ^1 avec une fonction de Gabor et utilisons les paramètres trouvés pour reconfigurer le noyau. Grâce à cette étape, le noyau est maintenant reconfiguré dans un espace sous-Riemmanien $\mathbb{R}^2 \times S^1$. Nous sommes maintenant en mesure de comparer les champs vectoriels et d'association induits par K^1 avec les modèles de la connectivité horizontale.

La théorie Retinex, proposée par Land [Land, 1964], décrit le phénomène de perception de contraste. Plus tard, il a développé cette théorie avec J. McCann dans [Land and McCann, 1971], proposant un algorithme et un programme informatique conçu pour simuler les processus Retinex se déroulant dans le système visuel: il associait à une image I la perception $\widetilde{I}$. Dans les années suivantes, plusieurs travaux ont contribué au développement de la théorie Retinex et son application [entre autres Enroth-Cugell and Robson, 1966; Brainard and Wandell, 1986; Provenzi et al., 2005; Solomon et al., 2006; Lei et al., 2007; Valberg and Seim, 2013; Yeonan-Kim and Bertalmío, 2017]. Certaines approches variationnelles ont été proposées par exemple par [Kimmel et al., 2003; Morel et al., 2010; Limare et al., 2011]. Dans les deux derniers travaux, les auteurs ont formalisé la théorie de Retinex comme solution du discret Poisson EDP suivant

$$-\Delta_d \widetilde{I} = M(I)$$

où Δ_d est le Laplacien discret classique et M est une version modifiée du Laplacien discret. Plus récemment, un modèle géométrique relatif à l'architecture du système visuel et aux invariances des PRs a été présenté dans [Citti and Sarti, 2013]. Notre objectif sera d'adapter cette approche en utilisant des noyaux et des structures apprises par CNN, au lieu des noyaux modélisés.

Nous appliquerons des CNNs inspirées aux corticales pour le regroupement, c.-à-d. la capacité du système visuel de percevoir les éléments de l'image comme étant distincts. Les principes fondamentaux qui régissent la façon dont nous percevons les éléments ont été décrits par Kanizsa [Kanizsa et al., 1979] à travers les lois de la Gestalt. Ensuite, plusieurs modèles ont été proposés pour relier l'activité corticale à ces phénomènes [voir par ex. Hoffman, 1989; Weiss, 1999; Shi and Malik, 2000]. Dans [Sarti and Citti, 2015; Favali et al., 2017] les auteurs mettent en relation l'équation "meanfield" d'Ermentrout et de Cowan [Ermentrout and Cowan, 1980] et Bressloff et Cowan [Bressloff et al., 2002; Bressloff and Cowan, 2003] qui décrit l'évolution de l'activité corticale en fonction d'un noyau de connectivité et des vecteurs propres de la matrice d'affinité. En particulier, ils ont prouvé que les vecteurs propres, ont commandé par rapport aux correspondant valeurs propres par ordre décroissant, représentent les objets les plus saillants dans l'image.

La dernière partie de notre travail vise à montrer que les filtres et les noyaux appris par la CGL-CNN avec la couche de connectivité sont capables de reproduire des phénomènes perceptifs visuels, notamment la perception du contraste, dont le modèle principal est la théorie et le regroupement Retinex, dont les propriétés sont régies par les lois de la Gestalt. Notre but est de substituer l'action de la PR au filtre appris par la CGL-CNN. Si le filtre devient une bonne approximation du ΔG_σ associé, alors on s'attendra à ce que son inverse permette de récupérer l'image perçue $\widetilde{I}$ en problèmes de perception du contraste. En effet, nous proposons un algorithme Retinex pour un noyau entraîné et l'appliquons au filtre entraîné Ψ^0 inspiré de CGL, en comparant les résultats avec les résultats classiques. Nous montrons que notre modèle atteint des capacités Retinex similaires par rapport à la théorie classique et au LoG, le modèle des PR CGL.

D'autre part, nous effectuons le regroupement avec des noyaux appris. En effet, nous proposons un algorithme qui permet de générer une matrice d'affinité à partir d'une banque de filtres et, ensuite, de trouver les premiers vecteurs propres, de localiser les unités perceptuelles. Nous testons d'abord notre algorithme avec une banque modélisée de filtres Gabor, montrant de bonnes performances dans

le cas d'éléments orientés position, tandis que de mauvaises performances dans le cas de nuages de points. Ensuite, nous testons notre algorithme sur plusieurs architectures CNN apprises, générant la matrice d'affinité à partir d'une ou de plusieurs couches convolutionnelles. Nous montrons que, grâce à la présence de filtres gaussiens et Gabor, le noyau généré à partir de ces banques de filtres est capable de détecter à la fois des éléments orientés position et des nuages de points. En outre, l'utilisation des couches plus convolutionnelles améliore la capacité de regroupement des noyaux.

La thèse est organisée comme suit. Les trois premiers chapitres contiennent une vue d'ensemble de la littérature existante. Notamment, le premier chapitre présente un aperçu de la structure de la voie visuelle et des symétries de groupe qui y sont présentées, ainsi qu'un examen des modèles mathématiques de l'architecture fonctionnelle du cortex visuel. Dans le deuxième chapitre, nous présentons les phénomènes de perception du contraste et le regroupement, tandis que dans le troisième, les principales structures des architectures CNN et les analogies avec le système visuel. Ensuite, dans les quatrième et cinquième chapitres, nous décrivons les architectures CNN inspirées biologiquement, l'une avec la couche CGL [proposée dans Bertoni et al., 2022] et l'autre en ajoutant le noyau de transition [proposée dans Bertoni et al., 2021]. Ici, nous montrons les invariances qui surgissent dans ces modèles. Dans les deux derniers chapitres, nous testons la capacité des filtres et des noyaux appris dans les CNNs à reproduire les phénomènes perceptifs typiques des architectures cérébrales. Dans le sixième chapitre, nous étudions les effets Retinex sur le filtre CGL inspiré appris, alors que dans le dernier nous montrons les résultats de regroupement sur les noyaux appris dans différents CNNs.

Contents

List of Figures

List of Tables

Chapter 1

The functional architecture of the Visual System

In this chapter we recall some of the main properties of the visual system that are useful for our research and refer the reader for a more general description to [E. R. Kandel and Jessell., 1993; Nolte and Sundsten, 2002]. We briefly summarize the structure of the visual system focusing on the Lateral Geniculate Nucleus (LGN) and the Primary Visual Cortex (V1) and we describe the geometric properties of the Receptive Profiles (RPs) of cells in both structures.

1.1 Architecture of the visual system

The visual system is one of the most studied and most understood part of the brain, composed by many cortices that elaborate the visual signal. The first element of this structure is the retina, a light-sensitive layer of tissue which receives the visual stimulus and translates it into electrical impulses. The latter first reach the LGN whose cells pre-process the visual stimulus and send the elaborated impulse to V1. Here the cells of V1 process the information and send the output to all the other layers of the visual system. Furthermore, each cortex receives information from other cortices, processes it through horizontal connectivity that allows cells of the same cortex to communicate with each others, forwards it to higher areas and

sends feedback to previous ones. The structure is very complex and not totally ordered as physiologically described for example in [Hubel, 1987].

1.1.1 Receptive Profiles of the LGN and V1 cells

Our main interest lies in the cells of the LGN and in the simple cells of V1. Each cell in the visual system receives the electrical impulse from a portion of the retina Ω called receptive field (RF). The RF of each cell is divided in excitatory and inhibitory areas, activated by the light. As a result, the behavior of each cell can be modeled as a function $\Psi : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ called receptive profile (RP), positive in excitatory areas and negative in the inhibitory ones. If the excitatory areas are activated the firing rate of the cell increases whereas it decreases in case of inhibitory areas activation.

As concern the LGN, figure 1.1 on the left shows a recording of the RP of one cell and on the right its approximation [see e.g. DeAngelis et al., 1995]. Indeed, the RP of LGN cells can be modeled as a Laplacian of Gaussian (LoG):

$$\Psi_{LoG}(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}, \tag{1.1}$$

where σ denotes the standard deviation of the Gaussian function [see e.g. Petitot and Tondut, 1999; Petitot, 2008, 2017]. Furthermore, in the LGN there are two types of cells RPs, respectively with excitatory or inhibitory internal region. In the second case the RP can be modeled by minus the LoG.

As concern V1, there exist several types of cells with different RPs, called simple cells and complex cells. We are mainly interested in the first kind that has sharply-tuned RPs with a preferred orientation. Figure 1.2 shows in the first row the RPs of two simple cells of V1 and in the second one their approximation. Indeed, an established model for V1 simple cells is represented by a bank of Gabor filters $\{\Psi_{x_0,y_0,\theta,\sigma}\}_{(x_0,y_0,\theta,\sigma)\in\mathbb{R}^2\times S^1\times\mathbb{R}^2_+}$ built by translations $T_{(x_0,y_0)}$, rotations R_θ and dilations D_{σ_x,σ_y} of the filter

$$\Psi_{0,0,0,1}(x,y) = Ae^{-\frac{x^2+y^2}{2}}\cos(2\pi fx + \phi), \tag{1.2}$$

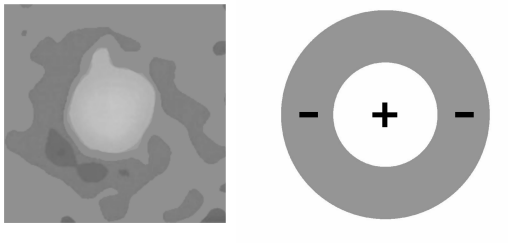


Figure 1.1 – On the left: RFP of an LGN cell where the excitatory area is in white and the inhibitory one is in gray. On the right: Its approximation by a LoG. *From*: [DeAngelis et al., 1995].

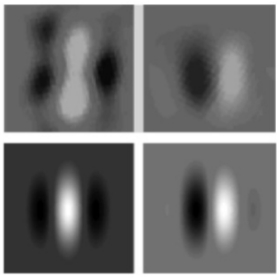


Figure 1.2 – First row: RFPs of two simple cells of V1 where the excitatory area is in white and the inhibitory one is in black. Second row: their approximations by Gabor functions. *From*: [Sarti and Citti, 2011].

where A is the amplitude and f is the frequency of the filter and ϕ is the phase which indicates if the Gabor filter is even or odd. See e.g. [Daugman, 1985] and [Lee, 1996].

Furthermore, several classes of visual cells, as e.g. the ones just described, are shown to act, to a first approximation, as linear filters on an optic signal: the response of this kind of cells to a visual stimulus, defined as a function on the retina $I : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$, is given by the integral of I against the RP Ψ of the neuron:

$$z(I) := \int I(x, y)\Psi(x, y)dxdy. \tag{1.3}$$

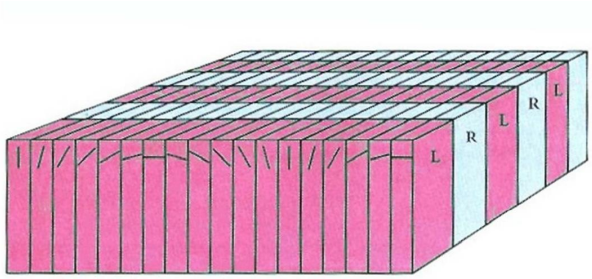


Figure 1.3 – Cube scheme of V1 by Hubel and Wiesel. Cells belonging to the same column have a similar receptive profiles, the orientation hyper-columns are arranged tangentially to the cortical sheet. *From:* [Hubel, 1987].

1.1.2 The hyper-columnar structure and non maxima suppression

In the 70s Hubel and Wiesel discovered that V1 is organized in the so called hyper-columnar structure [Hubel and Wiesel, 1962]. This means that for each retinal point (x, y) there is an entire set of cells each one sensitive to a specific instance of the considered feature, organized in "pinwheel" arrangements. The set of RPs of cells reacting to a specific retinal region is usually represented by a bank of linear filters $\{\Psi_p\}_{p\in\mathcal{G}} \subseteq L^2(\mathbb{R}^2)$ [for some references see e.g. Petitot and Tondut, 1999; Citti and Sarti, 2006; Sarti et al., 2008; Petitot, 2008, 2003]. The feature space $\mathcal{G}$ is typically described as a group of transformations of the plane $\{T_p,\ p \in \mathcal{G}\}$ under which the entire filter bank is invariant. Indeed, each profile Ψ_p can be obtained from any other profile Ψ_q through the transformation T_{p-q}. Furthermore, the feature space $\mathcal{G}$ usually has the product form $\mathbb{R}^2 \times \mathcal{F} \ni (x, y, f)$, where the parameters $(x, y) \in \mathbb{R}^2$ indicate the retinal location where each RP is centered, while $f \in \mathcal{F}$ encodes the selectivity of the neurons to other local features of the image, such as orientation and scale.

The figure 1.3 is a scheme of the hyper-columnar structure when the considered feature is orientation. In this case the space of feature can be modelled by $\mathcal{F} = S^1$ and so the higher dimensional cortical space is given by $\mathcal{G} = \mathbb{R}^2 \times S^1$. At a certain scale and resolution there exists a set of neurons centered at the same retinal

position (x, y) in which each neuron responds maximally to a certain orientation θ. All the possible orientations are expressed and a triple (x, y, θ) can be associated to each neuron.

When a visual stimulus is projected on a retinal point (x, y) all neurons belonging to the hyper-column over the point (x, y) will be activated and provide the response given by the formula (1.3). The one with the same orientation of the stimulus is maximally activated, giving rise to orientation selectivity. Indeed the cortex is equipped with a neural circuitry, called intracortical circuitry, that is able to keep the orientation of maximal response of cells' outputs. Furthermore, among the cells in the same hyper-column is present a short-range connectivity that induces excitation between cells with close orientation and inhibition between cells with different orientation. This kind of connectivity for V1 simple cells has been modelled as a "Mexican hat"-like function in [Bressloff and Cowan, 2002]. For a general feature described by a space $\mathcal{F}$ the process is the same and the non maxima suppression principle is modelled by:

$$z(I)(x_0, y_0, \overline{f}_0) = \max_{f_0 \in \mathcal{F}} z(I)(x_0, y_0, f_0) \tag{1.4}$$

where (x_0, y_0) is the considered retinal position and $\mathcal{F}$ represents the set of all possible instances. Within each hyper-column the feature associated to the maximum output is selected and all the others are suppressed, leading to the selection of the maximal responding feature. In this way our brain is able to locally detect a specific feature and to discard all the other possibility.

1.1.3 Statistics of V1 simple cells RPs

As just described, the RPs of V1 cells can be modeled as Gabor functions (1.2). Recently, Ringach in [Ringach, 2002] has proved that the RPs are not uniformly distributed with respect to all the Gabor parameters, but they have a very particular statistic. Indeed, Ringach defines two coefficients n_x and n_y which estimate the elongation in x and y directions respectively

$$(n_x, n_y) = (\sigma_x \cdot f, \sigma_y \cdot f). \tag{1.5}$$

In particular, if $f = 0$ the Gabor function in (1.2) simplifies to a Gaussian since the cosine becomes a constant. Otherwise the Gabor function is elongated in its preferred orientation θ. In order to study the statistics of recorded RPs of V1 simple cells, Ringach has followed these steps:

- Fitting the RPs with Gabor function $\Psi_{x_0,y_0,\theta,\sigma}$ built by translation $T_{(x_0,y_0)}$, rotation R_θ and dilation D_{σ_x,σ_y} of the filter defined in eq. (1.2);

- Comparing the results on $(n_x, n_y) = (\sigma_x \cdot f, \sigma_y \cdot f)$ plane.

Figure 4.10c shows the statistical distribution of RPs of V1 cells in monkeys in (n_x, n_y) plane obtained by Ringach in [Ringach, 2002]. In [Barbieri et al., 2014a] the authors have studied the same statistical distribution with respect to the Uncertainty Principle associated to the task of detection of position and orientation.

1.2 The horizontal connectivity

From the neurophysiological point of view, there is experimental evidence of the existence of connections between simple cells of different hyper-columns. It is the so called *long-range horizontal connectivity* that is responsible for the cortico-cortical propagation of the neural activity between hyper-columns.

In [Ts'o et al., 1986] correlation techniques have been used in order to estimate the correlation between connectivity and orientation of cells. Recently techniques of optical imaging allowed to measure the horizontal connectivity (see figure 1.4) by injecting a tracer (biocytin) in a simple cell and recording the trajectory of the tracer [see Bosking et al., 1997]. In this way it has been possible to clarify properties of horizontal connections on V1 of the tree shrew and it has been revealed that the linked cells of different hyper-columns not only share the angle of tuning but also the axis corresponding to the orientation is roughly the same.

It is believed that the long-range horizontal connections of V1 constitutes the neural implementation of contour completion, i.e. the ability to group

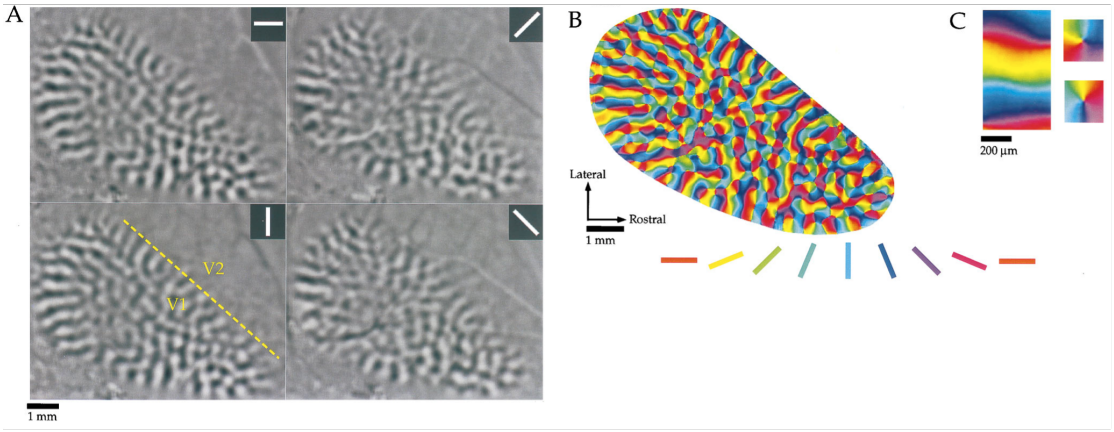


Figure 1.4 – **(A)** Images obtained for four stimulus angles $\theta = 0, 45, 90, 135$ degrees. Dark areas were preferentially activated by a stimulus with orientation θ; light areas were active during presentation of the orthogonal angle. **(B)** The orientation map obtained by vector summation of data obtained for each angle. The orientation preference of each location (x, y) is color-coded according to the key shown below. **(C)** Enlarged portions of the map show linear two zones (left) and two pinwheel arrangements (right). *From:* [Bosking et al., 1997].

local edge items into extended curves. This perceptual phenomenon has been described through *association fields* [Field et al., 1993], characterizing the geometry of the mutual influences between oriented local elements. See for example figure 1.5A from the experiment of Field, Heyes and Hess. Then, Petitot and Tondut in [Petitot and Tondut, 1999] proposed a geometric model of the functional architecture of V1 that also models the association fields. More recently, association fields have been described in Citti and Sarti [2006] as families of integral curves of the two vector fields

$$\vec{X_1} = (\cos\theta, \sin\theta, 0), \quad \vec{X_2} = (0, 0, 1) \tag{1.6}$$

that generates the sub-Riemannian structure on $\mathbb{R}^2 \times S^1$. Figure 1.5B shows the 3-D constant coefficients integral curves of the vector fields (1.6) and figure 1.5C their 2-D projection. These integral curves can be also modeled as the solution of the following ordinary differential equation:

$$\gamma'(t) = X_1(\gamma(t)) + kX_2(\gamma(t)).$$

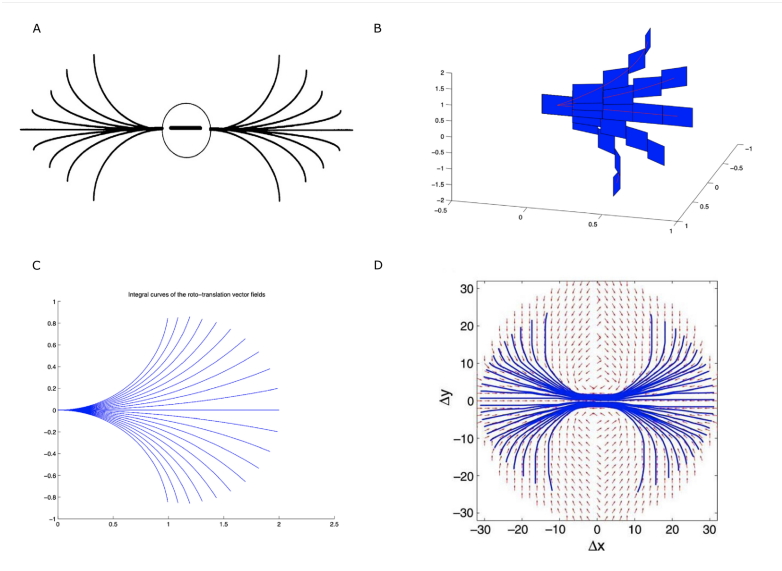


Figure 1.5 – **(A)** Association fields from the experiment of Field, Hayes and Hess, *from* [Field et al., 1993]. **(B)** 3D representation of the association field with contact planes of integral curves of the fields (1.6) with varying values of the parameter k, *from* [Citti and Sarti, 2006]. **(C)** Integral curves of the fields (1.6) with varying k, *from* [Citti and Sarti, 2006]. **(D)** The vector field of unitary vectors oriented with the maximal edge co-occurrence probability (red) with superimposed its integral curves (blue), *from* [Sanguinetti et al., 2010].

The curves starting from $(0, 0, 0)$ can be rewritten explicitly in the following way:

$$x = \frac{1}{k} sin(kt), \qquad y = \frac{1}{k}(1 - cos(kt)), \qquad \theta = kt. \tag{1.7}$$

while integral curves starting from a general point (x_0, y_0, θ_0) can be obtained from equations (1.7) by translations $T_{(x_0,y_0)}$ and rotations R_θ.

The probability of reaching a point (x, y, θ) starting from the origin and moving along the stochastic counterpart of these curves can be characterized as the fundamental solution of a second order differential operator expressed in terms of the vector fields $\vec{X}_1$, $\vec{X}_2$. This is why the fundamental solution of the sub-Riemannian

heat kernel or Fokker Planck (FP) kernel have been proposed as alternative models of the cortical connectivity.

This viewpoint based on connectivity kernels was further exploited in [Montobbio et al., 2020]: the model of the cortex was reformulated in terms of metric spaces, and the long range connectivity kernel directly expressed in terms of the cells RPs: in this way a strong link was established between the geometry of long range and feedforward cortical connectivity. Finally, in Sanguinetti et al. [2010] an important relation between these models of cortical connectivity and statistics of edge co-occurrence in natural images was proved; indeed, the FP fundamental solution has become a good model also for the natural image statistics. Moreover, thanks to a parameterization of the connectivity kernel in terms of position and orientation, in Sanguinetti et al. [2010] they obtained the 2-D vector field represented in red in figure 1.5D , whose integral curves (depicted in blue) offer an alternative model of association fields, learned from images.

1.2.1 A metric model of cortical connectivity

Let us recall the metric model of the cortex proposed in [Montobbio et al., 2019b]. First, we would like to choose a bank of linear, real or complex valued, filters that model the RPs of a set of simple cells of V1. In order to do so we can fix a feature space $\mathcal{G}$ that in general has the form $\mathcal{G} = \mathbb{R}^2 \times \mathcal{F}$. Then we can introduce the following bank of filters:

$$\{\Psi_p\}_{p\in\mathcal{G}} \subseteq L^2(\mathbb{R}^2)$$

where each point $(x, y) \in \mathbb{R}^2$ indicates the retinal position at which the filter is centered whereas $f \in \mathcal{F}$ encodes the specific feature that our bank of filters will detect. In the case of Gabor filters the space $\mathcal{F} = S^1$ but we can consider any other spaces. For example, we can consider filters modelling other types of cells or even filters learned by automatic algorithms.

The idea is now to understand how each pair of filters is correlated with each other. In the case of Gabor filters we know for example that filters that share a similar orientation can communicate. Then we can define for every $p, p_0 \in \mathcal{G}$ the

generating kernel as:

$$K(p, p_0) := Re\left(\int_{\mathbb{R}^2} \Psi_p(x, y)\overline{\Psi_{p_0}(x, y)}dxdy\right), \tag{1.8}$$

which expresses the strength of correlation between them. Since we would like to obtain a connectivity kernel that mimics the behaviour of the horizontal connectivity we have introduced an iterative procedure. This idea follows from the reproducing property of the kernel of a heat equation [see Montobbio et al., 2020]. Indeed, fixing a point p_0 and considering the generating kernel computed around it

$$p \longmapsto K(p, p_0) =: K_1^{p_0}(p),$$

we can define the connectivity kernel at the n-th step as

$$K_{n+1}^{p_0}(p) := \int_{\mathcal{G}} K(p, q)\nu(K_n^{p_0}(q))d\mu(q) \tag{1.9}$$

where ν is a non-linear activation function and μ is a suitable measure defined on $\mathcal{G}$ [see Montobbio et al., 2020]. Furthermore this approach allows to gain more information about the correlation between pairs of filters and in presence of a visual stimulus I we can use the connectivity kernel to express the correlation between points of I. Note that we will apply this method also to learned filters which are represented as small matrices (e.g. 7×7, 5×5) and it will allow to enlarge the area in which points can communicate with each other.

1.3 Group symmetries in the early visual pathway

Over the years, numerous models of the functional architecture of the early visual pathway has been proposed in terms of geometric invariances arising in its organization, e.g. in the spatial arrangement of cell tuning across retinal locations, or in the local configuration of single neuron selectivity. Many classes of visual cells act as linear filters on a visual stimulus I defined as the function (1.3). This is the case for cells in the LGN and for simple cells in V1 [see e.g. Petitot and Tondut,

1999; Citti and Sarti, 2006; Petitot, 2008, 2017] that reacts to specific retinal regions, i.e. the RFs. Each localized area of the retina is known to be associated with a bank of similarly tuned cells [see e.g. Hubel and Wiesel, 1977; Sarti and Citti, 2015], yielding an approximate invariance of their RPs under translations.

Furthermore, the RPs of the cells can usually be embedded in a feature space $\mathcal{G} = \mathbb{R}^2 \times \mathcal{F}$, where $\mathcal{F}$ encodes the selectivity of the neurons to other local features of the image w.r.t. the bank of cells is usually invariant.

1.3.1 Rotational symmetry in the LGN

An essential elaboration step for human contrast perception is represented by the processing of retinal inputs via the radially symmetric families of cells present in the LGN [Hubel and Wiesel, 1977; Hubel, 1987]. The RPs of such cells can be approximated by a LoG (1.1). A model of human contrast perception, called Retinex theory, has been proposed by Land in [Land, 1964]. We describe it more in detail in chapter 2.

1.3.2 Roto-translation symmetries in V1 and the lateral connectivity

Several mathematical models have been proposed to describe the invariances in the functional architecture of V1. The sharp orientation tuning of simple cells is the starting point of most descriptions. This orientation selectivity characterizes the response of each neuron to feedforward inputs; however, it is also present in the *horizontal* connections taking place between neurons of V1. These connections grant facilitatory influences for cells that are similarly oriented; furthermore, the connections departing from each neuron spread anisotropically, concentrating along the axis of its preferred orientation [see e.g. Bosking et al., 1997].

The evolution in time $t \mapsto h(p,t)$ of the activity of the neural population at $p \in \mathbb{R}^2 \times \mathcal{F}$ is assumed in [Bressloff and Cowan, 2003] to satisfy a Wilson-Cowan

equation [Wilson and Cowan, 1972]:

$$\partial_t h(p,t) = -\alpha\, h(p,t) + \; s\left(\int K(p,p')h(p',t)dp' + z(p,t)\right). \tag{1.10}$$

Let us observe that, if $K(p,p')$ is of the special form $K(p-p')$, then the integral in Eq. (1.10) becomes a convolution as follows

$$\partial_t h(p,t) = -\alpha\, h(p,t) + s(K * h + z). \tag{1.11}$$

In the Wilson-Cowan equation, s is an activation function; α is a decay rate; z is the feedforward input corresponding to the response of the simple cells in presence of a visual stimulus, as in Eq. (1.3); and the kernel K represents the strength of horizontal connections between p and p'. The form of this connectivity kernel has been investigated in several studies employing differential geometry tools. A breakthrough idea in this direction has been the description of the feature space as a fiber bundle with basis $\mathbb{R}^2$ and fiber $\mathcal{F}$. This approach appeared firstly in the works of [Koenderink and van Doom, 1987] and [Ferraro and Caelli, 1994]. Then, it was further developed by [Petitot and Tondut, 1999; Petitot, 2008, 2017], where they showed that the simple cells of V1 induce a fibration of orientations. Then, in [Citti and Sarti, 2006] the model is described as a sub-Riemannian structure on the Lie group $\mathbb{R}^2 \times S^1$ by requiring the invariance under roto-translations. Other works have extended this approach by inserting other variables such as scale, curvature, velocity [see e.g. Sarti et al., 2008; Barbieri et al., 2014b; Abbasi-Sureshjani et al., 2018].

Chapter 2

Contrast perception phenomena and grouping

In this chapter we introduce some visual perception phenomena that we will study in the following chapters. We first describe the Retinex theory of Land and some recent developments in this field. Moreover, we recall the Gestalt laws and a model and the corresponding numerical approach that describes how the visual system analyzes globally a visual stimulus, finding the perceptual units or percepts, i.e. the elements in the image perceived as distinct.

2.1 Contrast perception

2.1.1 The Retinex theory of Land

The Retinex theory has been formulated by E. H. Land in [Land, 1964] to describe the phenomenon of color constancy, in particular how the perceived color of objects, varying illumination conditions, remains relatively constant. Indeed, Land realized that, even if no green or blue wavelengths are present in a visual stimulus, the visual system would still perceive them as green or blue by overlooking the red illumination.

He has demonstrated this effect performing a famous experiment. First, he

showed to a subject a display called a 'Mondrian', i.e. an image composed by several colored patches named after Piet Mondrian, whose paintings are really similar. This display is illuminated by three white lights, whose intensity can be modified by the subject, each one projected through a different filter, a red, a green and a blue one respectively. Thus, Land asked the subject to adjust the intensity of the lights so that a specific patch in the image appears white and measured the intensities of red, green, and blue lights reflected from this white-appearing patch. After this step, the subject had to identify the color of a neighboring patch, which, e.g., appeared red. Then Land adjusted the light intensities in such a way that the red patch reflected the same red, blue, and green light intensities measured from the white patch. Eventually, The subject showed color constancy: indeed, the white patch continued to appear white, the red patch continued to appear red and all the other patches continued to show their original colors.

Some years later, Land further developed his theory with J. McCann in [Land and McCann, 1971], proposing an algorithm and a computer program designed to simulate the Retinex processes taking place in the visual system. Indeed, the algorithm associated to an image $I : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ the perceived image $\widetilde{I}$.

2.1.2 The development of the Retinex model

After the work of Land, many developments and algorithms have been proposed to describe the Retinex theory [e.g. Brainard and Wandell, 1986; Provenzi et al., 2005; Lei et al., 2007, among others]. A further analysis of the retinal and cortical components causing Retinex effects has been performed by several research teams in the past years [as e.g. Valberg and Seim, 2013; Yeonan-Kim and Bertalmío, 2017]. Furthermore, it has been demonstrated in [Enroth-Cugell and Robson, 1966; Solomon et al., 2006] that the Magnocellular cells are the ones involved in contrast perception.

Some variational approaches have been proposed for example by [Kimmel et al., 2003; Morel et al., 2010; Limare et al., 2011]. In particular, in the last two works the authors have formalized the Retinex theory as the solution of the following

discrete Poisson PDE

$$-\Delta_d \widetilde{I} = M(I) \tag{2.1}$$

where Δ_d is the classical discrete Laplacian and M is a modified version of the discrete Laplacian where at each difference between the central pixel and one in proximity is applied a threshold function.

More recently, a geometrical model making a first step in relating the architecture of the visual system and the invariances of RPs has been presented in [Citti and Sarti, 2013]. The RP of the LGN cell which takes in input the visual signal acting by convolution on it, modeled as a modified discrete LoG, $M(G_\sigma) \approx \Delta G_\sigma$, where G_σ is a Gaussian bell. Thus, the action on an input image I is the following:

$$Out_{LGN}(I) = \Delta(G_\sigma * I) \approx \Delta I$$

Also the horizontal connectivity in this layer is radially symmetric and modeled as the fundamental solution $T = \log(\sqrt{x^2 + y^2})$ whose associated operator is the inverse of the Laplacian Δ^{-1} and allows to recover the function $\widetilde{I}$:

$$\widetilde{I} = T * Out_{LGN}(I).$$

As a result

$$\Delta \widetilde{I} = Out_{LGN}(I) \approx \Delta I,$$

becomes the Retinex equation. In general, $\widetilde{I}$ does not coincide with I, but differs by a harmonic function.

2.2 Perceptual grouping

2.2.1 The Gestalt laws

So far we have described only local aspects of perception but now we introduce some global perception phenomena, as e.g. the individuation of perceptual units. In particular, V1 is able to analyze more global feature of the visual stimulus thanks to the long-range connectivity and distinguishes the different percepts. Indeed, a model for the functional architecture of V1 should reproduce this mechanism.

The main principles that describe how our visual system perceives the elements in a scene are called *Gestalt laws* of grouping. Here we summarize some of them, following [Kanizsa et al., 1979]:

- *Law of Similarity* states that perception tends to group together elements that physically resemble each other as part of the same object and to recognize as distinct elements that are different. The visual system is then able to distinguish between adjacent and overlapping objects thanks to similar features;
- *Law of Proximity* states that objects or shapes that are close to one another appear to form a group. In particular, we can group together close objects even if they are different in shape and size;
- *Law of Good continuation* states that objects intersecting with each other are perceived as single uninterrupted objects. Indeed, we have a tendency to group lines and curves with similar direction over those with sharp changes of direction;
- *Law of Closure* states that the visual system tends to complete figures or forms even if they are incomplete, partially hidden or some information misses. Therefore, there exists a natural tendency to recognize patterns familiar to us and to fill missing information;
- *Law of Symmetry* states that elements symmetrical with each other are perceived as a unified group. This principle has a central role in determining figure-ground perception;
- *Law of periodicity* states that the visual system tends to detect periodicity patterns grouping them together. Therefore, if an element in the image breaks this periodicity the visual system easily distinguishes it among all the other elements.

In figure 2.1 there are some examples of Gestalt laws just listed. At this point we do not want to give a deep explanation of all Gestalt laws but only the idea behind them in order to describe which are the main tasks the visual system faces.

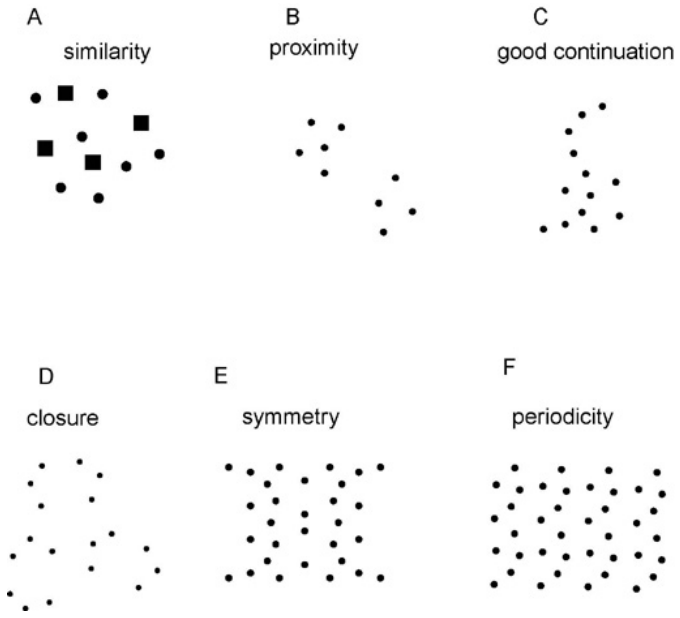


Figure 2.1 – Examples of Gestalt laws.

On the other hand, we display a famous example by Kanizsa [see Kanizsa et al., 1979]. Figure 2.2 shows the square of Kanizsa in which there are 4 black circles with a white quarter; the radii that delimit the white quarters are collinear and forms a square shape. If we look at this image we should not detect the four circles as distinct perceptual units but we should see the white square delimited by the radii of the squares. This is due to the good continuation law.

2.2.2 A neural model of perceptual units

In this section we present a neural mechanism responsible to perceptual unit constitution, and provide a model in terms of spectral analysis The most classical equation describing the cortical activity is the mean field equation (1.10) of Ermentrout and Cowan [Ermentrout and Cowan, 1980] and Bressloff and Cowan [Bressloff et al., 2002; Bressloff and Cowan, 2003]. This equation describes the evolution of the cortical activity depending on a connectivity kernel.

In the work of [Sarti and Citti, 2015] the authors find a relation between the stable states of this equation and perceptual units of the input We refer also to

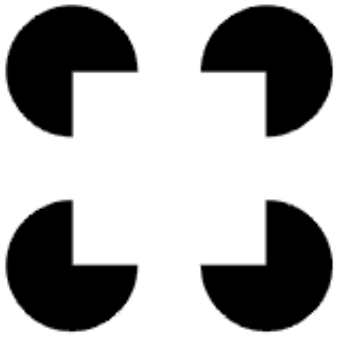


Figure 2.2 – Square of Kanizsa.

[Hoffman, 1989; Weiss, 1999; Shi and Malik, 2000; Favali et al., 2017].Let us fix the cortical space $\mathcal{G} = \mathbb{R}^2 \times \mathcal{F}$ where $p = (x, y, f)$ is a point on this space and consider the mean field eq. (1.10).

The input configurations are constituted by a finite number N of elements $p_i = (x_i, y_i, f_i)$ and their responses to a visual stimulus I is given by the function z defined in equation (1.3). We can define the set of cells activated by the image I as the set where the output of simple cells is over a fixed threshold:

$$\Omega_I = \{p_0 | z_I(p_0) > \epsilon\} \subset \mathcal{G}.$$

In this way the responses of all the cells that have a low response or a negative one are set to zero; from a neurophysiological point of view a neuron fires only if its excitation is strong enough and our model mimics this particular behaviour. Furthermore, the cells communicate with each others through the horizontal connectivity (as described in section 1.2). Indeed, between each couple of cells RPs Ψ_{p_i}, Ψ_{p_j} it can be defined a kernel $K(p_i, p_j)$, as in eq. (1.9), expressing this relation. Therefore, the cortical connectivity on an image I restricted on this set defines a neural *affinity matrix* A_I:

$$A_I(i, j) = K(p_i, p_j)I(x_i, y_i)I(x_j, y_j),\ i, j = 1, \dots N \tag{2.2}$$

In general the kernel K will have the anysotropic pattern of the considered family of cells. Kernels can be sensible to position alone, or position and orientation,

curvature or movement if the metric of simple cell is considered [Citti and Sarti, 2006; Barbieri et al., 2014b; Abbasi-Sureshjani et al., 2018]

In this discrete setting, the mean field equation (1.10) reduces to:

$$\frac{dh(i)}{dt} = -\lambda_I h(i) + \sum_{j=1}^{N} A_I(i,j)h(j) \; i = 1, \dots, N, \tag{2.3}$$

where λ_I is a physiological parameter, whose value can be modulated by the image. We can also perform the stability analysis of equation (2.3), namely we set $\frac{dh}{dt} = 0$ obtaining

$$\sum_{j=1}^{N} A_I(i,j)h(j) = \lambda_I h(i), \; i = 1, \dots, N, \tag{2.4}$$

and then we can rewrite it as the eigenvalue problem

$$A_I h^k = \lambda_I^k h^k, \tag{2.5}$$

where h^k is the k-th eigenvectors of the affinity matrix A_I associated to the eigenvalue λ_I^k.

The eigenvectors describe the stationary states of the mean field equation hence the emergent perceptual units; in this way it is possible to select them in the scene and segment it. Indeed, we can order the eigenvectors in decreasing order with respect to the corresponding eigenvalues and, in particular, the first one will correspond to the most salient object in the image.

2.2.3 Grouping with the geometric kernel

The first family of cells that can be considered is the retinal one, whose RPs are radially symmetric. Thus, the connectivity kernel associated to these cells is, in turn, radially symmetric and we can assume it is a Gaussian one defined on $\mathcal{G} = \mathbb{R}^2$:

$$K_{x_0,y_0}(x,y) = \frac{1}{4\pi t} \exp\left(-\frac{(x-x_0)^2 + (y-y_0)^2}{4t}\right) \tag{2.6}$$

centered at the point (x_0, y_0). This particular connectivity kernel has only the position and t as parameters; indeed it will correlate points that are close with each

others. The parameter t will enlarge or shrink the area of influence of the kernel. Indeed this connectivity kernel can detect objects with different sizes depending on the choice of t. We can calculate the affinity matrix between each pair of points $p_i = (x_i, y_i)$ and $p_j = (x_j, y_j)$ using the formula (2.6) for the kernel:

$$A_I(i,j) = K_{x_i,y_i}(x_j, y_j) \cdot I(x_i, y_i) \cdot I(x_j, y_j),$$

where $I(x, y)$ is the value at the position (x, y) of the image. This matrix is equivalent to the affinity matrix introduced by Perona and Freeman in [Perona and Freeman, 1998] to perform perceptual grouping. They modeled the affinity matrix in term of an heuristic distance $d(p)$ that gives high correlation to collinear and cocircular couple of elements. In particular, the affinity matrix was defined as

$$A(i,j) = e^{-\frac{d^2(p_i,p_j)}{d_0^2} - \frac{(I(p_i)-I(p_j))^2}{dI_0}}.$$

where $d_0 = 3$ and dI_0 were constant values. Note that the images are gray-scale ones represented by a $N \times N$ matrix, i.e. with N^2 pixels; indeed $0 \leq I(x, y) \leq 1$ where $I(x, y) = 0$ if the pixel is black and $I(x, y) = 1$ if the pixel is white. Since the this kernel has a Gaussian shape it will assume high values when the point p_i is close to p_j and lower values when it is far. Therefore, there is no orientation selectivity but the only criterion is the closeness of points. This is why this kernel is mainly used for grouping clouds of points as we can see in figure 2.3A where there are represented the first three eigenvectors in red, green and blue that clearly detect the 3 clouds of points of the image.

On the other hand if we try to use this kernel on an image with position-orientation elements the kernel can not find the contours. On the other hand it selects the areas with the highest number of white pixels, see the first eigenvector in red in figure 2.3B. Similar results are obtained with both cloud of points and position-orientation elements (see figure 2.3C). Therefore, the Gaussian kernel does not show the grouping properties stated by the the Gestalt law of good continuation as described in section 2.2.1. It is also well known that it poorly performs on more general images. Furthermore, the usual technique used in literature introduces different ad hoc geometry and kernels for every family of images. For example, in

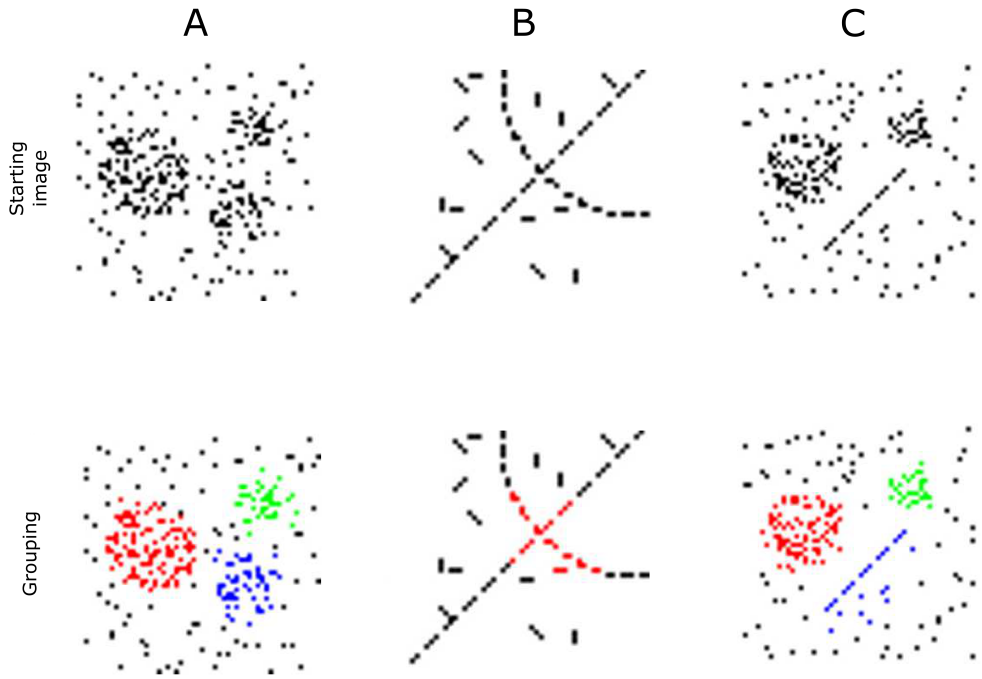


Figure 2.3 – First line: starting images. Second line: corresponding grouping with Gaussian kernel. If the first three eigenvectors are displayed they are red, green and blue respectively. Columns: **(A)** image containing three clouds of points; **(B)** image containing collinear and cocircular oriented elements; **(C)** image containing two clouds of points and a set of collinear segments.

[Barbieri et al., 2014b] the authors proposed an algorithm for the individuation of perceptual units in images with position-orientation elements through a stochastic kernel (see figure 2.4). On the other hand, in [Abbasi-Sureshjani et al., 2018] the authors modeled a kernel that was able to detect the curvature at each spatial location and then performed grouping (see figure 2.5). This necessity of introducing specific geometry and kernels for each kind of images places a problem on the grouping task.

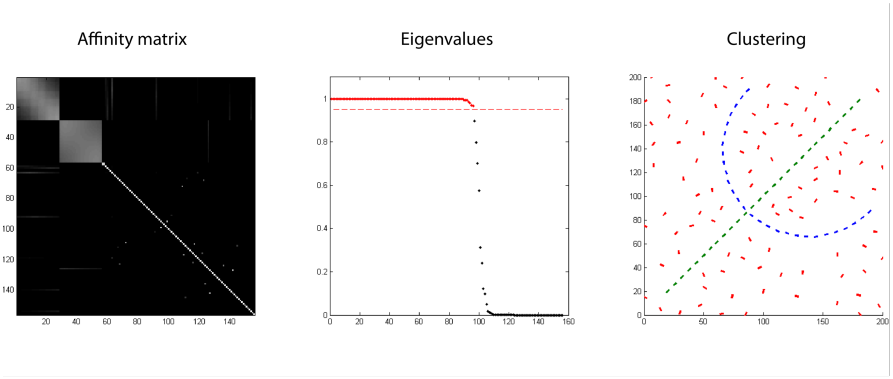


Figure 2.4 – Grouping with stochastic kernel performed on image with position-orientation elements. *From:* [Barbieri et al., 2014b].

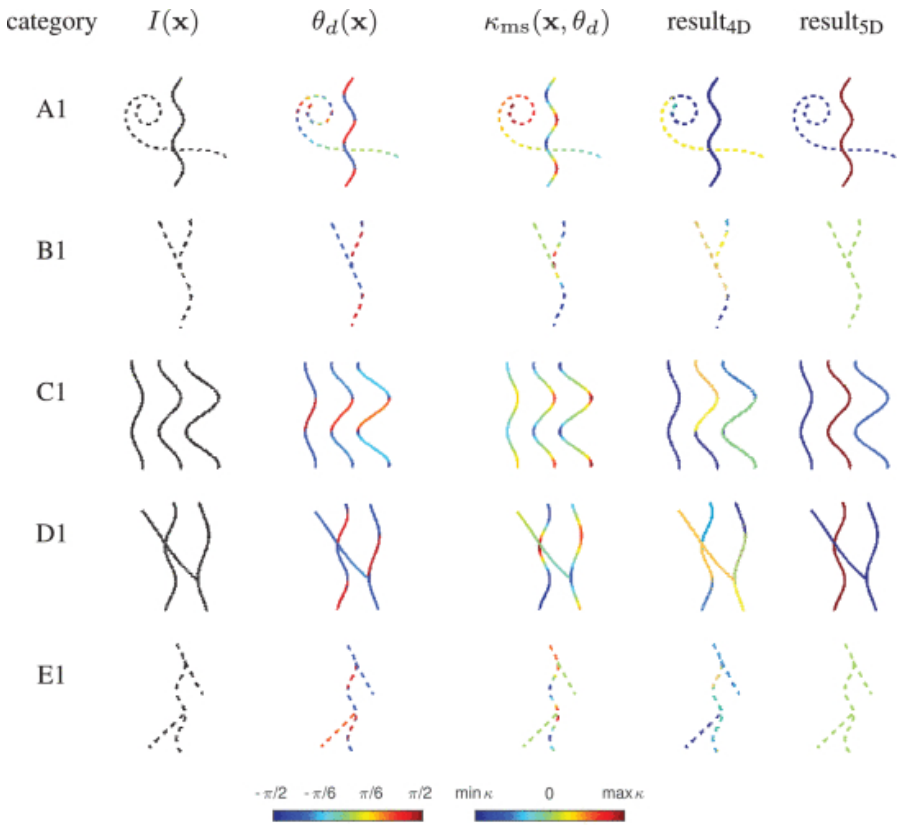


Figure 2.5 – Grouping with modeled kernel on different image. This kernel allows to recover the curvature at each spatial location. *From:* [Abbasi-Sureshjani et al., 2018].

Chapter 3

Convolutional Neural Networks

Artificial neural networks (ANNs) are computing systems inspired by biological neural networks of the brain, able to perform several tasks. The first classical model of perceptron have been introduced around 1958 by Frank Rosenblatt [see Rosenblatt, 1958] who also developed some of the main concepts of deeplearning in [Rosenblatt, 1962]. However, the first working learning algorithm for supervised, deep, feedforward, multilayer perceptrons was published in 1967 by Alexey Ivakhnenko and Lapa [Ivakhnenko et al., 1967]. Only after the introduction of CPU these models have been systematically implemented with great results on different tasks. We refer to [Girosi et al., 1995; Cucker and Smale, 2001; Szegedy et al., 2014; Higham and Higham, 2018] for a recent review. A Convolutional Neural Network (CNN) is a particular class of Artificial Neural Network (ANN) that can be used to face different tasks, usually related to visual imagery as for example object-detection [Redmon et al., 2016; Ren et al., 2017] or image classification [He et al., 2015; Simonyan and Zisserman, 2015]. For more detailed information about CNNs [see e.g. Lawrence et al., 1997; LeCun et al., 1998].

3.1 The architecture of a CNN

The CNN architecture is a machine learning procedure inspired by the visual cortices that approximates a functional $y : V^0 \to \mathcal{H}$, known only on a subset of

the domain called *training set*, through a minimization process of a *loss functional*. More precisely, a CNN is obtained by the sequentially application of a linear functional, a non linear one and a pooling operator. Then, to describe the CNN we recall:

- The structure of an artificial neuron that, as just said, has inspired the CNNs;
- The description of the layers that compose the CNN: the convolutional layer, the non linear layer and the pooling one. The last layers of a CNN are instead fully-connected;
- The description of a loss functional
- The training and test sets;
- The overfitting problem and the choice of the hyperparameters.

3.1.1 The multilayered structure

Since the whole neural network is composed by multiple layers we can express it as a composition of different functionals. If $T_{\Psi_f^i} : V^{i-1} \to V^i$ represents the i-th convolutional layer, $\Theta : V^i \to V^i$ a non linear layer, $\Pi : V^i \to V^i$ a pooling layer and $\Upsilon^1 : V^L \to \mathcal{H}$ and $\Upsilon^j : \mathcal{H} \to \mathcal{H}$, $(j = 2, \dots, k)$ the fully connected layers, then the CNN can be expressed as a functional

$$F_{\Psi_f^0,\dots,\Psi_f^L,\varphi^1,\dots,\varphi^k} : V^0 \longrightarrow \mathcal{H}$$

$$F_{\Psi_f^0,\dots,\Psi_f^L,\varphi^1,\dots,\varphi^k}(I) = \Upsilon^k \circ \cdots \Upsilon^1 \circ \Pi \circ \Theta \circ T_{\Psi_f^L} \circ \cdots \Pi \circ \Theta \circ T_{\Psi_f^0} \tag{3.1}$$

3.1.2 The artificial neuron

Considering the linear filter model (1.3) and the subsequent suppression step (1.4), the output of each neuron is given by the function

$$z(x, y, f) = \max((\Psi_f * I)(x, y) + b_f(x, y), 0), \tag{3.2}$$

where Ψ_f is the f-th filter and b_f the corresponding bias term. Indeed, an artificial neuron, which formally generalizes this structure, is composed by a linear transformation, followed by a non-linear one. Considering the i-th artificial neuron, the linear operator becomes

$$T_{\Psi_f} : V^{i-1} \to V^i \quad T_{\Psi_f}(I) = \Psi_f * I + b_f. \tag{3.3}$$

On the other hand, the non linear step becomes

$$\Theta : V^i \to V^i \quad \Theta(z)(x) = \sigma(z(x)),$$

where σ is the associated activation function. As result the model of the i-th artificial neuron will be

$$\Theta \circ T_{\Psi_f^i} : V^{i-1} \to V^i$$

The convolutional layer

The discrete formalization of the linear step is the convolutional layer which gives the name to this particular kind of ANNs. Indeed, it is a tensor of size $s \times s \times n_i \times n_o$ where s is the spatial size of the layer, n_i is the number of channels of the previous convolutional layer and n_o is the number of channels of the current convolutional layer. Fixing one channel of the last dimension of the tensor, we obtain a tensor of size $s \times s \times n_i$ called *filter*. The filter is indeed convolved with the input of the layer, allowing the structure to be equivariant to translation [see e.g. Cohen and Welling, 2016; Cohen et al., 2020]. The output of the i-th convolutional layer, following eq. (3.3) becomes

$$z_I^i(x, y, f) = T_{\Psi_f^i}(I)(x, y)$$

where z_I^i is a $[n \times n \times n_o]$ tensor, where n is the spatial dimension of the input.

The non linear layer

The non linear step can be modelled via the application of a function σ called activation function. A typical choice of this function can be the $ReLU$ (rectified

linear unit) function:

$$\sigma : \mathbb{R} \to \mathbb{R}, \quad \sigma(x) = \max(0, x). \tag{3.4}$$

Indeed, a neuron fires if the received input is large enough, otherwise remains inactive; indeed a step function is a natural model for a neuron, giving x when firing, zero when not. We can generalize the definition of the ReLU function in a multi dimensional space. Thus, considering $x \in \mathbb{R}^m$, $\sigma : \mathbb{R}^m \to \mathbb{R}^m$ is defined by applying the one dimensional ReLU to each component, i.e.

$$(\sigma(x))_i = \sigma(x_i).$$

3.1.3 The pooling operation

The pooling operation allows the CNN to decrease the spatial dimensions of the input and can be modeled as a functional

$$\Pi : V^i \to V^i.$$

Since the image I has a compact support, we can assume that the input to Π has still a compact support $\Omega \subseteq \mathbb{R}^2$ in the spatial dimensions. Then, we can split Ω in r mutually disjoint open sets $\{\Omega_j\}_{j=1,\dots,r}$ and we can choose a function τ (e.g. the max function). We should also consider a function $\varpi : V^i \to V^i$ that rescales of a parameter α the support of a given function, i.e. $\varpi(z(x, y)) = z(\alpha x, \alpha y)$. Then, the output of the pooling operator becomes

$$\Pi(z)(x, y, f) = \varpi(\tau(z(\xi, \eta, f))) \text{ , for } (\xi, \eta) \in \Omega_j \ni (x, y) \text{ , } j = 1, \dots, r$$

The pooling layer

The discrete formulation of the pooling operation is the pooling layer. A common approach is to split the spatial dimensions (width, height) in $s \times s$ boxes and shrink them to a single value applying the function τ (e.g. mean or max) to the function $z_I(x, y, f)$ on each box $B_{x,y,f}$ where $x, y = 1, \dots, n$ and are even and $f = 1, \dots, m$. In particular,

$$\Pi(z_I)(x, y, f) = \tau(z_I(\xi, \eta, f) : (\xi, \eta) \in B_{x,y,f}),$$

As a result this operator changes the size of the tensor to $[n/s \times n/s \times n_o]$. The pooling layer is not always applied after an artificial neuron and a specific choice can be done for each CNN architecture; however, if it is applied, the output becomes

$$h_I^i(x, y, f) = \Pi(\Theta(T_{\Psi_f^i}(z_I^{i-1}(x, y, f))(x, y)).$$

3.1.4 The fully-connected step

At the end of the CNN architecture, several fully-connected layers are usually applied to modify the output sizes. The fully-connected layer can be modeled as a function

$$\Upsilon^j(z)(\eta) = \int_{\mathcal{G}} \varphi^\eta(p) z(p) d\mu(p)$$

where $\mathcal{G} = V^L$ for $j = 1$ and $\mathcal{G} = \mathcal{H}$ for $j = 2, \ldots, k$. Similarly, $\varphi^\eta : V^L \to V^L$ for $j = 1$ and $\varphi^\eta : \mathcal{H} \to \mathcal{H}$ for $j = 2, \ldots, k$.

The fully-connected layer

The fully connected layer is a classical layer that connects each neuron in one layer to any neuron in another layer. The discrete formulation of the j-th one is

$$h^j(z)(\eta) = \sum_{p \in \mathcal{G}} \varphi^\eta(p) z(p)$$

for the different choice of $\mathcal{G}$. The output of each fully-connected layer becomes a tensor of size $[1 \times 1 \times M^j]$. In the case of $j = k$, i.e. the last fully-connected layer, $M^k = M$, i.e. the number of the categories of the classification task.

3.1.5 The loss functional

The CNN, modeled as a functional $F = F_{\Psi_f^0, \ldots, \Psi_f^L, \varphi^1, \ldots, \varphi^k}$, contains unknown a-priori weights, in particular in the convolutional and fully-connected layers; furthermore, it should approximate the operator $y : V^0 \longrightarrow \mathcal{H}$. The latter is a-priori known over a subset Γ of the domain space V^0, called training set. This is a finite set of N samples in V^1, $(I^i), i = 1, \ldots, N$ and the corresponding outputs

$y(I^i), \quad i = 1, \ldots, N$ in $\mathcal{H}$. Thus, we should define a loss function that estimates the error between F and y. Various loss functions, depending on the specific task, can be used. For example, in the case of a classification task, the *softmax loss function* is commonly used. This is defined as follow:

$$L(F(I), y(I)) = \log(\sum_{u=1}^{M} e^{(F_u(I) - F_{\widetilde{u}}(I))}) + F_{\widetilde{u}}(I) - F_{y(I)}(I) \tag{3.5}$$

where $\widetilde{u}$ is the label selected by the neural network among M labels and $y(I)$ is the true one. The choice of this training set and the regularity of the operator F should ensure that the operator itself remains a good approximating on the whole set of images. This extension process is called generalization.

3.1.6 The learning phase

During the learning phase all the weights in the CNN are updated in order to diminishing the loss function (3.5). This is done through some optimization algorithm, for example the steepest descent. However, the reduction of the loss function on the training set Γ does not always imply good performances on the entire set V^0. To control if the neural network is able to generalize to never seen images, it is tested during the training phase on a test set $\widetilde{\Gamma} \subseteq V^0$ disjoint with the training set, i.e. $\widetilde{\Gamma} \cap \Gamma = \emptyset$. Indeed, it is possible that the neural network performance increases on the training set but remains stable or decreases on the test set, leading to overfitting. If the performances continue to decrease on the test set is possible to adjust some parameters of the training or stop the training and selecting the CNN weights that give highest performances.

Furthermore, the architecture of a CNN has to be determined before the training phase. Modifying the number of layers, the sizes of the tensors of the convolutional layers and any other parameters of the architecture could lead to an improvement or a worsening of the performance of the architecture. All these hyperparameters are hard to determine and many trainings are usually done to find the ones that lead to the highest performance. There exists some other methods to increase performance, e.g. the *dropout* (that also avoids overfitting). At each training stage,

each node is either dropped out (i.e. ignored) with probability $1 - p$ or kept with probability p. Indeed, a different reduced CNN is left at each stage; this allows to reduce node interactions, leading them to learn more robust features and then, generalizing better to new input data.

3.2 Analogy between the visual system and CNNs

We are now interested to analyze the connections between the visual system and CNNs. The first neural nets have been inspired by a simplification of the structure of the biological visual system, presenting a hierarchical structure, where each layer receives input from the previous one and gives output to the next one. Despite such simplification, CNNs have reached optimal performances in processes typical of the human visual system, as e.g. object-detection [Redmon et al., 2016; Ren et al., 2017] or image classification [He et al., 2015; Simonyan and Zisserman, 2015]. Along with the hierarchical organization, translation invariance is present in CNNs thanks to local convolutional windows shifting over the spatial domain. This structure was first inspired by the localized RF of cells in the early visual pathways, and by the approximate translation invariance in their tuning.

In more recent years relationships between CNNs and the visual system have been further studied, with the eventual goal of making the CNNs even more efficient in specific tasks. To this end, a model of the first cortical layers described as layers of a CNN has been proposed in [Serre et al., 2007]. In [Yamins et al., 2015; Yamins and DiCarlo, 2016] the authors were also able to study higher areas by focusing on the encoding and decoding ability of the visual system. In the last years, Recurrent Neural Networks (RNNs) have been proposed to implement horizontal connections [e.g. Sherstinsky, 2020], or feedback terms [e.g. Liang and Hu, 2015]. A modification of these neural networks, more geometric and closer to the structure of the visual system, have been recently proposed in [Montobbio et al., 2019a]. In particular, they have modeled the horizontal connectivity of V1 cells, implementing a connectivity kernel between each pair of filters in a single layer that estimates their similarities.

Moreover, it is well known that both V1 RPs and the first convolutional layer of a CNN are mainly composed by Gabor filters. To this end, biological based models of V1 cells RPs in terms of Gabor filters have been proposed in [Serre et al., 2007; Zhang et al., 2019] and the statistic of the RPs of a macaque's V1 was studied in [Ringach, 2002]; however a comparison between these results and the statistics of learned filters in CNNs is still missing.

3.3 Invariance properties of CNNs

In this last section we are going to describe some invariance properties of CNNs. We first remind that Gabor functions, i.e. a model of V1 cells RPs, are mainly invariance with respect to translation and rotations. On the other hand CNNs are translation equivariant since they are defined in terms of convolutional kernels [see Cohen and Welling, 2016; Cohen et al., 2020].

Other invariance properties, strictly related to the invariances of the Gabor filters, can be imposed to optimize the training phase. In [Wu et al., 2015; Marcos et al., 2016] the authors learned just a small number of filters, and then obtained an entire bank of filters by rotation. In [Barnard and Casasent, 1991] the authors have studied invariances of neural nets with respect to specific feature spaces. Another method for introducing invariances 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 neural network is trained on a larger set of images and can learn different kinds of invariances. A way to achieve rotation invariance in classification task is to rotate every test image by S different angles as performed in [Fasel and Gatica-Perez, 2006; Dieleman et al., 2015]. In [Gens and Domingos, 2014; Dieleman et al., 2016; Laptev et al., 2016] the authors introduce one or more novel layers to face the rotation invariance problem. In [Dieleman et al., 2016] they propose four different layers that work on the input rotated by 0, 90, 180 and 270 degrees. A different kind of pooling is used in [Laptev et al., 2016] 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 [Gens and Domingos, 2014].

Chapter 4

A biologically inspired CNN architecture: LGN-CNN

In this chapter we introduce one of the novelty of our work, a CNN architecture inspired by the structure of the visual system, proposed in [Bertoni et al., 2022]. To begin with, we will introduce a first convolutional layer that mimics the role of the LGN and that eventually will approximate the LGN cells' RP, i.e. a LoG. Then, in next chapter, we will add a kernel to the second convolutional layer mimicking the horizontal connections of the V1 cells as proposed by Montobbio in [Montobbio et al., 2019a]. Finally, we will study how the filters approximate LGN and V1 cells RPs, how the kernel approximates the connectivity kernel of V1, if some Retinex effects occur and we perform grouping.

4.1 LGN in a CNN

As far as we know, the action of the LGN has not been implemented in a CNN. As we have already discussed in section 1 the LGN receives the visual stimulus from the retina and processes it through its cells. Each cell, connected to its RF, acts as a linear filter that can be modeled as a LoG. Our aim is to build a CNN architecture that takes into account this behavior; indeed, it should contain a first layer that pre-processes the visual stimulus before it reaches the second layer that,

in turn, should mimic V1. We have called this neural network architecture as Lateral Geniculate Nucleus Convolutional Neural Network (LGN-CNN).

The RPs of LGN cells can be modeled as a LoG which is a rotationally symmetric function. Thus, the first convolutional layer ℓ^0, that models the LGN, should contain just a single filter that eventually should obtain a LoG shape. On the contrary, the second convolutional layer ℓ^1, that models V1, should contain several filters that eventually should obtain Gabor shapes with different orientations. Figure 4.1 shows a scheme of the first layers of the visual pathway in comparison with the first layers of the LGN-CNN architecture.

As described in chapter 3, CNNs are architectures composed by several convolutional layers, each containing many filters. Indeed, given a classical CNN architecture, we can add before the other convolutional layers, a layer ℓ^0 composed by only one filter Ψ^0 of size $s^0 \times s^0$ and a ReLU function. In this way the first two convolutional layers ℓ^0 and ℓ^1 of this new CNN architecture model the first stages of the visual pathway. Furthermore, the introduction of ℓ^0 does not change the rest of the neural network since the input of ℓ^1 keeps the same dimensions as before. Thus, our model of LGN can be easily implemented to any CNN architecture by adding ℓ^0 and a ReLU function. Moreover, the number of parameters of the neural network is just increased by $s^0 \times s^0$.

We expect the first filter Ψ^0 to be a good model of the RP of LGN cells thank to a simple result on rotational symmetric convex functionals. Indeed, let us consider a rotational symmetric convex functional F with a unique minimum ω; then ω is also rotational symmetric. In particular, since F is rotational symmetric, $F(\omega \circ g) = F(\omega)$ for a rotation g of an angle θ. Furthermore, since the minimum is unique, $\omega = \omega \circ g$ and this implies the rotation symmetry of the solution. Many results on symmetries of minima of functionals can be found for example in [Gidas et al., 1981; Lopes, 1996]. Our purpose is to extend these theoretical results on the LGN-CNN architecture. Thus, the filter Ψ^0 should attain a rotational symmetric pattern and should eventually approximate a LoG if the LGN-CNN architecture we have proposed is a good model of the first stages of the visual pathways.

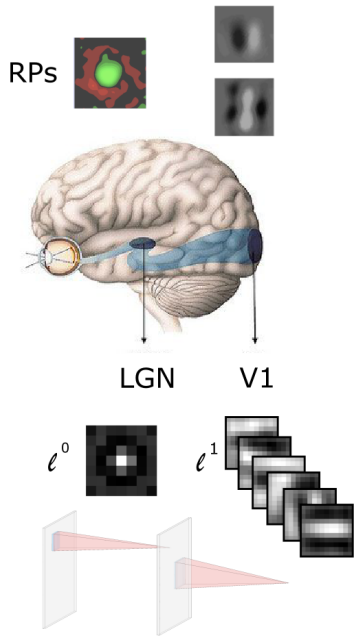


Figure 4.1 – Scheme of the LGN and V1 in parallel with the first two layers ℓ^0 and ℓ^1 of the LGN-CNN architecture.

4.2 The LGN-CNN architecture

In this section we are going to introduce the first LGN-CNN architecture and the settings in which we have trained and analyzed it. For this architecture we have used MatLab2019b for academic use.

Let us start by describing the neural network architecture shown in figure 4.2. The LGN-CNN acts on an input image of size $64 \times 64 \times 1$ through the application of 5 different convolutional layers that modify the dimensions of the tensor. In light blue it is shown one filter of a given convolutional layer and in red its corresponding application in a specific spatial coordinate. The last three vectors are three fully-connected layers.

More in details, the first convolutional layer ℓ^0 is composed by a single filter Ψ^0 of size 7×7, followed by a batch normalization layer b^1 and by a ReLU R. Thus, the second convolutional layer ℓ^1 receives as input a tensor of size $64 \times 64 \times 1$. We do not insert a pooling layer after the first layer ℓ^0, i.e. this layer does not change

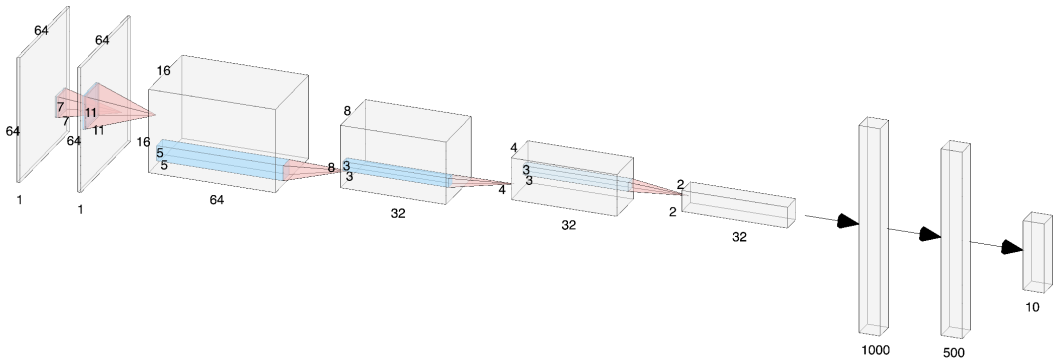


Figure 4.2 – Architecture of LGN-CNN.

the dimension of the input image.

The second convolutional layer ℓ^1 is composed by 64 filters of size 11×11 and its stride is 2, halving the spatial dimensions. After ℓ^1 it is applied a batch normalization layer b^{64}, a ReLU R and a max POOLING p_m^2 with squares of size 2×2. Then, the third convolutional layer ℓ^2, composed by 32 filters of size $5 \times 5 \times 64$, is followed by a batch normalization layer b^{32}, a ReLU function R and a max POOLING p_m^2. Thus, the fourth convolutional layer ℓ^3 is composed by 32 filters of size $3 \times 3 \times 32$ followed by a batch normalization layer b^{32}, a ReLU function R and a max POOLING p_m^2. Finally, the last convolutional layer ℓ^4 has 32 filters of size $3 \times 3 \times 32$ followed by a batch normalization layer b^{32}, a ReLU function R and a max POOLING p_m^2.

Eventually three fully-connected (FC^1, FC^2, FC^3) layers of size 1000, 500 and 10 respectively are applied giving as output a vector of length 10. As a last step, a softmax σ is applied to the output vector of size 10 giving a probability distribution over the 10 classes. The functional that models this neural network is the following

$$\begin{aligned} F(I) :=& (\sigma \circ FC^3 \circ FC^2 \circ FC^1 \circ \\ & p_m^2 \circ R \circ b^{32} \circ \ell^4 \circ p_m^2 \circ R \circ \\ & b^{32} \circ \ell^3 \circ p_m^2 \circ R \circ b^{32} \circ \ell^2 \circ \\ & p_m^2 \circ R \circ b^{64} \circ \ell^1 \circ R \circ b^1 \circ \ell^0)(I) \end{aligned} \tag{4.1}$$

A cross-entropy loss for softmax function defined as in equation (4.2) is applied

to the functional (4.1) where $\widetilde{u}$ is the label selected by the neural network and $y(I)$ is the true label.

$$L(F(I), y(I)) = \log(\sum_{u} e^{(F_u(I) - F_{\widetilde{u}}(I))}) + F_{\widetilde{u}}(I) - F_{y(I)} \tag{4.2}$$

As just described, after each convolutional layer there is a batch normalization layer b with the same size of the number of filters of the corresponding convolutional layer. In particular, the batch normalization layer b^1 after ℓ^0 performs a normalization similar to the one that the retina performs as described in [Carandini and Heeger, 2011]. One of the main difference is the subtraction of the mean value μ performed by the batch normalization layer defined as follows

$$\widehat{x_i} = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}},$$

where x_i is the element to normalize, μ is the mean value of the batch, σ is the standard deviation of the batch and ϵ is a small value that prevents bad normalization in case of small standard deviation. However, the input images have zero mean which imply that the convolution with Ψ^0 have still zero mean. Thus, the batch normalization layer between ℓ^0 and ℓ^1 has similar characteristics as the biological one. In spite of this, the two approaches differ since the statistical parameters (μ, σ) of b^1 are calculated channel-wise over all input of the batch, while in the biological normalization described in [Carandini and Heeger, 2011] the σ value is calculated from a single input instance over a restricted spatial neighborhood. However, we expect that the final result will not differ heavily applying the batch normalization layer.

We trained our LGN-CNN architecture on a dataset of natural images called STL-10 [see Coates et al., 2011] that contains 5000 training images divided in 10 different classes. The steps we have followed for modifying the training set are:

- Changing the images from RGB color to grayscale color using the built-in function *rgb2gray* of MatLab;
- Applying a circular mask to each image, leaving unchanged the image in the circle and putting the external value to zero;

LGN + 4CL	LGN + 8CL	LGN + 12CL
70.41%	72.73 %	73.61 %

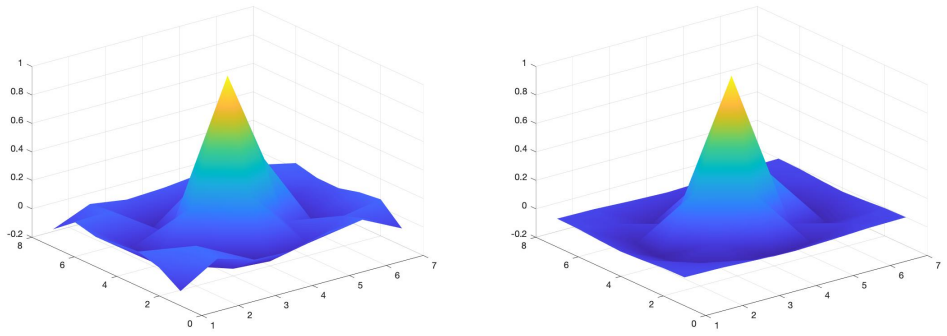
Table 4.1 – Mean performances of several architectures over 10 different trainings with LGN layer plus 4, 8 and 12 other convolutional layers (CL).

- Rotating each image by 5 random angles augmenting the dataset to 25000 images; thanks to the previous step no boundary effects are present;
- Cropping the 64×64 centered square that does not contain the black boundaries;
- Subtraction of the mean value in order to have zero mean input images.

Thus, after these steps we have obtained a rotation invariant training set composed by 25000 64×64 images. We have applied the same steps to the test set but we have rotated each image to just one random angle. Since the images are 64×64 we have decided to use quite large filters in the first and second layer (7×7 and 11×11 respectively) in order to obtain more information about their shapes.

We have trained the neural network for 30 epochs with an initial learning rate of 0.01, a learning rate drop factor of 0.97 and a piecewise learning rate schedule with a learning rate drop period of 1. The mini batch size is 128 with an every-epoch shuffle, the L^2 regularization term is 0.02 and the momentum is 0.9.

In Table 4.1 there are summarized the mean performances over 10 different trainings of three CNN architectures with the LGN layer ℓ^0 plus several convolutional layers (4, 8 and 12 respectively). As expected, the performance increases when the architecture is deeper. We stress out that the results obtained on the LGN layer and the first convolutional layer are not affected from the rest of the neural network. Indeed, it is possible to add more layers and build more deeper architectures to obtain better results on the classification task without losing the properties of the first layers of the LGN-CNN. Since our focus was to analyze the structures that arises in the first two layers, we have not further investigated the performance of the CNN.



(a) *Filter Ψ^0 of first layer of LGN-CNN.* (b) *Minus the Laplacian of Gaussian (LoG).*

Figure 4.3 – Comparison between the filter Ψ^0 and minus the LoG. The two have a high correlation of 95.21% computed using the built-in function of MatLab *corr2*.

4.3 The filters of the LGN-CNN architecture

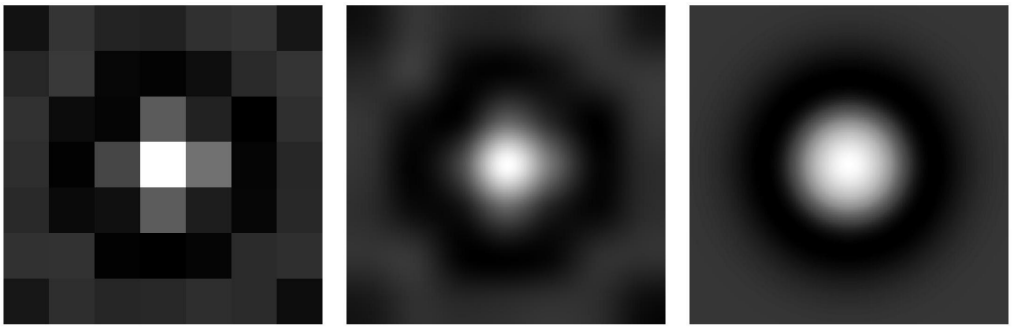
4.3.1 The layer ℓ^0 of LGN-CNN

In this section we focus on the filter Ψ^0 of the layer ℓ^0. Indeed, we are interested in analyzing its symmetry properties.

Figure 4.3 shows the 3D filter Ψ^0 obtained after the training phase. Then, figure 4.3b shows its approximation with minus the LoG.The two have a high correlation of 95.21% computed using the built-in function of MatLab *corr2*. Thus, Ψ^0 spontaneously attains a LoG shape, modeling the RPs of LGN cells.

Moreover, figure 4.4a shows the 2D image of Ψ^0 obtained after the training phase. Then, in figures 4.4b and 4.4c we plot a $2D$ approximation of Ψ^0 as a 280×280 filter and minus the LoG which shows the rotational symmetric pattern of Ψ^0.

So far we have modeled the RPs of LGN cells with a single filter Ψ^0. However it is well known that in the LGN there exists both cells with on-center/off-surround as well as off-center/on-surround RPs. In order to model these two kinds of cells we modify the current LGN-CNN architecture by adding a second filter in the layer ℓ^0. This is not the case and figure 4.5 shows the on-center/off-surround and



(a) *Filter Ψ^0 of first layer of LGN-CNN.*

(b) *Approximation of the filter Ψ^0 of first layer of LGN-CNN.*

(c) *Minus the Laplacian of Gaussian (LoG).*

Figure 4.4 – The first figure shows the 7×7 filter Ψ^0 of the LGN-CNN. To better visualize the filter Ψ^0, we provide an approximating 280×280 filter and minus the LoG.

off-center/on-surround obtained after the training phase. However, for simplicity, we consider the model with a single filter.

Since we are interested in the rotational properties of Ψ^0, we aim to quantify them. To this end, we can generate a new filter Ψ^0_S from Ψ^0 via a rotation invariance symmetrization. In particular, the normalized rotational invariant filter Ψ^0_S is obtained by the following procedure:

- Using the function *imresize* with scale 3 and bilinear method to enlarge the

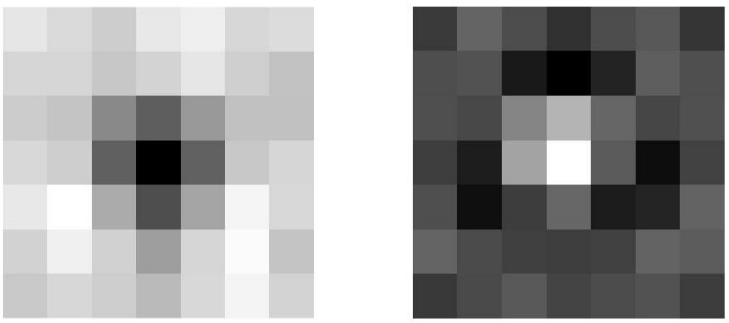


Figure 4.5 – On-center/off-surround and off-center/on-surround filters of ℓ^0 with 2 filters.

filter;

- Rotating the filter with *imrotate* by 360 discrete angles between 1 and 360 degrees and summing them up;
- Applying again the function *imresize* with scale 1/3 and nearest method to recover a filter with the same size of Ψ^0;
- Normalizing the filter by subtracting the mean and dividing it by L^2 norm.

Then, we compute the correlation between Ψ^0_S and the normalized Ψ^0 using the Matlab function *corr2* in order to quantify the rotational invariance of Ψ^0. Indeed, the computed correlation estimates how close the normalized filter Ψ^0 is w.r.t its corresponding normalized rotational invariant filter.

Thank to this tool we are now able to test the behavior of the first filter Ψ^0 of LGN-CNN for different values of the L^2 regularization term and adding more convolutional layers. In particular, we add to the previous architecture described in chapter 4.2 two convolutional layers composed by 32 filters of size $3 \times 3 \times 32$ (each one followed by a batch normalization layer b and ReLU R) after the third layer ℓ^3. We also add other two convolutional layers with the same characteristics after the layer ℓ^4.

L^2 term	LGN + 4 layers	LGN + 8 layers
0.01	88.3 %	85.40 %
0.02	97.15 %	90.79 %
0.03	93.06 %	91.41 %
0.04	95.61%	92.58 %
0.045	93.55 %	94.79 %
0.05	95.44 %	94.11 %

Table 4.2 – Correlation between Ψ^0 and Ψ^0_S varying the L^2 regularization term and the number of layers of the LGN-CNN architecture. Best rotational symmetric filters are selected in cyan for both architectures.

	No DA	Mild DA	Hard DA
LoG corr	93.77 %	93.68 %	95.21 %
Ψ^0_S corr	93.92 %	94.16 %	97.15 %

Table 4.3 – Correlation between Ψ^0 and its LoG approximation and between Ψ^0 and Ψ^0_S varying the data augmentation (DA) applied to the dataset.

In Table 4.2 we report the correlations with different values of L^2 regularization term and for two LGN-CNN architectures, the one introduced in section 4.2 which is indicated by 'LGN + 4 layers' and the deeper one described above indicated by 'LGN + 8 layers'. All the other training parameters are provided in section 4.2. As we can see for 'LGN + 4 layers' architecture the L^2 regularization term that let Ψ^0 to be the closest rotational symmetric filter is 0.02. This is the filter we have shown in figures 4.3 and 4.4. On the other hand, in the case of 'LGN + 8 layers' architecture the more rotationally symmetric filter is the one with 0.045 as the L^2 regularization term. This behavior suggests that the architecture of LGN-CNN influences directly the properties of the filters. Furthermore, the Table 4.2 shows that the rotational symmetry of Ψ^0 is stable with respect to variations of L^2 regularization term and adding convolutional layers.

Furthermore, we study the properties of Ψ^0 varying the data augmentation (DA) applied to the dataset. Table 4.3 shows the correlation of Ψ^0 with the LoG and the correlation of Ψ^0 with Ψ^0_S with three different DA applied to the dataset. In the first one we do not apply any DA (No DA); in the second one we randomly rotate the images of an angle of 0, $\frac{\pi}{2}$, π, $\frac{3\pi}{2}$ radiant (Mild DA); in the third one we apply the DA described in section 4.2 (Hard DA). From Table 4.3 it emerges that the introduction of rotation invariances in the dataset by rotating the images lightly affects the correlation with the LoG but gives stability to Ψ^0 allowing the filter to be more rotational symmetric.

In conclusion, thanks to the analysis performed on the properties of Ψ^0, we can argue that the structure of the architecture itself influences the shape of the filters and that the introduction of ℓ^0 with a single filter Ψ^0 is a good model of the

LGN. In the next section we are going to prove that Ψ^0 is rotational invariant for a specific architecture whereas in chapter 6 we will study the Retinex effects of Ψ^0.

4.3.2 Rotation invariance proof of Ψ^0

We first define the setting in which we study the rotation symmetry of Ψ^0. Let us consider the architecture of an LGN-CNN in which we can split the first convolutional layer composed by only one filter from the rest of the neural network which will be fixed. Thus, this first layer can be approximated by a function $\Psi^0 : \mathbb{R}^2 \to \mathbb{R}$, assuming $\Psi^0 \in L^1(\mathbb{R}^2)$. A general image is defined as a function $I : \mathbb{R}^2 \to \mathbb{R}$ where we assume $I \in L^1(\mathbb{R}^2)$. We consider a subset Γ of all the images where, for each image I, is defined a labelling $y : \Gamma \to \mathbb{R}$, where $y(I)$ is the corresponding label to the image I.

We require some rotational invariant properties on this set Γ. In particular, let us consider a rotation R_θ of an angle θ on R^2 plane around its center; then, the composition $I_\theta = R_\theta(I) = I(R_{-\theta}(x))$ is still an image and we can also assume that $I_\theta \in \Gamma$ (i.e., that the subset Γ is close under rotation). Furthermore, the rotated image should maintain the same label, i.e., $y(I_\theta) = y(I)$. Since the images we consider in this problem are gray-scale ones (thus with values between 0 and 1) we can also assume that the images are normalized with $||I||_{L^1} \leq 1$. Summarizing all these properties we assume that $\Gamma = \{I \in L^1(\mathbb{R}^2) : ||I||_{L^1} \leq 1\}$ Thus, the rest of the neural network is defined by a convex regularization C of the ReLU function σ defined in eq. (3.4)

$$C : \mathbb{R}^2 \to \mathbb{R}^2, \quad \text{s.t.} \quad ||C(x) - \sigma(x)||_{L^1} < \epsilon \tag{4.3}$$

for a small enough $\epsilon > 0$. And then we can define

$$\begin{aligned} &F : L^1(\mathbb{R}^2) \times L^1(\mathbb{R}^2) \to \mathbb{R} \\ &F(I, \Psi^0) := \int_{\mathbb{R}^2} C\left((I * \Psi^0)(z)\right) dz. \end{aligned} \tag{4.4}$$

Then $F(I, \Psi^0)$ is the label that the LGN-CNN architecture associates to the image I and should eventually approximates $y(I)$.

Thus, our aim is to minimize the following functional

$$\min_{\Psi^0 \in L^1(\widetilde{\Omega})} \int_\Gamma |F(I, \Psi^0) - y(I)| d\mu(I) \tag{4.5}$$

where the integral done over the set Γ is a Bochner's integral (see e.g., section 5 of chapter V of [Mikusiński, 1978; Yosida, 1995].

Thus, the function Ψ^0 attains the minimum of (4.5) where F is defined in (4.4) and C is defined in (4.3). We aim to find out if there exist any rotational invariant properties on the function Ψ^0. Indeed, the functional defined in (4.5) is convex thanks to the convexity of the function C defined in (4.3) and continuous. Then, the existence and uniqueness of a solution is guaranteed [see e.g., section 1.4 of Brezis, 2010].

Remark 1. Let us consider two function $f, g \in L^1(\mathbb{R}^2)$ and a rotation R_θ of an angle θ. Then

$$f * R_\theta(g)(x) = (R_{-\theta}(f) * g)(R_{-\theta}(x))$$

Proof.

$$\begin{aligned} f * R_\theta(g)(x) &= \int_{\mathbb{R}^2} f(x-y) R_\theta(g(y)) dy \\ &= \int_{\mathbb{R}^2} f(x-y) g(R_{-\theta}(y)) dy = \end{aligned}$$

then we substitute $y' = R_{-\theta}(y)$ whose Jacobian has determinant equal to 1

$$\begin{aligned} &= \int_{\mathbb{R}^2} f(x - R_\theta(y')) g(y') dy' \\ &= \int_{\mathbb{R}^2} f(R_\theta(R_{-\theta}(x)) - R_\theta(y')) g(y') dy' \\ &= \int_{\mathbb{R}^2} f(R_\theta(R_{-\theta}(x) - y')) g(y') dy' \\ &= \int_{\mathbb{R}^2} R_{-\theta}(f(R_{-\theta}(x) - y')) g(y') dy' = \end{aligned}$$

then we substitute $y'' = R_{-\theta}(x) - y'$ whose Jacobian has determinant equal to 1

$$\begin{aligned} &= \int_{\mathbb{R}^2} R_{-\theta}(f(y'')) g(R_{-\theta}(x) - y'') dy'' \\ &= (R_{-\theta}(f) * g)(R_{-\theta}(x)) \end{aligned}$$

and this concludes the proof. □

Thanks to this Remark we are now able to demonstrate the rotational invariance of Ψ^0.

Theorem 4.3.1. *Let Ψ^0 be a solution to the problem (4.5) where F is defined in (4.4) and C is defined in (4.3). Then Ψ^0 is rotational invariant.*

Proof. Let us consider the rotated solution $R_\theta(\Psi^0)$ of an angle $\theta \in [0, 2\pi]$.

$$\int_\Gamma | \int_{\mathbb{R}^2} C\left((I * R_\theta(\Psi^0))(z)\right) dz - y(I)| d\mu(I) =$$

and because of remark 1

$$= \int_\Gamma | \int_{\mathbb{R}^2} C\left((R_{-\theta}(I) * \Psi^0)(R_{-\theta}(z))\right) dz - y(I)| d\mu(I)$$

Since for a general $f \in L^1(\mathbb{R}^2)$ it holds

$$\int_{\mathbb{R}^2} f(R_\theta(x)) dx = \int_{\mathbb{R}^2} f(x) dx,$$

then

$$= \int_\Gamma | \int_{\mathbb{R}^2} C\left((R_{-\theta}(I) * \Psi^0)(z)\right) dz - y(I)| d\mu(I) =$$

Finally, since Γ is close under rotation and $y(I) = y(R_{-\theta}(I))$ by hypothesis

$$= \int_\Gamma | \int_{\mathbb{R}^2} C\left(((I) * \Psi^0)(z)\right) dz - y(I)| d\mu(I)$$

Thus, $R_\theta(\Psi^0)$ attains the same value of Ψ^0 for every choice of θ. But thanks to the uniqueness of the solution of this problem because of the compactness of Γ and convexity of the functional (4.5), $\Psi^0 = R_\theta(\Psi^0)\ \forall \theta \in [0, 2\pi]$ and this concludes the proof. □

Remark 2. Let us note that the first part of the proof of theorem 4.3.1 is valid for a general LGN-CNN architecture. In particular, if we have a solution Ψ^0 to the minimization problem (4.5), then every rotation $R_\theta(\Psi^0)$ of an angle $\theta \in [0, 2\pi]$ is still a solution. The uniqueness of solution in theorem 4.3.1 guarantees that Ψ^0 is rotational invariant whereas for a general LGN-CNN this does not hold.

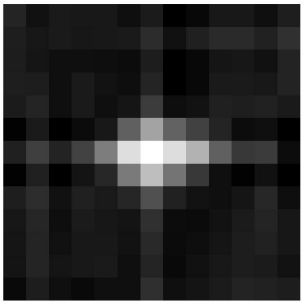


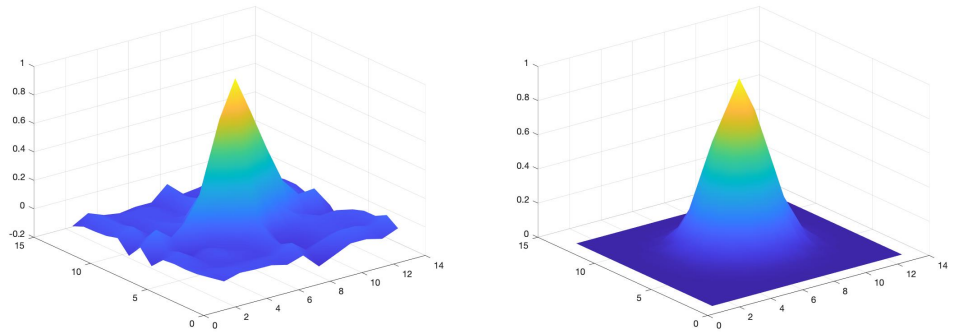
Figure 4.6 – Filter Ψ^0 of LGN-CNN obtains after training.

Testing the theorem on the LGN-CNN architecture

We aim now to test the theorem just proved on the LGN-CNN architecture used in the theorem itself. In particular, we aim to see if ℓ^0 becomes rotational invariant; however, since we train a simple architecture composed by a single convolutional layer ℓ^0 with a single filter Ψ^0 and a ReLU we do not expect to obtain a LoG shape filter as for the LGN-CNN defined in chapter 4.2. Indeed, in the previous case the LGN-CNN architecture has other convolutional layers whose purpose is to further analyze the image and in particular the contours of the objects. For this reason we expect in that case that the LGN-CNN behave similarly to the LGN and the V1, i.e., Ψ^0 has a LoG shape. On the other hand, in the test we perform now we can only expect a rotational invariant filter as stated in theorem 4.3.1.

To perform this test we build a new dataset of images starting from the dataset MNIST [a set of digits images, see LeCun and Cortes, 2010] and the dataset Fashion-MNIST [a set of cloths images, see Xiao et al., 2017]. They are two similar datasets, composed by gray-scale images of size 28×28. The purpose of our LGN-CNN is to classify the input as a digit or a cloth, indeed if the image belongs to MNIST or Fashion-MNIST dataset. The new training set is built by taking the first half of MNIST dataset (30000 images) and the first half of Fashion-MNIST dataset (30000 images), for a total of 60000 images. We follow the same steps for the test set (for a total of 10000 images) and we randomly sort the training and test sets. Then each image is labeled by 0 if it is a digit and by 1 if it is a cloth.

The LGN-CNN is a really simple architecture that contains only a first layer



(a) *Filter Ψ^0 of first layer of LGN-CNN.* (b) *Gaussian function.*

Figure 4.7 – Comparison between the filter Ψ^0 and a Gaussian. We can see that Ψ^0 is really close to a discrete approximation of a Gaussian fitted to the data.

with a single filter Ψ^0 of size 13×13 followed by a ReLU and by a fully connected layer. We train this neural network for a total of 25 epochs obtaining an accuracy of 98.55 % on the classification task. Figure 4.6 shows Ψ^0 of this LGN-CNN architecture. We can observe that it has a rotational invariant shape as we expect from theorem 4.3.1. Then, we approximate the filter Ψ^0 with a Gaussian function defined by the following formula $G(x, y) = \alpha e^{\frac{-x^2-y^2}{2\sigma^2}}$. Figure 4.7 shows Ψ^0 and its approximation. The rotation invariance of Ψ^0 is now enforced thanks to the approximation obtained with a rotational invariant function as the Gaussian.

4.3.3 The layer ℓ^1 of the LGN-CNN

Now we aim to study the second layer ℓ^1 of the architecture. As described in chapter 3, the first layer of a classical CNN contains filters approximated by Gabor functions (1.2). Thus, to enforce the relation between our architecture and the structure of the visual system, we compare the filters in ℓ^1 with the statistics of RPs shape from [Ringach, 2002]. To this end, we train two different CNNs, an LGN-CNN defined by the functional (4.6)

$$F(I) := (\sigma \circ FC^1 \circ R \circ \ell^3 \circ p_a^4 \circ R \circ \ell^2 \circ p_m^4 \circ R \circ \ell^1 \circ R \circ \ell^0)(I) \tag{4.6}$$

and a classical CNN defined by the functional (4.7) in which we eliminate

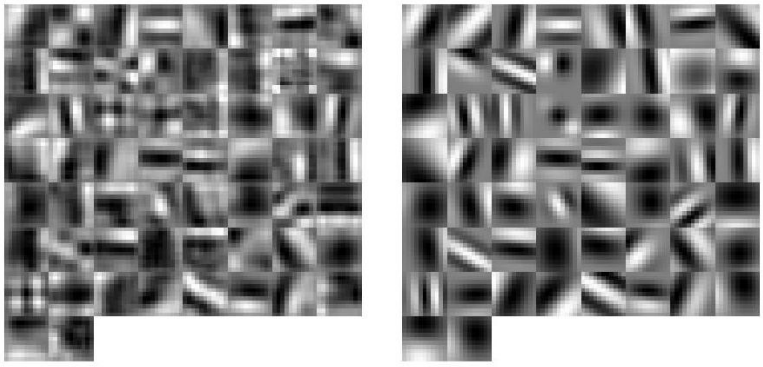


Figure 4.8 – On the left: filters from LGN-CNN. On the right: their approximation with the function (1.2).

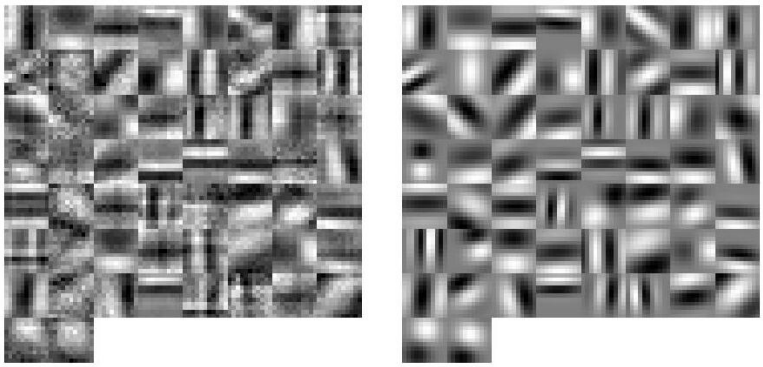
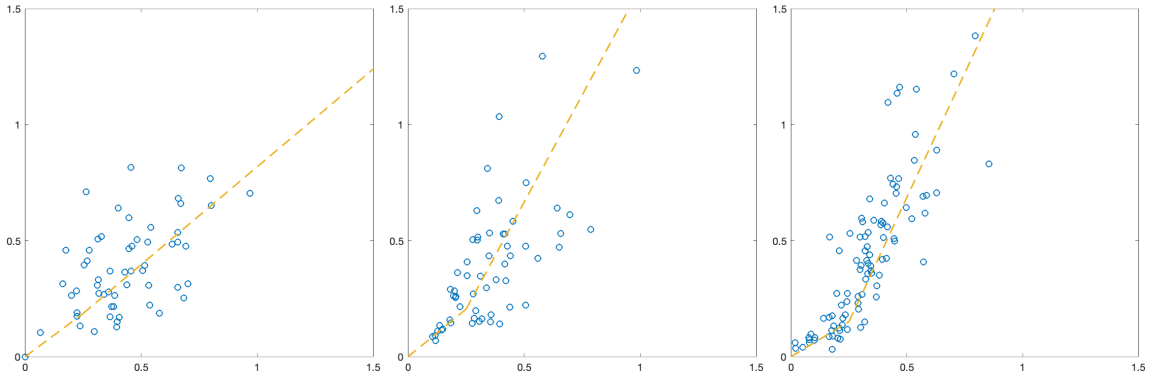


Figure 4.9 – On the left: filters from classical CNN. On the right: their approximation with the function (1.2).

the first convolutional layer ℓ^0 and its following ReLU R, characteristic of our architecture.

$$F(I) := (\sigma \circ FC^1 \circ R \circ \ell^3 \circ p_a^4 \circ R \circ \ell^2 \circ p_m^4 \circ R \circ \ell^1)(I) \tag{4.7}$$

As expected, in both architectures ℓ^1 contains filters with Gabor-like shapes after the training phase. Thus, we study the statistical distribution of these banks of filters comparing the results with the real data of Ringach. We aim to see if the introduction of ℓ^0 influences in any way the filters of ℓ^1. Since the filters in a standard CNN have input on the retina, we will construct analogous filters in the



(a) *Distribution of filters in ℓ^1 of classical CNN defined by functional* (4.7).

(b) *Distribution of filters in ℓ^1 of LGN-CNN defined by functional* (4.6).

(c) *Distribution of RPs of simple cells,* from: *[Ringach, 2002].*

Figure 4.10 – Comparison between the statistical distribution on (n_x, n_y) plane of filters of a classical CNN, of the LGN-CNN architecture and of RPs of real data.

LGN-CNN architecture. Consequently we do not consider the filters in ℓ^1 but the filters obtained by the convolution with Ψ^0.

As a first step, we need to approximate the filters in ℓ^1 using the function (1.2); in this way we are able to obtain for each filter the parameters defined in eq. (1.5) that Ringach has used for the statistical comparison as described in chapter 1. Figure 4.8 shows some of the ℓ^1 filters of the LGN-CNN and their approximation; the same occurs in figure 4.9 in the case of classical CNN. As a first result, the mean correlation estimated with the built-in MatLab function *corr2* increases from classical CNN to LGN-CNN from just 71.62% to 93.50%. This suggests that introducing the layer ℓ^0 with a single filter better regularize the filters in the following convolutional layer ℓ^1.

To compare the statistical distributions of the ℓ^1 filters of the classical CNN and the LGN-CNN with the real data from Ringach we follow the same step in [Ringach, 2002], described in chapter 1. Indeed, we plot the distribution in the (n_x, n_y) plane and approximate the data with two lines in the following way. We first approximate the points closer to the origin with a line $y = \alpha x$ and then the rest of the points are approximated with a line starting from the end of the previous

one.

Figure 4.10 shows the three plots. As a first result, the introduction of ℓ^0 modifies the elongation of Gabor filters in ℓ^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 4.10a. In figure 4.10b we can see that the slope changes greatly and that the filters become much more elongated in the y direction. This behavior is similar to the RPs of simple cells in V1 as can be observed from the distribution of Ringach in figure 4.10c. Indeed, the slopes of the interpolating lines of Ringach distribution and of LGN-CNN ℓ^1 distribution are quite close. This enforces even more the link between the LGN-CNN and the structure of the visual system motivating us to pursue in this direction.

Chapter 5

The LGN-CNN architecture with the transition kernel

5.1 Horizontal connectivity of V1 in a CNN

Even if the analogy with biological vision is strong, CNN architectures contains a simplified feedforward mechanism that ignores many of the main processes allowing the visual system to interpret a visual scene. Thus, we insert a mechanism of horizontal propagation first proposed in [Montobbio et al., 2019a]. In particular, this mechanism is defined by entirely learned connectivity kernels which will be analyzed in terms of invariance patterns we expect to arise as a result of learning from natural images. The connectivity kernel will be implemented only in the layer ℓ^1 and it should express long range connections between the learned filters of ℓ^1.

Some implementations of horizontal connections of convolutional type have been already proposed by [Liang and Hu, 2015; Spoerer et al., 2017]. In particular, they have applied a recurrent formula analogous to (1.10), describing an evolution in time. However, the horizontal kernels employed in these works are very localized, so that the lateral kernels applied at each step only capture the connections between neurons very close-by in space.

More recently, Montobbio in [Montobbio et al., 2019a] has proposed a two-step version of this model that allows the kernel to express long range connections

between neurons. In the first step, the output of the first layer ℓ^1 is mapped to the corresponding feature space through the feedforward stream, leading to an activation pattern $h^1 = ReLU(z^1)$; then it is convolved with a connectivity kernel K^1 with a wider support. In the second step, the new output $\tilde{h}^1$ is obtained by averaging between this propagated activation $K^1 * h^1$ and the original activation h^1. Indeed, the update rule reads:

$$\tilde{h}^1 = \frac{1}{2}\Big(K^1 * h^1 \; + \; h^1\Big). \tag{5.1}$$

We stress out that the entire CNN architecture is first trained over a classification task, with the exception of the lateral kernel K^1. In a second step, the rest of the architecture is kept fixed and only the connectivity kernel K^1 is updated. From a physiological point of view it means that in an embryo first there is the formation of filters, and then the connectivity between them. This is proved by experiments which show that there is no lateral connectivity in animal who do not seen alignments in the beginning of their life. Thanks to this procedure it is possible to easily add the lateral kernel to a pretrained CNN without any strong modifications. Furthermore, this two-step training allows the filters of ℓ^1 to be trained freely without any constraints from K^1. In fact, since at the beginning of the first training all the filters are initialized randomly, training K^1 simultaneously with the filters of ℓ^1 could force them to attain particular shapes. However, our purpose is to extract long range information from the bank of filters without imposing any constraints on them.

Let us describe in more details the connectivity kernel K^1. K^1 is a 4-dimensional tensor parameterized by 2-D spatial coordinates (i, j) and by the indices (f, g) corresponding to all pairs of ℓ^1 filters. Indeed, for fixed f and g filters, the function

$$(i, j) \mapsto K^1(i, j, f, g) \tag{5.2}$$

expresses the strength of connectivity between the filters Ψ_f and Ψ_g, where the spatial coordinates (i, j) indicate the displacement in space between the two filters. In order to express all possible displacements between filters, the spatial dimensions of K^1 are almost doubled with respect to the filters ones. The idea

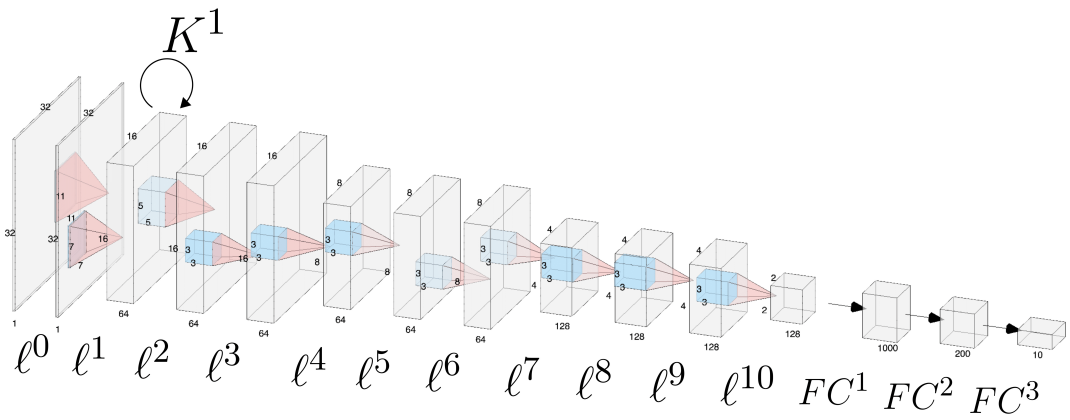


Figure 5.1 – Scheme of the CNN architecture. Convolutional layers are numbered from ℓ^0 to ℓ^{10}, fully connected classification layers are denoted as FC^1, FC^2, FC^3. Horizontal connections governed by the connectivity kernel K^1 are represented as an operator acting on the feature space associated with layer ℓ^1.

behind this model is that K^1 behaves like a 'transition kernel' on the feature space of ℓ^1 and modifies the feedforward output according to the learned connectivity between filters. Thus, similarly to the horizontal connectivity of cells in V1, a filter activation increases the activation of the filters strongly related with it.

Let us also observe that K^1 takes the form of a linear propagation applied before the nonlinear activation function of ℓ^1. Indeed, an even wider connectivity may be modelled by iterating the process, obtaining a larger kernel in spatial dimensions. This can be obtained via 'self-replication' by convolving K^1 against itself in the following way:

$$\frac{1}{2}\Big(K^1 * \tilde{h}^1 \; + \; \tilde{h}^1\Big) = \frac{1}{4}\Big(K^1 * K^1 * h^1 \; + \; 2K^1 * h^1 \; + \; h^1\Big).$$

5.2 The proposed architecture

In this section we give a detailed overview of the LGN-CNN architecture with a transition kernel in the second convolutional layer,proposed in [Bertoni et al., 2021], as well as the data and training scheme employed.

The architecture is composed by 11 convolutional layers, each one followed by a ReLU function and a batch normalization layer; at the end of the architecture 3 fully-connected layers are present. Appropriate zero padding are applied in order to keep the spatial dimensions unchanged after each convolutional layer. Figure 5.1 shows the network architecture whose layers are listed below. Convolutional layers are denoted by $\ell^0, \dots, \ell^{10}$, whereas fully connected layers are denoted by FC^1, FC^2, FC^3. For simplicity we omit the ReLU function and the batch normalization layer.

ℓ^0	LGN layer: a single filter Ψ^0 of size 11×11;
ℓ^1	64 filters of size 7×7; lateral connectivity kernel K^1 of size $13 \times 13 \times 64 \times 64$; max pooling of square size 2;
ℓ^2	64 filters of size $5 \times 5 \times 64$;
ℓ^3, ℓ^4	64 filters of size $3 \times 3 \times 64$; max pooling of square size 2;
ℓ^5, ℓ^6	64 filters of size $3 \times 3 \times 64$;
ℓ^7	128 filters of size $3 \times 3 \times 64$; max pooling of square size 2;
ℓ^8, ℓ^9, ℓ^{10}	128 filters of size $3 \times 3 \times 128$; max pooling of square size 2;
FC^1, FC^2, FC^3	1000, 200 and 10 output units respectively.

Since the neural network should receive as input a gray scale image, we transform the CIFAR-10 dataset into gray scale images. All the parameters except the kernels K^1 were first pre-trained in the absence of lateral connections for a maximum of 800 epochs, with early stopping when validation accuracy failed to increase for 80 consecutive epochs. After the pre-training we add the lateral connections in layer ℓ^1, thus employing the full update rule in Eq. (5.1), and implement a second

training stage: we re-update the whole architecture including the lateral kernel K^1 for a maximum of 800 epochs with early stopping as in the first phase. The initial purely feedforward training is intended both to obtain more stable learning of the receptive profiles, and to simulate the pre-existing orientation tuning of receptive profiles in V1 prior to the development of horizontal connections [Espinosa and Stryker, 2012]. However, after the "initialization" stage, all the network weights are trained jointly: this allows the feedforward weights to possibly readjust in the presence of lateral connections.

In order to enhance the generalization ability of the neural network, along with the weight regularization and the batch normalization layers, we apply dropout with dropping probability equal to 0.5 after the convolutional layer ℓ^{10}. This dropout is intended to reduce overfitting in the final classification layers by weakening the reciprocal dependencies between weights of the same layer: this allows the network to have a more stable selection of the features relevant for image classification [Srivastava et al., 2014]. We also apply dropout with dropping probability 0.2 when applying the kernel K^1 [see also Semeniuta et al., 2016] in the second training. Randomly dropping 20% of the lateral connections at each weight update has the purpose of avoiding co-adaptation of the connectivity weights, thus reinforcing their dependency on the intrinsic geometric properties of the receptive profiles. Applying dropout increase the performance on the test set from 72.24% to 80.28%. The network architecture and optimization procedure are coded using Pytorch [Paszke et al., 2017].

The performance of the neural network are quantified as accuracy (fraction of correctly predicted examples) on the CIFAR-10 testing dataset. We train several CNN architectures varying the number of layers and features. For the analyses described in section 5, we select the architecture that have reached the best mean performances (80.28%$\pm$ 0.17 over 20 trained instances of the

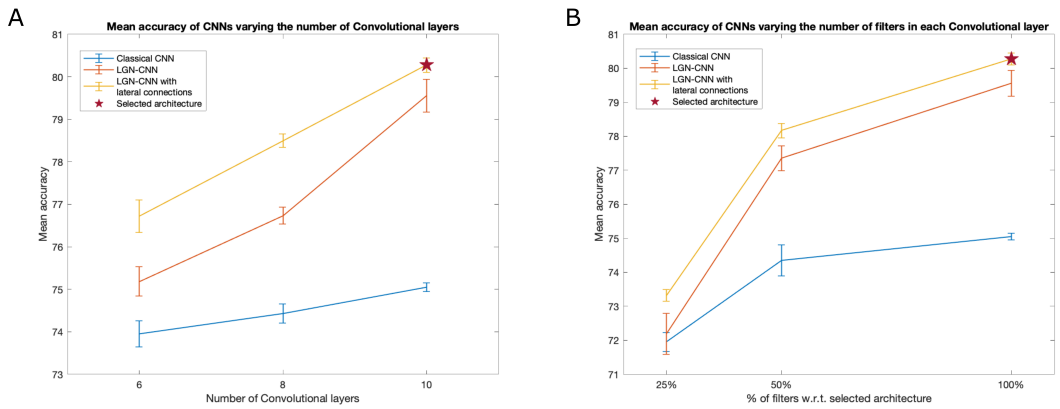


Figure 5.2 – Behavior of the mean testing accuracy w.r.t. the number of layers (A) and units (B), for both classical CNNs and LGN-CNNs, with and without the lateral connectivity. Error bars represent the standard error of the mean over 20 trainings. The selected LGN-CNN with transition kernel is the one marked by a red star in both plots. **(A)** The x-axis represents the number of "standard" convolutional layers. **(B)** The x-axis represents the number of filters in each convolutional layer, expressed as a percentage w.r.t. the number of filters in the selected model.

model), i.e. the one detailed above. Figure 5.2 shows the mean testing accuracy of several CNNs varying the number of layers and units, for both classical CNNs and LGN-CNNs, with and without the lateral connectivity. As expected, increasing the number of convolutional layers lead to better performances – see panel A, plotting the accuracy varying the number of standard convolutional layers (i.e. not including ℓ^0 or lateral connections). The same happens varying the number of units of each convolutional layer: panel B compares the mean performance of the selected model (marked by a star in both plots) with CNN architectures including only the 25% and 50% of the number of convolutional filters in each layer (e.g. each curve shows the accuracy values for architectures that have 16, 32 and 64 filters in ℓ^1 respectively). Let us note that, since we are mainly interested in the emergence of symmetries, we do not focus on reaching state-of-the-art performances on classifying the CIFAR-10 dataset. However, our architecture reaches fair performances that

can be increased by adding further convolutional layers and/or increasing the number of units.Furthermore, we observe that all the architectures examined show comparable behaviors as regards the invariance properties of the convolutional layers ℓ^0 and ℓ^1 and the connectivity kernel K^1. Therefore, it is reasonable to expect a similar behavior also when further increasing the complexity of the model.

5.3 Results on the LGN-CNN with transition kernel K^1

5.3.1 Emergence of rotational symmetry in the LGN layer

As described in [Bertoni et al., 2022] and shown in chapter 4 the introduction of a convolutional layer ℓ^0 containing a single filter Ψ^0 models the role of the LGN. This first layer acts as a pre-filtering step of the input visual stimulus before it reaches the V1 cells. We have also shown that a rotational invariant pattern is attained by Ψ^0, arguing that such behavior should be present also for deeper and more complex architectures. In this section we are going to see if this behavior persists.

Figure 5.3A shows the filter Ψ^0 obtained after the training phase. As expected it has a radially symmetric pattern and its maximum absolute value is attained in the center. Thus, we approximate the filter with the classical LoG model for the RP of an LGN cell. To do so, we find the optimal value for the parameter σ in Eq. (1.1) by using the built-in function *optimize.curve_fit* from the Python library SciPy. Figure 5.3B shows its approximation with $\sigma = 0.184$. The two functions have a high correlation of 93.67%, obtained by applying the built-in function *corrcoef* from the Python library NumPy.

This result enforces the one obtained in the LGN-CNN architecture described in chapter 4.2. Indeed, Ψ^0 spontaneously evolves into a radially symmetric pattern during the training phase, and more specifically its shape approximates the typical geometry of the RPs of LGN cells.

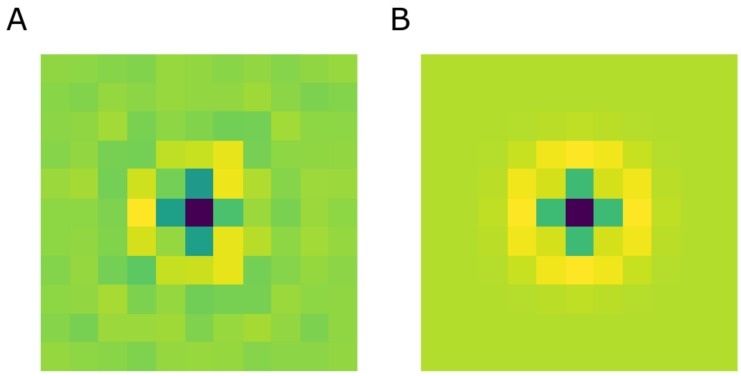


Figure 5.3 – **(A)** The learned filter Ψ^0 of the LGN-CNN architecture with transition kernel. **(B)** Its approximation as a LoG, with optimal $\sigma = 0.184$, yielding a correlation of 93.67% with the learned filter.

5.3.2 Emergence of Gabor-like filters in the first layer

As introduced in section 1, the RPs of V1 simple cells can be modeled as Gabor functions by Eq. (1.2). Furthermore, the first convolutional layer of a classical CNN architecture usually shows Gabor-like filters [see e.g. Serre et al., 2007]. This behavior suggests that the first convolutional layer of classical CNNs assumes a role analogous to V1 orientation-selective cells. In this section, we first approximate the filters of ℓ^1 as a bank of Gabor filters. This allows us to obtain a parameterization in terms of position (x_0, y_0), orientation θ and parity ϕ of each filter. Thus, the parameterization will provide a suitable set of coordinates for studying the corresponding lateral kernel in the $\mathbb{R}^2 \times S^1$ domain in the following sections.

After the training phase, Gabor-like filters emerge spontaneously in ℓ^1 as expected (see figure 5.4A). We approximate the filters in ℓ^1 with the Gabor function in Eq. (1.2), where all the parameters are found using the built-in function *optimize.curve_fit* from the Python library SciPy. The obtained approximation is significant; indeed, the mean Pearson's correlation coefficient (obtained using the built-in function *pearsonr* from the Python library SciPy)

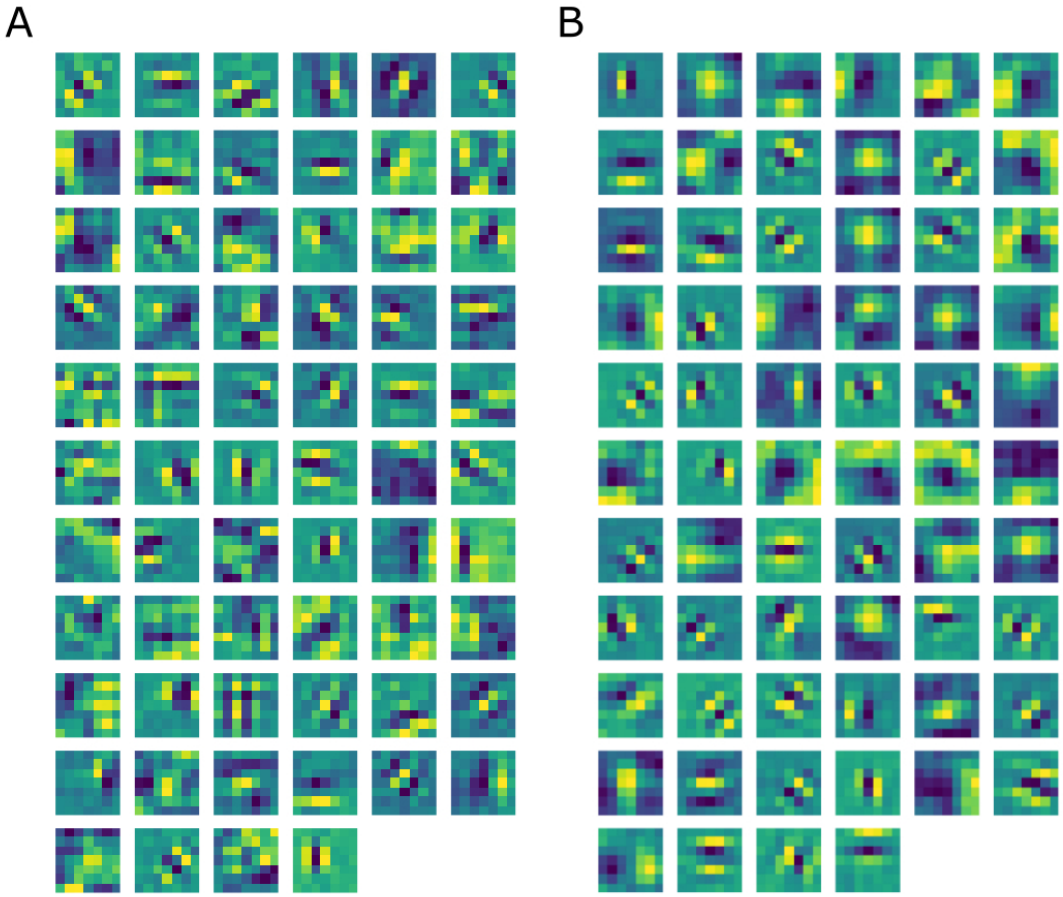


Figure 5.4 – **(A)** Learned filters of ℓ^1 of our CNN architecture. **(B)** Learned filters of ℓ^1 of the same CNN architecture, but without ℓ^0.

between the filters and their Gabor approximations is 89.14%. Furthermore, all correlation values are statistically significant ($p < .001$). This step is crucial for the analysis of the kernel K^1 since a good approximation of ℓ^1 filters will allow us to re-parameterize the kernel itself in the space $\mathbb{R}^2 \times S^1$.

We also expect that the introduction of the pre-filtering layer ℓ^0 should influence the filters of the following layer ℓ^1. Since Ψ^0 acts as a LoG on the input image we expect that ℓ^1 filters should not show any Gaussian-like filters. Indeed, as shown in figure 5.4A, the layer ℓ^1 only contains filters sharply tuned for orientation, with no Gaussian-like filters. On the other hand, by removing ℓ^0, the two types of

filters are mixed up in the same layer. Figure 5.4B shows the ℓ^1 filters of a trained classical CNN with the same architecture, but without ℓ^0, where there are several Gaussian-like filters. However, we can observe from figure 5.4A that some filters of the LGN-CNN architecture with transition kernel have more complex shapes that are neither Gaussian-, nor Gabor-like; indeed, since we have not introduced other geometric structures on following layers, it is reasonable to observe the emergence of more complex patterns.

As a last step, we decide to re-organized the filters of ℓ^1 in a different way. We first split the bank in odd and even filters using the parameter ϕ and then we reordered them w.r.t. the orientation θ. The interest of the organization in odd and even filters comes from comparison with neurophysiological data. Indeed, it is well known that the RPs of simple cells are organized in odd and even profiles, with different functionality. The odd ones are responsible for boundary detection, the even ones for the interior. The reorganization w.r.t. θ is useful to see if all the orientations are analyzed by the filter bank and to rearrange the kernel K^1 w.r.t. that parameter.

Indeed, we split the filter bank of ℓ^1 w.r.t. the parity of their approximation, indicated by the parameter ϕ, that was forced to be between $-\pi$ and π. Specifically, we labelled a filter as odd if $\frac{\pi}{4} < |\phi| < \frac{3\pi}{4}$, as even if $0 < |\phi| < \frac{\pi}{4}$ or $\frac{3\pi}{4} < |\phi| < \pi$. Figures 5.5 and 5.6 show the odd and even filters rearranged w.r.t. the orientation θ. For the sake of visualization, the even filters whose central lobe have negative values are multiplied by -1. The orientation values are quite evenly sampled, allowing the neural network to detect even small orientation differences. Most of the filters have high frequencies, allowing them to detect thin boundaries, but some low-frequency filters are still present.

5.3.3 Re-parameterization of the connectivity kernel using the first layer approximation

In this section, we examine the learned transition kernel K^1, to investigate whether it shows any invariances compatible with the known properties of the lateral connectivity of V1. In order to study the selectivity of the connectivity

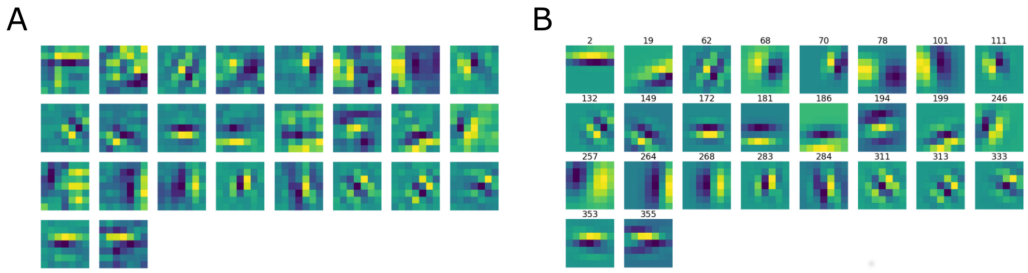


Figure 5.5 – **(A)**: the learned filters of ℓ^1 with odd parity, ordered w.r.t. the orientation θ obtained from the Gabor approximation. **(B)**: the approximating odd Gabor filters, labelled by their orientation.

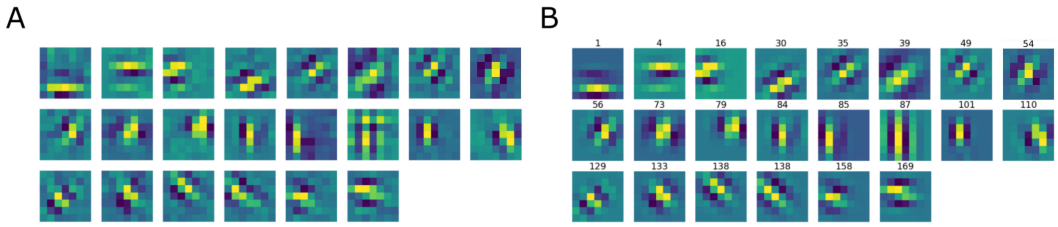


Figure 5.6 – **(A)**: the learned filters of ℓ^1 with even parity, ordered w.r.t. the orientation θ obtained from the Gabor approximation. **(B)**: the approximating even Gabor filters, labelled by their orientation.

kernel to the properties of ℓ^1 filters, we first rearrange it based on the set of coordinates in $\mathbb{R}^2 \times S^1$ induced by the Gabor approximation of the filters.

We first split the kernel w.r.t. the parity of ℓ^1 filters, resulting in two separate connectivity kernels for even (figure 5.6A) and odd (figure 5.5A) filters. We then adjust the spatial coordinates of each kernel using the estimated Gabor filter centers (x_0, y_0) in order to re-center the kernel of each filter w.r.t. any other filter of ℓ^1. Specifically, for each f, g in $\{1, \dots, n\}$, we shift the kernel $K^1(\cdot, \cdot, f, g)$ of Eq. (5.2) so that a displacement of $(i, j) = (0, 0)$ corresponds to the situation where the centers of the filters Ψ_f and Ψ_g coincide.

So far we have re-organized each kernel w.r.t. the spatial domain $\mathbb{R}^2$ without modifying the third dimension. Indeed, the original ordering of f, g in $\{1, \dots, n\}$

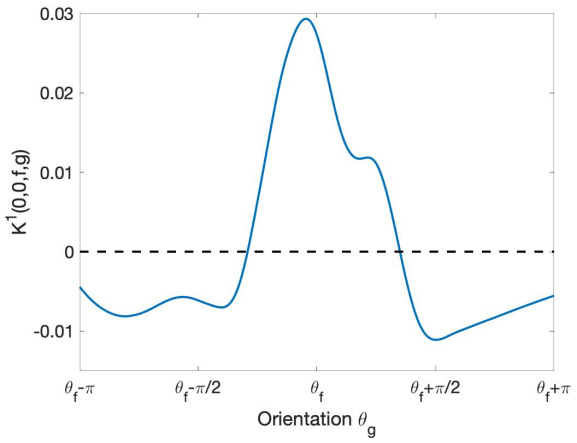


Figure 5.7 – Behavior of learned short-range intracortical connections w.r.t. the relative orientation. The curve displays the strength of interaction between the filter Ψ_f with orientation $\theta_f = \frac{2\pi}{5}$ and the other filters Ψ_g centered at the same point (i.e. with relative displacement $= (0, 0)$) as a function of their orientation θ_g. The curve has been smoothed using the MatLab built-in function *smoothdata*.

has no geometric meaning. However, each filter Ψ_f is now associated with an orientation θ_f obtained from the Gabor approximation. Therefore, we are able to rearrange the slices of K^1 so that the f and g coordinates are ordered by the corresponding orientations θ_f and θ_g. Indeed we expect a high correlation between filters with similar orientation and a low correlation between filters with different orientation as it happens in the horizontal connectivity of V1 cells. By fixing one filter Ψ_f, we then obtain a 3-D kernel

$$(i, j, g) \mapsto K^1(i, j, f, g) \tag{5.3}$$

defined on $\mathbb{R}^2 \times S^1$, describing the connectivity between Ψ_f and all the other ℓ^1 filters, each shifted by a set of local displacements $(i, j) \in \{-6, \dots, 6\} \times \{-6, \dots, 6\}$.

5.3.4 Non-maximal suppression within orientation hypercolumns

The re-parameterized connectivity kernel K^1 describes the strength of interaction as a function of relative displacement and relative orientation of the profiles. In this section, we restrict ourselves to the case of a displacement $(i, j) = (0, 0)$, i.e. the case of profiles belonging to the same hypercolumn of orientation. We thus study, for fixed f, the function

$$\theta_g \mapsto K^1(0, 0, f, g) = K^1(0, 0, \theta_f, \theta_g). \tag{5.4}$$

This function, plotted in figure 5.7, describes the learned pattern of excitation and inhibition between the profiles Ψ_f and Ψ_g as a function of the orientation θ_g. Notably, the resulting interaction profile showed a "Mexican hat"-like shape, indicating an enhancement of the response to the optimal orientation and a suppression of the response to non-optimal orientations. This type of profile has been shown to yield a sharpening of the orientation tuning [Bressloff and Cowan, 2002].

5.3.5 Association fields induced by the connectivity kernel

We are now interested in the analysis of the re-parameterized kernel in terms of the association fields induced by the kernel itself. Indeed, starting from the re-parameterized kernel centered around a filter Ψ_f as in Eq. (5.3), we use the θ-coordinates to construct a 2-D association field as in [Sanguinetti et al., 2010]. We first define a 2-D vector field by projecting down the orientation coordinates weighted by the kernel values. Specifically, for each spatial location (i, j), we define

$$V(i, j) := \max_g K^1(i, j, f, g) \cdot \frac{\sum_{g=1}^n K^1(i, j, f, g) v_g}{\| \sum_{g=1}^n K^1(i, j, f, g) v_g \|}, \tag{5.5}$$

where $v_g \in \mathbb{R}^2$ is a unitary vector with orientation θ_g. This yields for each spatial point (i, j) a vector whose orientation is essentially determined by the leading θ values in the fiber, i.e. the ones where the kernel has the highest values. The norm of the vector is determined by the maximum kernel value over (i, j). Finally, we

defined the association field as the integral curves of the so-obtained vector field V starting from points along the trans-axial direction in a neighborhood of $(0, 0)$.

Figure 5.8B shows the association field obtained from the kernel computed around the filter Ψ_f in figure 5.8A, with orientation $\theta_f = \frac{3\pi}{10}$. The vector field V is plotted using the Matlab function *quiver*, and the integral curves are computed using the Matlab function *streamline*. The vectors and curves are plotted over a 2-D projection of the kernel. The latter is obtained by first resizing the kernel by a factor of 10 using the built-in Matlab function *imresize* for better visualization, and then projecting down on the spatial coordinates by taking the maximum over g. Note that around $(0, 0)$ the association field is aligned with the orientation of the starting filter Ψ_f, while it starts to rotate when it moves away from the center. This behavior is consistent with the typical shape of psychophysical association fields of the horizontal connectivity of V1 cells, see section 1.

We can also note that a similar pattern arises with the integral curves of [Sanguinetti et al., 2010], see figure 1.5D. However, in our case the rotation is less evident since the spatial displacement encoded in the kernel K^1 is more localized than the edge co-occurrence kernel constructed in [Sanguinetti et al., 2010]. In spite of that, we approximate each integral curve as a circular arc, obtained by fitting the parameter k of Eq. (1.7) to minimize the distance between the two curves. Figure 5.8C shows in red the approximation of each integral curve. The empirical curves induced by the learned connectivity kernel are very close to the theoretical curves, with a mean Euclidean distance of 0.0036. Indeed, the learned kernel generates a good approximation of the theoretical integral curves; in the next section we are going to see if also the vector field is well modelled by the theoretical one.

5.3.6 Comparison of transition kernel with the solution of the heat equation

As recalled in chapter 1, there exists a relationship between connectivity kernel and the fundamental solution of Sub-Riemannian operators. In analogy with this relation, we compare the learned transition kernel K^1 with the fundamental solution

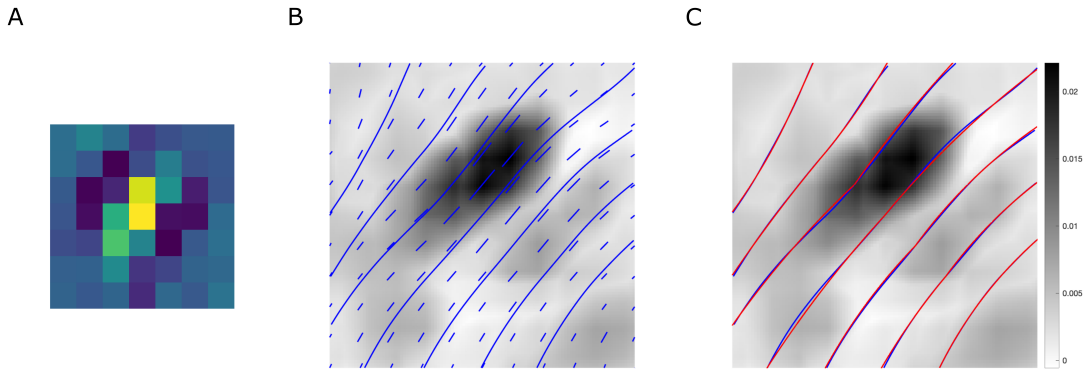


Figure 5.8 – **(A)** The 7×7 filter Ψ_f, with orientation $\theta_f = \frac{3\pi}{10}$. **(B)** The vector field V and its integral curves obtained from the kernel K^1 computed around Ψ_f, with $\theta_f = \frac{3\pi}{10}$. **(C)** The association field of the transition kernel K^1 (blue), and its approximation using the integral curves defined in (1.7) (red). For better visualization the kernel has been resized by a factor of 10 using the built-in Matlab function *imresize*.

of the sub-Riemannian heat equation in $\mathbb{R}^2 \times S^1$

$$\frac{\partial u}{\partial t} = -\alpha \Delta_{X_1, X_2} u = -\alpha \left(X_1^2 u + X_2^2 u \right), \tag{5.6}$$

where the sub-Laplacian operator is generated by the vector fields introduced in [Citti and Sarti, 2006] and defined by Eq. (1.6). To this end, we fit the parameter α by comparing the vector field V generated from the kernel K^1 described in the previous section with the 2-D vector field obtained with the same procedure from the fundamental solution of (5.6). More precisely, the optimal parameter α is chosen as the one minimizing the mean angular difference between the two vector fields. We obtain an optimal value $\alpha = 4.40$, with a mean angular difference of 0.0340. Figure 5.9A shows the vector field V in blue, and the sub-Riemannian vector field with optimal α in red. As already observed, the learned kernel K^1 captures the connections over a localized spatial area. Figure 5.9B shows in green the sub-Riemannian vector field over a wider spatial region, with the local area that best fits the vector field V highlighted in red.

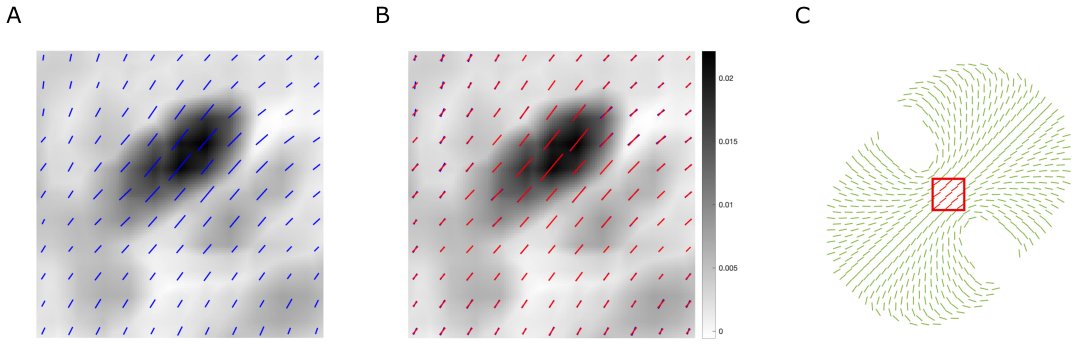


Figure 5.9 – **(A)** In blue: the vector field of the transition kernel K^1 around Ψ_f. In red: the vector field of the best-fitting fundamental solution of the sub-Riemannian heat equation, with $\alpha = 4.40$. **(B)** In green: the vector field of the best-fitting fundamental solution of the sub-Riemannian heat equation over a wider spatial region. In red: the local vector field fitted to the learned kernel.

Chapter 6

The Retinex effects of LGN-type learned filters

In this chapter we focus on testing the rotational symmetric filter Ψ^0 of the LGN-CNN architecture described in chapter 4 on Retinex effects and contrast-based illusions. In chapter 2 we have described how the model of [Morel et al., 2010] have been neurally interpreted in [Citti and Sarti, 2013] and applied to LoG. Our purpose is to extend this algorithm to a general operator and to test it on Ψ^0.

6.1 Retinex algorithm via learned kernels

We now introduce the Retinex algorithm we apply to the operator Ψ^0. Indeed, our approach aims to find the perceived image $\widetilde{I}$ starting from a visual stimulus I. We represent the action of the cortical RPs as a convolutional operator

$$O_M : L^2(R^2) \to L^2(R^2), \tag{6.1}$$

associated to a kernel

$$M : \Gamma \subset \mathbb{R}^2 \to \mathbb{R}. \tag{6.2}$$

Hence the action of the input image will be represented as

$$O_M(I) = M * I. \tag{6.3}$$

We assume that the perceived image is obtained via the application of another operator associated to a kernel

$$\widetilde{M} : \Gamma \subset \mathbb{R}^2 \to \mathbb{R} \tag{6.4}$$

via the following equation

$$\widetilde{I} = \widetilde{M} * (M * I). \tag{6.5}$$

We also impose that $\widetilde{M}$ is the kernel associated to the inverse operator to O_M, so that

$$M * \widetilde{M} = \delta. \tag{6.6}$$

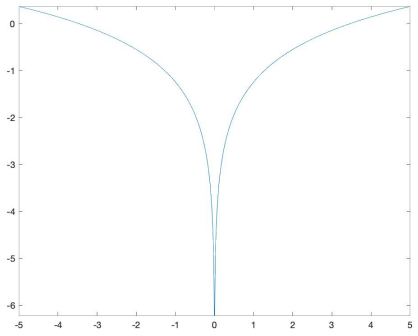
Consequently

$$M * \widetilde{I} = M * I. \tag{6.7}$$

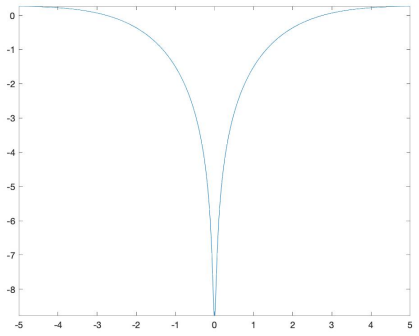
We observe that one main difference between our approach and the one proposed by Morel regards the Neumann boundary conditions. Indeed, Morel imposes directly the Neumann boundary conditions to the Poisson equation whereas, in our algorithm, the Neumann boundary conditions imposed to the filter are inherited from the inverse operator itself. In order to compare our algorithm with the Morel one we face the problem for a Laplacian operator as performed in [Morel et al., 2010]. As we will see the results for a Laplacian operator are comparable.

6.2 Application of the algorithm

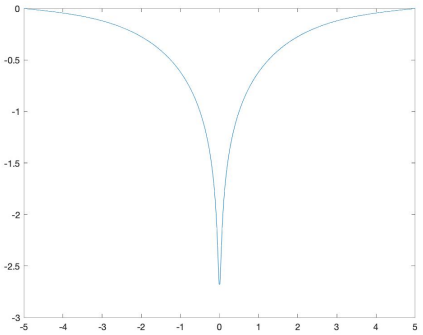
We can now test our algorithm to different operators and see if Retinex effects occur. We start by finding the inverse operators of some modeled and learned operators; then, we test the algorithm on two images. We point out that the Retinex model is an algorithm of contrast perception and does not try to improve the image quality.



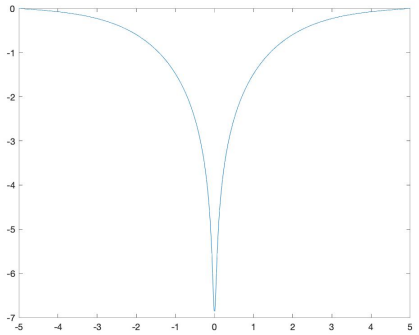
(a) *Exact inverse of a Laplacian* $\log(\sqrt{x^2+y^2})$ *in the interval* $[-5,5]$.



(b) *Inverse of discrete Laplacian.*



(c) *Inverse of LoG.*



(d) *Inverse of the first layer of an LGN-CNN.*

Figure 6.1 – Comparison between inverse operators. First row: inverse of the discrete Laplacian and its exact inverse operator. Second row: inverse operators of a discrete LoG and the first filter Ψ^0 of an LGN-CNN.

6.2.1 Detect the inverse operators

Applying the steepest descent method, we obtain an iterative process, which can be formally expressed as

$$\widetilde{M}_{t+1} = \widetilde{M}_t + dt \cdot (M * \widetilde{M}_t - \delta). \tag{6.8}$$

The algorithm stops at time T when $\frac{||\widetilde{M}_{T+1} - \widetilde{M}_T||_{L^1}}{dt} < \epsilon$, for a fixed error $\epsilon > 0$. Thus, $||M * \widetilde{M}_T - \delta||_{L^1} < \epsilon$ and indeed $\widetilde{M}_T$ becomes a good approximation of $\widetilde{M}$.

Finally, we can compare the image I with the reconstructed one $\widetilde{I}$ and see if any Retinex effects occur.

We show the inverse operators obtained through the algorithm described by equation (6.8). We start from the classical discrete Laplacian operator and then we move to some convolutional operators, in particular a discrete LoG and the filter Ψ^0 of the LGN-CNN. In order to compare the inverse operators, we show the 2D plots obtained by selecting a slice of the 3D inverse operator itself.

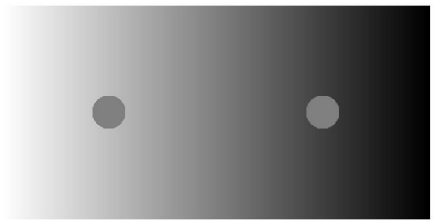
We first compare the inverse of the discrete Laplacian w.r.t. its exact inverse operator given by the function $\log(\sqrt{x^2+y^2})$. Figure 6.1b shows the approximation of the inverse of the discrete Laplacian and 6.1a shows the exact inverse in the interval $[-5, 5]$. We observe that the approximation is close to the exact one.

In the second row of figure 6.1 we compare the inverse operators of the LoG and Ψ^0 where figure 6.1c shows the LoG inverse operator and figure 6.1d shows the Ψ^0 inverse operator. Indeed, since we have shown in chapter 4 that Ψ^0 approximates the LoG we should expect similar inverse operators. In the next section we will see if also the Retinex effects are similar.

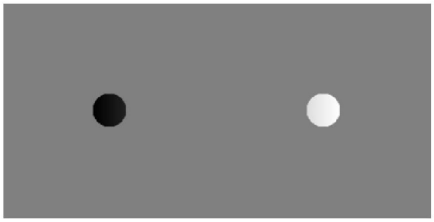
6.2.2 Circles on a gradient background

We start with a simple gray-scale image (see figure 6.2a) in order to see how the different operators work. We use the same image as in [Morel et al., 2010; Limare et al., 2011] in which they obtain remarkable Retinex effects. It has a background that shifts from white (value 1) to black (value -1) maintaining the gradient constant from left to right. There are also two gray dots (same value 0), the left one with a brighter background, the right one with a darker one. Because of the different backgrounds, our visual system perceives the two dots differently: the left one is perceived as darker with respect to its true color; the right one is perceived as brighter.

We first consider the discrete Laplacian operator since we expect to obtain the same Retinex effects as in [Morel et al., 2010]. Figure 6.2c shows the



(a) *Starting image with two gray dots on a gradient background.*



(b) *Retinex effects of exact inverse of Laplacian.*

(c) *Retinex effects of discrete Laplacian.*

(d) *Retinex effects of LoG.*

(e) *Retinex effects of Ψ^0 of LGN-CNN.*

Figure 6.2 – Retinex effects of some inverse operators on starting image 6.2a.

result obtained with our method. It is clearly the same result of the experiments performed by Morel and the values obtained in the position of the dots are close to -0.9 and 0.9 (indeed really close to completely black and white dots) whereas the entire background is gray with 0 value.

Thus, we apply the algorithm to the exact inverse of the Laplacian operator $\log(\sqrt{x^2+y^2})$. From figure 6.2b it is clear that the Retinex effects occur to the gray dots. Furthermore, we can observe that the contours of the two dots are more defined w.r.t. the discrete Laplacian operator. Also in this case the values in the position of the dots are close to -0.9 and 0.9.

Then, we perform our experiment with the inverse of a LoG. We can note from figure 6.2d that the Retinex effects occur also in this case (with values close to -0.9 and 0.9), similarly to the Retinex effects of the discrete Laplacian. Also in this case the contours of the dots are not completely clear.

Finally, we test the inverse of the convolutional operator Ψ^0 of the LGN-CNN introduced in chapter 4.2. In figure 6.2e we see that this operator shows Retinex effects where the values of the dots are close to -0.9 and 0.9. In this case the contours of the dots are even clearer than the ones obtained with the LoG and the discrete Laplacian.

To summarize, we have shown that our algorithm reproduces the same results of [Morel et al., 2010] in the case of the Laplacian operator. Furthermore, we have tested it on other operators obtaining remarkable results. It is particularly interested the case of the convolutional operator Ψ^0 since it is able to show Retinex effects even if it is a learned filter with no a-priori structure, enforcing the link between the LGN-CNN architecture and the visual system.

6.2.3 Adelson's checker

We can also test our algorithm on the Adelson's checker shadow illusion as in [Morel et al., 2010; Limare et al., 2011]. Since we are more interested in the Retinex effects of LoG and Ψ^0 operators we analyze their behaviors on the gray-scale image (see figure 6.3a). It shows a checkerboard with light gray and dark gray squares with a cylinder on it that shadows a part of the squares. In particular, the square labeled 'A' and the square labeled 'B' have the same gray-scale value (in our case, since -1 is black and 1 is white, they have -0,4953 value). The illusion is built in such a way that, even if they have the same value, they are perceived in a completely different way. Indeed square 'A', which is outside the shadow and surrounded by light gray squares, is perceived as a dark gray square. On the other hand, square 'B', which is inside the shadow and is surrounded by dark gray squares, is perceived as lighter.

We expect that the two operators should reproduce the same behavior of our perception, in particular square 'A' should have a smaller value whereas square 'B' should have a bigger value. Figure 6.3c shows the Retinex effects obtained with

the LoG operator. In particular, the value of square 'A' changes to -0,6553 whereas the value of square 'B' changes to 0,02351. Thus, we study the behavior of Ψ^0 operator whose results are shown in figure 6.3e. Even in this case Retinex effects occur where the value of square 'A' changes to -0,6035 and the value of 'B' changes to 0,2348.

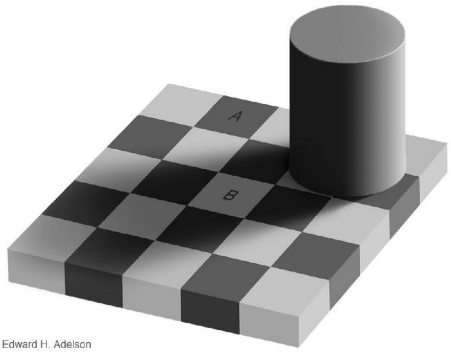
Figures 6.3d, 6.3d and 6.3f highlight the two squares 'A' and 'B' in the starting image and in the recovered images using the LoG and Ψ^0 operators. In this way it is clearer that the Retinex effects occur in both cases.

To summarize, we have shown that the filter Ψ^0 considered as a convolution operator shows Retinex effects really closed to the Retinex effects of the LoG. This enforces again the link between the structure of the LGN-CNN architecture and the structure of LGN.

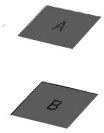
6.3 Information transmission efficiency

We are now interested to analyze the information transmission efficiency properties of the filter Ψ^0 and compare them to the LGN ones, expecting a similar behavior. Indeed, it is well established (see e.g. [see e.g. Reinagel and Reid, 2000; Zaghloul et al., 2003; Uglesich et al., 2009; Im and Fried, 2015; Pregowska et al., 2019] that the average firing rate of the retinal neurons that drive information to the LGN is much bigger than that of the LGN. In particular, LGN is able to delete spikes preserving the more informative ones leading to a loss of information.

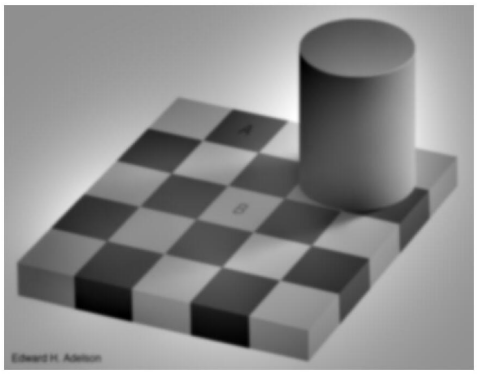
Thus, we study the information loss on the 8000 images of STL10 test set by convolving each image with Ψ^0 and computing the entropy from the histogram of the gray-scale values via the built-in MatLab function *entropy*. It turns out that on average the entropy decreases from 7.04 to 5.97 with a loss of 15.27 % of the information. Thus, we reconstruct the images using the Retinex algorithm described in section 6.1. The average entropy increases to 6.92 leading to a loss of just 1.83% of the information w.r.t. the original dataset. This suggests that almost the entire information contained in the visual stimulus can be reconstructed via some feedback or horizontal connections, where the reconstructed stimulus



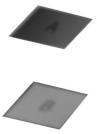
(a) *Grayscale Adelson's checker shadow illusion.*



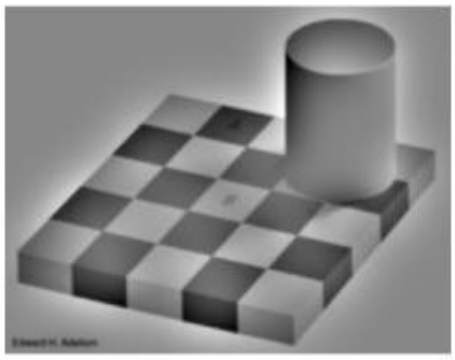
(b) *Grayscale Adelson's checker shadow illusion: the two squares.*



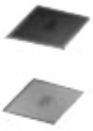
(c) *Retinex effects of LoG.*



(d) *Retinex effects of LoG: the two squares.*



(e) *Retinex effects of Ψ^0 of LGN-CNN.*



(f) *Retinex effects of Ψ^0 of LGN-CNN: the two squares.*

Figure 6.3 – Comparison between the Retinex effects of some operators on grayscale Adelson's checker shadow illusion.

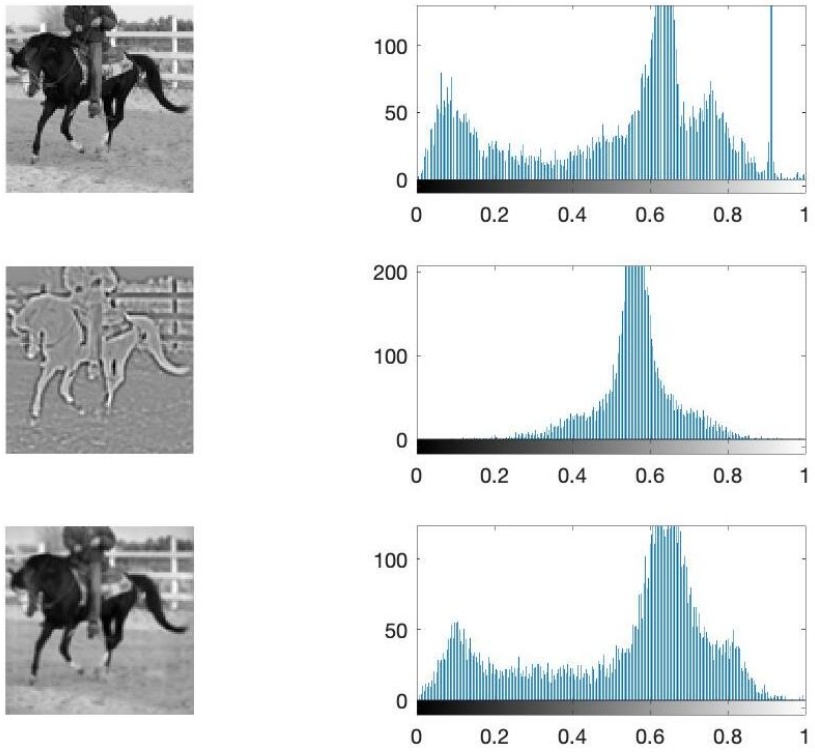


Figure 6.4 – On the left: a gray-scale image I, the convolved image $\widetilde{I}$ and the reconstructed image $\widetilde{I}$ via eq. (6.7). On the right: the corresponding histograms of the gray-scale values.

becomes invariant w.r.t. lightness constancy. Figure 6.4 shows an example of the convolution and the reconstruction performed on an image of the dataset with the corresponding gray-scale values histogram with respect to the entropy is calculated.

Chapter 7

Perceptual grouping with learned kernels

In this chapter we propose a generalization of the algorithm proposed in chapter 2, for detection of perceptual units. Up to now it has been proved that different kernels have different grouping capabilities. Hence, for every different problem it was necessary to introduce an ad hoc kernel. We propose to learn directly the kernels from a set of images with a neural network and to combine the kernels found at different layers.

7.1 A general method of group detection

Thus, we can now describe the general algorithm we can use to find the perceptual units on the images.

The method:

1. Choose a bank of filters;

2. Define a kernel with respect to the chosen bank;

3. Apply at each spatial location of the image all the filters and select the one with the best response (if necessary);

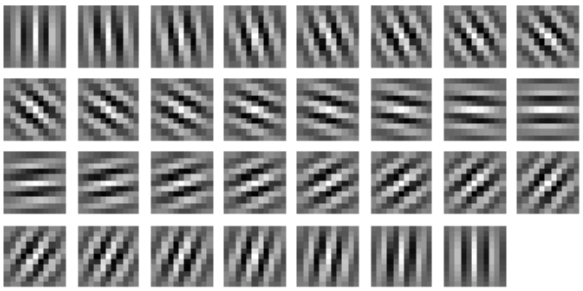


Figure 7.1 – Bank of 31 11×11 Gabor filters with $\sigma = 1$ and $\lambda = 1$.

4. Create an affinity matrix $N \times N$, where N is the number of pixels of the image, expressing the correlation between each pair of points, considering the kernel and the filters with best responses;

5. Find the eigenvectors with the highest eigenvalues that represent the main perceptual units of the image.

7.2 Modeled kernel

7.2.1 Kernel computed from Gabor filters

In this section we will generate a kernel from a bank of Gabor filters using the procedure introduce in section 1.2.1. First, we consider a bank of Gabor filters

$$(\Psi_p) = (\Psi_{x,y,\theta}),$$

as defined in equation (1.2), where $p = (x, y, \theta) \in \mathcal{G} = \mathbb{R}^2 \times S^1$. In the present case we consider Gabor filters defined in the interval $[-1.5, 1.5] \times [-1.5, 1.5] \times [-1.5, 1.5]$ with $\sigma = 1$ and $\lambda = 1$. The step in each spatial dimension is 0.3 and in the third dimension is 0.1; in this way our bank of filters is composed of 31 filters of size 11×11 (see figure 7.1).

We can now compute the generation kernel defined by equation (1.8) which can

be expressed in the case of Gabor filters by the following analytic formula:

$$K(p,0) = \sigma^2 \pi \exp\left(-\frac{x^2+y^2}{4\sigma^2} - \frac{2\sigma^2\pi^2(1-\cos\theta)}{\lambda^2}\right)\cos\left(\pi\frac{x(1+\cos\theta)+y\sin\theta}{\lambda}\right). \tag{7.1}$$

Note that this formula expresses the correlation between a generic point $p = (x, y, \theta)$ and the point $(0, 0, 0)$. If we compute the kernel $K(p, p_0)$ for a generic point $p_0 = (x_0, y_0, \theta_0)$ we can apply a rotation and a translation similarly to the generation of a Gabor filter described in section 1.3.2. Denoting $(x_0, y_0, \theta_0)_{\mathcal{G}}^{-1}$ the inverse in the group $\mathcal{G}$, we impose:

$$K((x,y,\theta),(x_0,y_0,\theta_0)) = K((x_0,y_0,\theta_0)_{\mathcal{G}}^{-1}(x,y,\theta),(0,0,0)).$$

Then, starting from this kernel we can define the connectivity kernel through the iterating procedure introduced in equation (1.9). In figure 7.2 it is displayed the second iteration starting from the filter with $\theta = 0$; in particular, we select at each position the filter with the highest correlation and we visualize only the ones that reach a certain minimum value of correlation. Note that there is a central connectivity area that has a similar behaviour with respect to the integral curves displayed in figure 1.5. Furthermore, there are two lateral lobes that represent the connectivity with filters with a similar direction at a certain distance; indeed neurally there exists a link between cells with collinear receptive profiles at a certain frequency distance. From a perceptive point of view this correlation permits to analyze visual stimuli containing parallel lines at the same distance.

In figure 7.3 we can see that the isosurface computed on this connectivity kernel is a good approximation of the solution of the heat equation. In particular the isosurface in the center is the approximation we consider whereas the lateral surfaces are due to the connections with collinear filters we have just explained.

Then, in presence of an image I, we apply the spectral grouping algorithm explained in section 2.2.2 with the resulting kernel K. The main difference between applying this connectivity kernel and the previous example is the presence of the direction of Gabor filters, so that the associated kernel acts on $R^2 \times S^1$. Indeed Gabor filters lift the input image defined on R^2 to an output $z_I = z_I(x, y, \theta)$ defined on the cortical plane via equation (1.3). Equation (1.4) allows to associate an

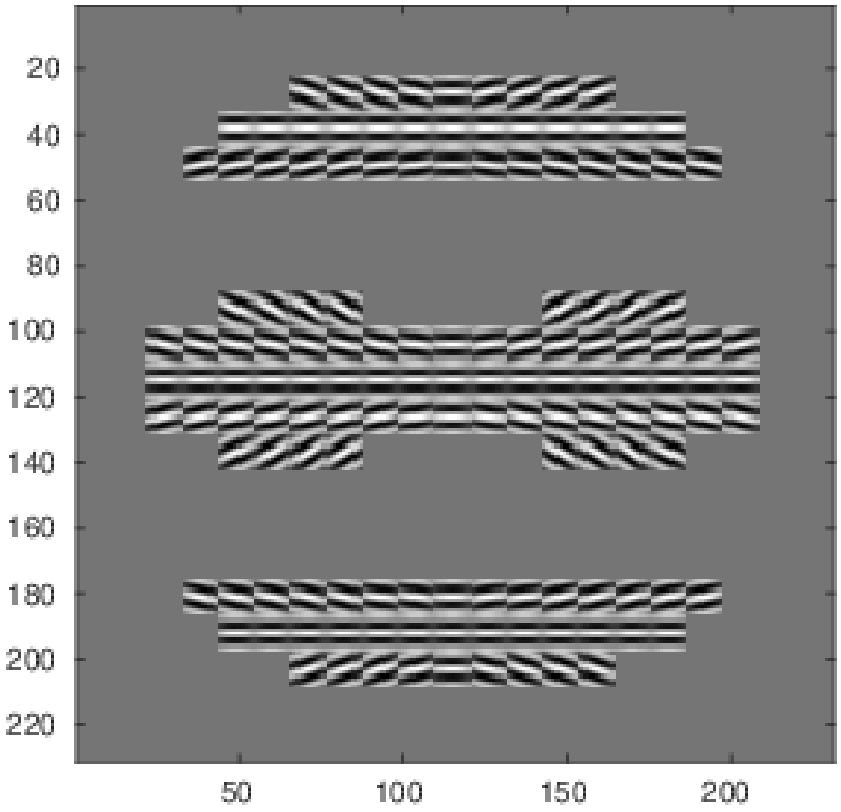
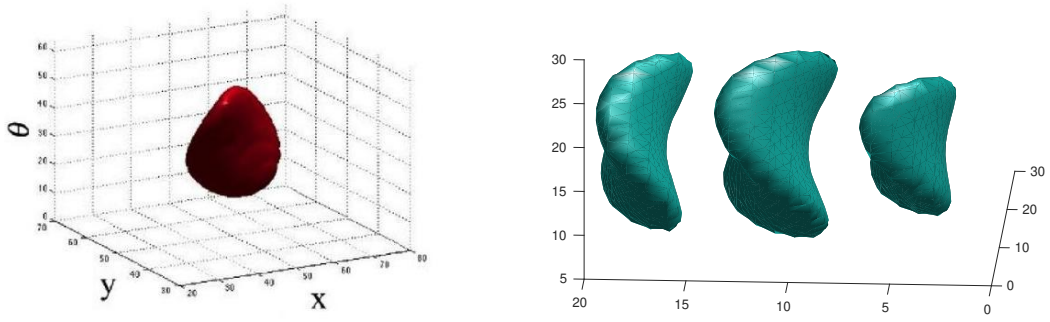


Figure 7.2 – 2 iterations Kernel with respect to the filter with $\theta = 0$.



(a) (b)

Figure 7.3 – (a) Solution of the heat equation. *From:* [Favali, 2017]. (b) Isosurface computed on the connectivity kernel iteratively generated from equation (7.1).

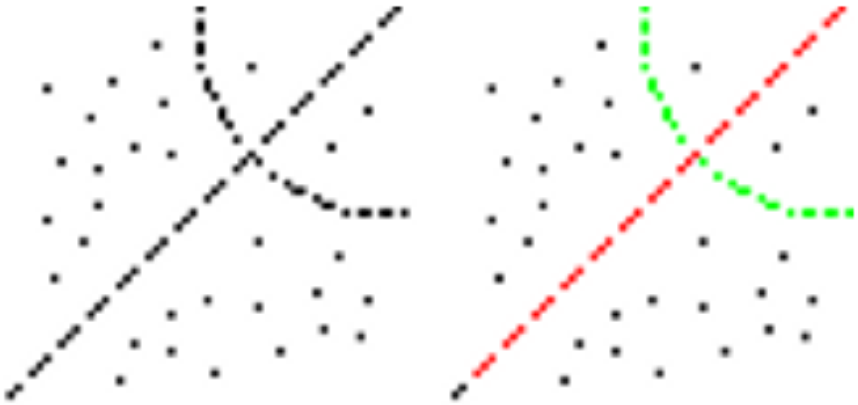


Figure 7.4 – On the left image containing collinear and cocircular oriented elements. On the right the grouping operated via a connectivity kernel generated by a bank of Gabor filters. In order: red and green eigenvectors.

orientation $\bar{\theta} = \bar{\theta}(x, y)$ to each point (x, y) selecting the highest response. Then we apply equation (2.2) obtaining the affinity matrix on the set Ω_I and we compute the eigenvectors.

Figure 7.4 displays the first two eigenvectors in red and green obtained through the algorithm; in particular, the image contains both collinear and cocircular orientation elements that represent a line and a arc of a circumference and this kernel clearly detects them. However, if we try to use this kernel in presence of clouds of points we can see that the results are completely different as the ones obtained using the Gaussian kernel. Figure 7.5 displays the first three eigenvectors. We can note that the three clouds are not perceived as distinct but the eigenvectors cross each other and the kernel seems to select orientation patterns. This result is reasonable since Gabor filters are able to detect the orientation of boundaries but not clumps of points.

Finally, we test this kernel with an image containing two clouds of points and a set of collinear segments. Figure 7.6 shows the grouping. The first eigenvector selects an orientation pattern in the bigger clouds of points and the second one represents the line with a set of points in the bigger cloud that are almost parallel with the line. Indeed, since the first eigenvector has not selected the entire cloud the

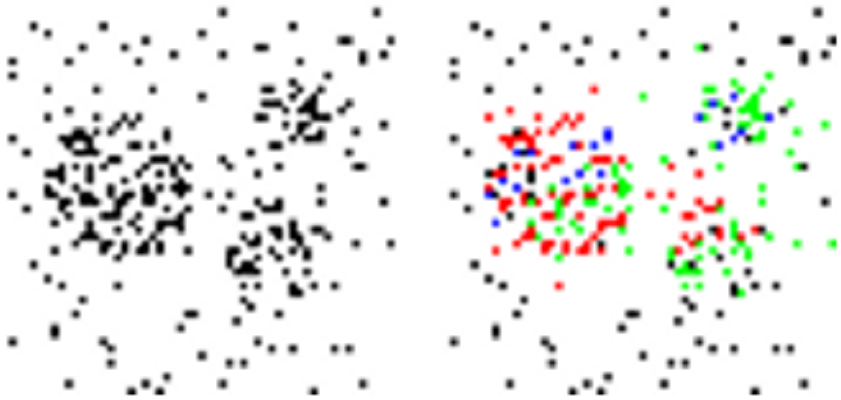


Figure 7.5 – On the left image containing three clouds of points. On the right its grouping operated via a connectivity kernel generated by a bank of Gabor filters. In order: red, green and blue eigenvectors.

algorithm has considered the remaining points and the line as the same perceptual unit. The third eigenvector has an orientation shape similar to the first one. With such an image the algorithm is not able to distinguish the three percepts but tries to find orientation patterns in the image.

7.3 Bank of learned filters

Now we test our algorithm with some learned banks of filters. We first describe the architectures of the neural networks and then we analyze their behavior on different images. We use different neural networks: a pre-trained one called GoogLeNet model imported from the Princeton version [DagNN format] [see Szegedy et al., 2014] and two neural networks we train on a set of gray-scale images containing numbers from 0 to 9 called MNIST [Deng, 2012].

GoogLeNet is a CNN trained on a set of RGB images for a classification task and it is composed of 2 convolutional layers: the first one is a bank of 64 filters of $[7 \times 7 \times 3]$ sizes and the second one a bank of 192 filters of $[3 \times 3 \times 64]$ sizes. After each convolutional layer it is applied a POOLING; then there is a sequence of 9 inception layers with a POOLING after the first two and before the last two.

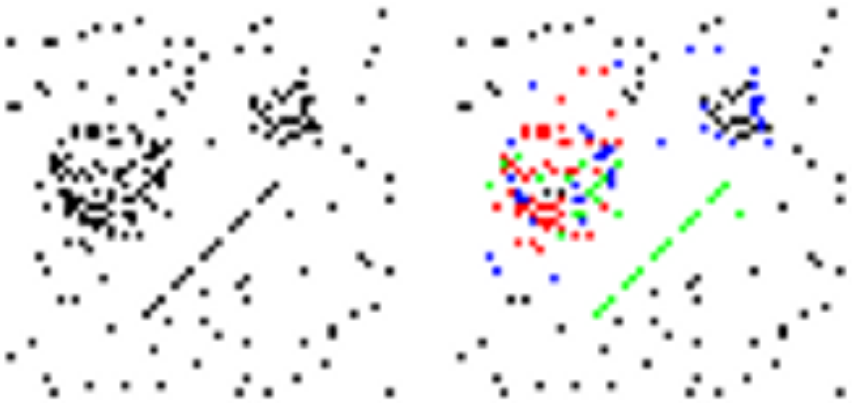


Figure 7.6 – On the left image containing two clouds of points and a set of collinear segments. On the right its grouping operated via a kernel generated from a bank of Gabor filters. In order: red, green and blue eigenvectors.

In particular we are interested in the first two convolutional layers that we use to find the perceptual units on the images. Furthermore the visual input is composed by a $[224 \times 224 \times 3]$ image and the output is a vector of 1000 size in which the n-th component represents the probability of the image to be the n-th label of the classification task.

On the other hand, the two CNN architectures are trained for a classification task on MNIST In particular, MNIST is a set of $70'000$ gray-scale images of sizes 28×28 representing a number from 0 to 9, divided in a training set of $60'000$ images and a test set of $10'000$ images. The objective of the neural networks is to detect the number that is displayed on the image and give it as the answer.

The first architecture is composed of three convolutional layers without 0-padding followed by a RELU or a POOLING. In particular, the first convolutional layer, which receives a 28×28 input, is composed of 128 filters of 5×5 sizes. The output of the first convolutional layer is indeed a tensor of size $[24 \times 24 \times 128]$ and after a POOLING it becomes a $[12 \times 12 \times 128]$ input for the second convolutional layer. The latter is composed again of 128 filters but with sizes $[5 \times 5 \times 128]$ and therefore its output is a $[8 \times 8 \times 128]$ tensor. Then, after another POOLING, there is a third convolutional layer that, receiving a $[4 \times 4 \times 128]$ input, is composed

of 500 layers of sizes $[4 \times 4 \times 128]$ that gives an output of sizes $[1 \times 1 \times 500]$. At this point a RELU is used and a fully connected layer is applied, modifying the length of the output vector to 10. Finally a softmax loss is used to calculate the error done to each batch during the training session. The output represents the probabilities of the image to be a number from 0 to 9 and the CNN selects the label with the highest probability.

The main difference between the first architecture and the second one is that the latter has a higher number of layers and receives larger images as input. In particular, the first convolutional layer, composed by 32 9×9 filters, receives a 56×56 gray-scale image. Then, a POOLING and a convolutional layer with 64 $[5 \times 5 \times 32]$ filters are applied; after them another POOLING and another convolutional layer with 64 $[5 \times 5 \times 64]$ filters are used. After the last POOLING and a convolutional layer composed by 500 $[3 \times 3 \times 64]$ filters a RELU is applied. Finally, it is applied a fully-connected layer, giving an output of length 10, and the error is calculated using a softmax loss.

7.3.1 Perceptual grouping with GoogLeNet

In this section we analyze the kernels generated from the first two banks of learned filters of GoogLeNet. We consider the first bank whose red canal is displayed in figure 7.7. The space in which the bank is defined is $\mathcal{G}^1 = \mathbb{R}^2 \times \mathcal{F}^0 \times \mathcal{F}^1$ where $\mathcal{F}^0 = 1 \dots 3$ refers to 3 color canals (RGB) and $\mathcal{F}^1 = 1 \dots 64$ labels the entire set of filters. Indeed, we can translate in the x-y direction each filter obtaining the following set

$$(\Psi^1_p) = (\Psi^1_{x,y,f^0,f^1}),$$

where $p = (x, y, f^0, f^1) \in \mathcal{G}^1$. Then we compute the connectivity kernel applying the iterating procedure introduced in equation (1.9).

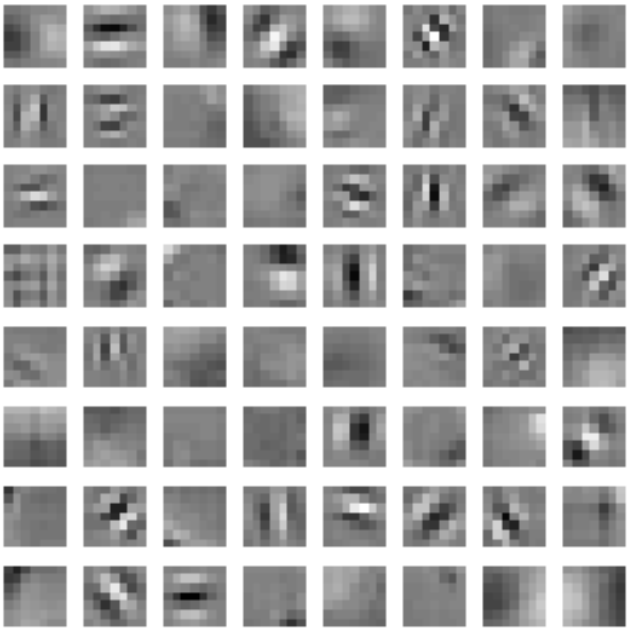


Figure 7.7 – Bank of 64 $[7 \times 7]$ filters of the red canal of the first convolutional layer of GoogLeNet.

Similarly, considering the second convolutional layer, we can define the space $\mathcal{G}^2 = \mathbb{R}^2 \times \mathcal{F}^1 \times \mathcal{F}^2$ where F^1 is the space of features of the first layer defined above and $\mathcal{F}^2 = 1 \dots 192$ labels the second bank of filters. Therefore, the second layer is composed of 192 $[3 \times 3 \times 64]$ filters that can be translated in the x-y direction, obtaining

$$(\Psi^2_p) = (\Psi^2_{x,y,f^1,f^2}),$$

where $p = (x, y, f^1, f^2) \in \mathcal{G}^2$. Also in this case we compute the connectivity kernel applying the iterating procedure introduced in equation (1.9).

Now let us consider a visual stimulus I to which we apply the spectral grouping algorithm explained in section 2.2.2. In the present case we should consider the presence of the parameter $f^1 \in \mathcal{F}^1$ that labels the set of filters and also the presence of the 3 canals of colours. Since we are not interested in the differences between each colour in the image we ignore that parameter and, thus, we can associate at each position $(x, y) \in \mathbb{R}^2$ the filter that gives the highest response, i.e. the maxima

suppression principle becomes:

$$z_I^1(x, y, \overline{f}^1) = \max_{f_i^1 \in \mathcal{F}^1} z_I^1(x, y, f_i^1), \tag{7.2}$$

where, analogously to equation (1.3), we can define

$$z_I^1(x_0, y_0, f_0^1) = \int_D \Psi^1_{x_0,y_0,f^0,f_0^1}(x, y, f^0) I(x, y, f^0) dx dy d\mu(f^0), \tag{7.3}$$

which expresses the response of a filter to a visual stimulus. The second layer receives an input of sizes $[112 \times 112 \times 64]$ after I has passed through the first convolutional layer and a POOLING. Analogously to the first layer we can define the function z_I^2 and the maxima suppression principle as follows:

$$z_I^2(x_0, y_0, f_0^2) = \int_D \Psi^1_{x_0,y_0,f^1,f_0^2}(x, y, f^1) z_I^1(x, y, f^1) dx dy d\mu(f^1), \tag{7.4}$$

$$z_I^2(x, y, \overline{f}^2) = \max_{f_i^2 \in \mathcal{F}^2} z_I^2(x, y, f_i^2). \tag{7.5}$$

Now we can build the two affinity matrices, A_I^1 for the first layer applying equations (7.3) and (7.2) and A_I^2 for the second one applying equations (7.4) and (7.5) as described in section 2.2.2. In this way we can use one matrix at a time to find the perceptual units of the image or weighting them together. In particular we define the two layers affinity matrix as

$$\widetilde{A}_I^\alpha = \alpha A_I^1 + (1 - \alpha) A_I^2, \tag{7.6}$$

where $0 \leq \alpha \leq 1$. Note that $\widetilde{A}_I^0 = A_I^2$ and $\widetilde{A}_I^1 = A_I^1$. In figure 7.8 we can see the first perceptual unit detected using $\widetilde{A}_I^1 = A_I^1$, i.e. considering only the first layer, and $\widetilde{A}_I^{\frac{1}{2}} = \frac{1}{2}(A_I^1 + A_I^2)$, i.e. a combination of the two layers. Note that there is a good improvement considering two layers instead of only one; however, none of them gives good results as the ones obtained using a kernel generated from a bank of Gabor filters. This is reasonable since some of the learned filters have a Gabor shape but not many orientations are present as we can see in figure 7.7.

In section 7.2.1, the kernel computed from a bank of Gabor filters was unable to detect clouds of points (see figure 7.5). Then, the idea is to use the two layers

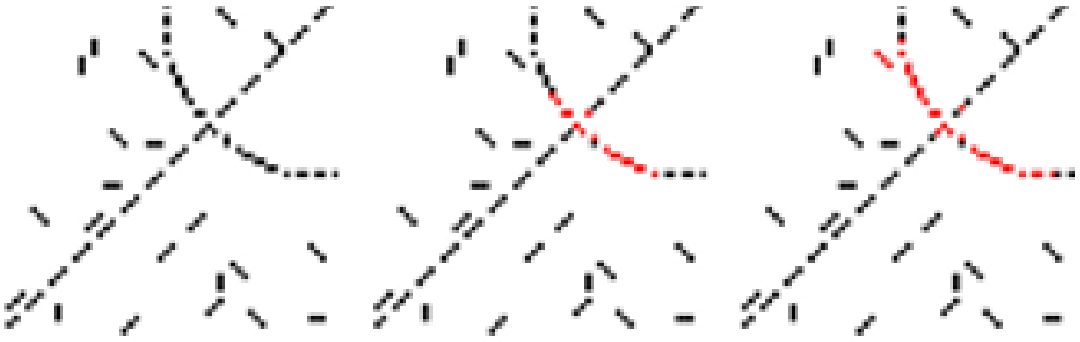


Figure 7.8 – On the left image containing collinear and cocircular oriented elements. In the center first eigenvector obtained with only the first layer. On the right first eigenvector obtained with a weighted combination of two layers.

of GoogLeNet to approach the same task since in this case we do not have only Gabor filters. Figure 7.9 represents the three eigenvectors obtained in this case. If we compare this result with respect to the Gaussian kernel one (see figure 2.3A) we can see that they are really similar and the eigenvectors are also in the same order.

Eventually, we use the two layered structure with an image containing both clouds of points and collinear elements. Figure 7.10 displays the grouping. We can see that the first two eigenvectors are the two clouds of points (however the first one does not contain the upper part of the cloud) and the third one is the line. Therefore the two layers structure of GoogLeNet is extremely flexible and can face completely different problems with better results with respect to the ones obtained with the models of section 7.2 thanks to the heterogeneity of the filters. Furthermore we have seen that using two layers gives better results than a single one; in section 7.3.2 we are going to study deeply this correlation for two multi-layered architectures.

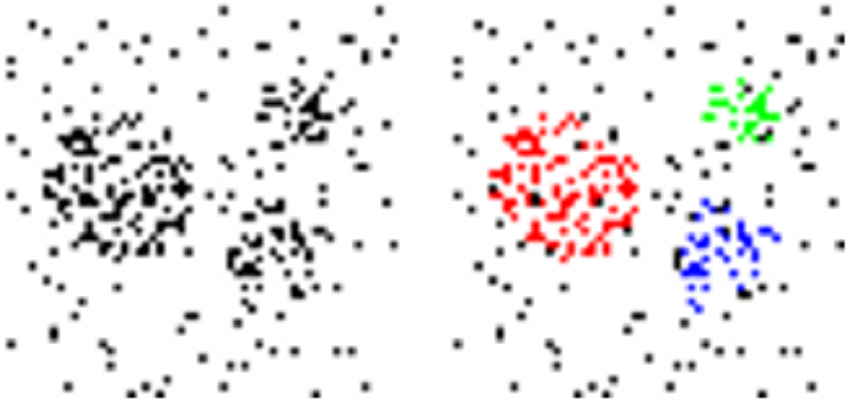


Figure 7.9 – On the left image containing three clouds of points. On the right its grouping operated via a weghted combination of two layers. In order: red, green and blue eigenvectors.

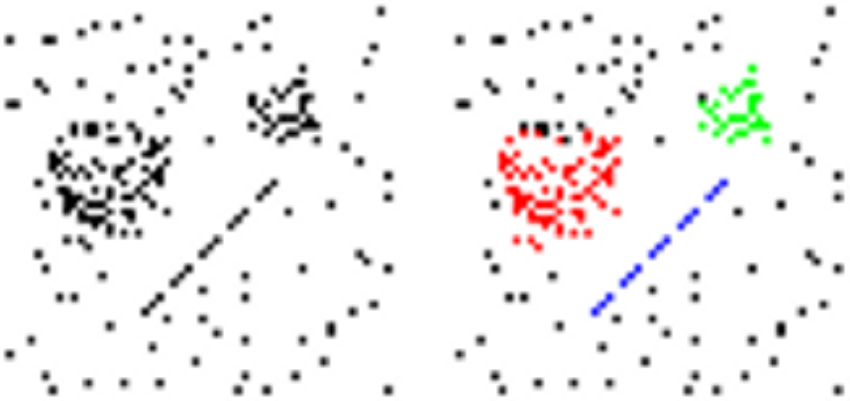


Figure 7.10 – On the left image containing two clouds of points and a set of collinear segments. On the right its grouping operated via a weighted combination of two layers of GoogLeNet. In order: red, green and blue eigenvectors.

7.3.2 Perceptual grouping with neural networks trained on MNIST

We can now study the perceptual grouping ability of two trained CNNs. Since they have several layers we can analyze their behaviors varying the number of layers involved in the task.

Let us consider the first architecture trained on MNIST. We analyze just the behaviour of the first two layers since the third one is extremely local (recall that the spatial dimension of the input is 4 as the spatial dimension of the filters). Therefore, we can consider the first bank of filters

$$(\Psi^1_p) = (\Psi^1_{x,y,f^1})$$

defined on $\mathcal{G}^1 = \mathbb{R}^2 \times \mathcal{F}^1$ where $\mathcal{F}^1 = 1 \dots 128$ labels the filters and the second bank of filters

$$(\Psi^2_p) = (\Psi^2_{x,y,f^1,f^2})$$

defined on $\mathcal{G}^2 = \mathbb{R}^2 \times \mathcal{F}^1 \times \mathcal{F}^2$ where $\mathcal{F}^2 = 1 \dots 128$. Then, given an input I we can define

$$z^1_I(x_0, y_0, f^1_0) = \int_D \Psi_{x_0,y_0,f^1_0}(x, y) I(x, y) dx dy,$$

and the principle of maxima suppression as

$$z^1_I(x, y, \overline{f}^1) = \max_{f^1_i \in \mathcal{F}^1} z^1_I(x, y, f^1_i).$$

For the second layer we can give similar definitions

$$z^2_I(x_0, y_0, f^2_0) = \int_D \Psi_{x_0,y_0,f^1,f^2_0}(x, y, f^1) z^1_I(x, y, f^1) dx dy d\mu(f^1),$$

$$z^2_I(x, y, \overline{f}^2) = \max_{f^2_i \in \mathcal{F}^2} z^2_I(x, y, f^2_i).$$

Then, as in section 7.3.1 we can apply the perceptual grouping algorithm defined in section 2.2.2 for each layer obtaining two affinity matrices A^1_I and A^2_I that can be combined in a unique matrix

$$\widetilde{A}^\alpha_I = \alpha A^1_I + (1 - \alpha) A^2_I, \tag{7.7}$$

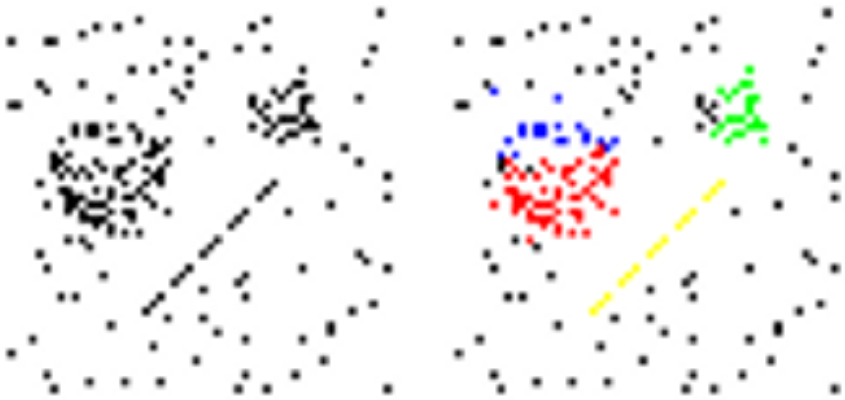


Figure 7.11 – On the left image containing two clouds of points and a set of collinear segments. On the right its grouping operated via a weighted combination of two layers. In order: red, green, blue and yellow eigenvectors.

with $0 \leq \alpha \leq 1$. Then we first test this architecture on the image with two clouds of points and collinear segments. Figure 7.11 displays the grouping result. The first and third eigenvectors represent the bigger clouds of points split into two parts. The second eigenvector contains the smaller cloud of points with a missing part and the last one the set of collinear segments. The grouping results are comparable with the one obtained using GoogLeNet architecture but they can be improved using a structure with more layers.

Now let us consider the second architecture and, in particular, the first three banks of filters Ψ^1, Ψ^2 and Ψ^3. In this case we have three spaces $\mathcal{G}^1 = \mathbb{R}^2 \times \mathcal{F}^1$, $\mathcal{G}^2 = \mathbb{R}^2 \times \mathcal{F}^1 \times \mathcal{F}^2$ and $\mathcal{G}^3 = \mathbb{R}^2 \times \mathcal{F}^2 \times \mathcal{F}^3$ and we can give similar definitions for h_I^1, h_I^2 and h_I^3 and the corresponding maxima suppression principles as for the first architecture. We can apply the perceptual grouping algorithm defined in section 2.2.2 to every layer obtaining three affinity matrices A_I^1, A_I^2 and A_I^3. Then we can define a linear combination of them as

$$\widetilde{A}_I^{\alpha,\beta} = \alpha A_I^1 + \beta A_I^2 + (1 - \alpha - \beta) A_I^3, \tag{7.8}$$

where $0 \leq \alpha, \beta, \alpha + \beta \leq 1$. In this setting we approach the same problem as before using the image with two clouds of points and a set of collinear segments to test

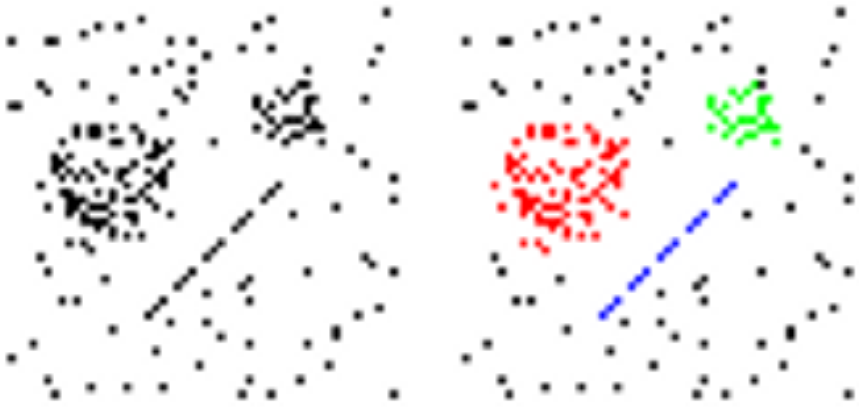


Figure 7.12 – On the left image containing two clouds of points and a set of collinear segments. On the right its grouping operated via a weighted combination of three layers. In order: red, green and blue eigenvectors.

the flexibility of the three layers structure. Figure 7.12 shows the grouping results. We can see that the three eigenvectors represents the two clouds of points and the line. Therefore also in this case the kernels are extremely flexible, adapting to different type of percepts, and they give better results than the ones obtained using the two layered architecture of GoogLeNet and the previous one.

7.3.3 Comparison between the two trained architectures

Here we test the two architectures on some simple images in order to understand how the first eigenvector changes and how far points can communicate.

Let us start with the 2 layers architecture. First, we consider a set of images containing two collinear segments at different distances, then a set containing the four corners of a square at different distances. The perceptual units obtained on the first set of images are displayed in figure 7.13 in which the first column represents the set of images whereas the other 3 columns the first eigenvector obtained using different choices of α in equation (7.7). In particular the second column uses only the first layer (i.e. $\alpha = 1$), the third

Figure 7.13 – First column: starting images with 2 segments at different distances. Second column: first eigenvector of first layer. Third column: first eigenvector of second layer. Fourth column: first eigenvector of first and second layers applied together.

column only the second layer (i.e. $\alpha = 0$) and the last column a weighted combination of the two layers (i.e. $\alpha = \frac{1}{2}$).

Looking carefully to the perceptual units, we can see that in the first three rows the two segments are seen as the same percepts for each choice of α. Then, increasing a bit the distance in the fourth row, the first layer becomes unable to link the two segments but can perceive only one. This means that at this distance for the first layer is hard to connect collinear segments. Besides, the second layer and the combination of them are still able to detect the two segments. When the segments become too far the second layer and the combination of layers can only detect a single segments as we can see in the last two rows. Note that this is an important result since there are experimental clues that suggest that the second cortical layer can connect points that are more distant than the first cortical layer alone.

A similar result is displayed in figure 7.14. Indeed in the second and third row the first eigenvector of the first layer is only a corner whereas the second

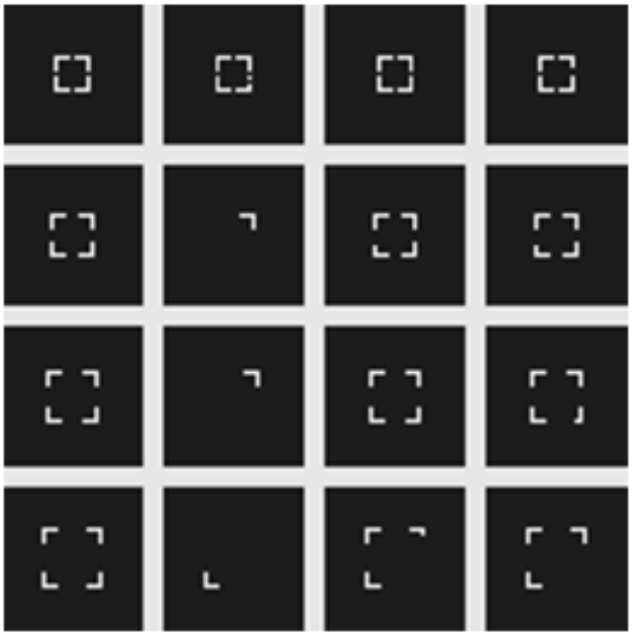


Figure 7.14 – First column: starting images with corners of a square at different distances. Second column: first eigenvector of first layer. Third column: first eigenvector of second layer. Fourth column: first eigenvector of first and second layers applied together.

layer and the combination can detect the entire square. In the last row the second layer and the combination can perceive three corners which is still far better than the eigenvector obtained using only the first layer. Note also that in this row the combination of the first two layers seems to work slightly better than only the second layer. This suggests that using many layers at the same time should improve the performances of the algorithm. However this is not enough to give such a conclusion and we should try to use an architecture with more layers.

Let us consider the 3 layers architecture and approach the same problem as before using a set of images representing the corners of a square at different distances. The perceptual grouping results are displayed in figure 7.15 considering the affinity matrix defined in equation (7.8). The second, third and fourth columns represent the first eigenvector obtained using respectively the first layer (i.e. $\alpha = 1, \beta = 0$), the second layer (i.e. $\alpha = 0, \beta = 1$) and the third layer (i.e. $\alpha = 0, \beta = 0$). Moreover,

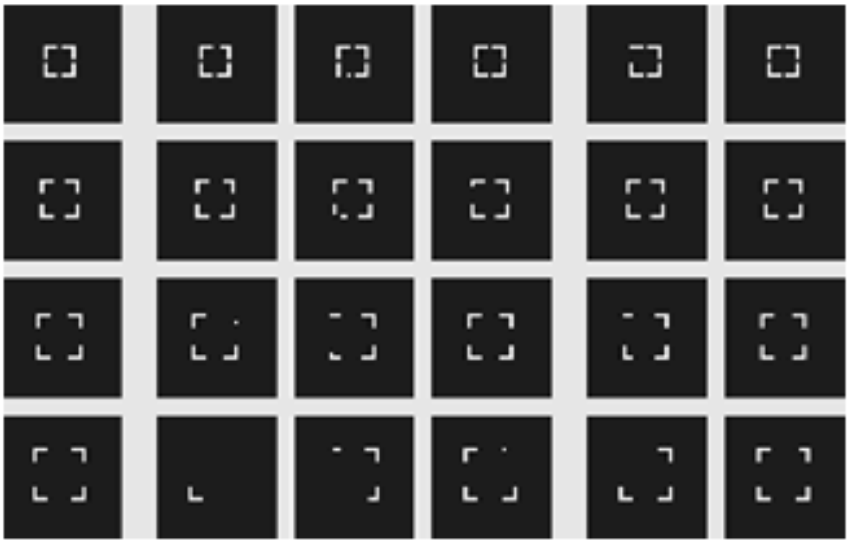


Figure 7.15 – First column: starting images with corners of a square at different distances. Second column: first eigenvector of first layer. Third column: first eigenvector of second layer. Fourth column: first eigenvector of third layers. Fifth column: first eigenvector of first and second layers applied together. Sixth column: first eigenvector of first, second and third layers applied together.

the fifth column represents the eigenvector obtained through a combination of the first two layers (i.e. $\alpha = \frac{1}{2}, \beta = \frac{1}{2}$) and the last one a combination of all three layers (i.e. $\alpha = \frac{1}{3}, \beta = \frac{1}{3}$).

The hypothesis suggested by the first architecture in this case are reinforced. Indeed we can see that the first layer gives worse results than the others two even applied alone. Moreover, if we use the first two layers or all three layers at the same time we obtain better results in general. Finally, the last row gives us a strong suggestion. Indeed, we can see that the first two layers detect only three corners whereas the combination of all three layers is able to perceive the entire square. This result is even clearer than the previous one and suggests that using more layers leads to better result on perceptual grouping.

Chapter 8

Conclusions

We introduced a biologically inspired CNN architecture able to correctly model the early visual pathway and tested the efficiency of the learned kernels on two perceptual problems: Retinex effects and individuation of perceptual units.

We have first described some state-of-the-art topics useful in the subsequent chapters. We started from the main structures of the visual system, focusing on the LGN and V1 and analyzing the group symmetries present in these layers. Then, we have recalled the Gestalt laws with the related grouping and the Retinex theory of Land with its more recent developments. Thus, we have introduced the main structures composing the CNNs, analyzing the similarities with the visual system and the invariance properties.

Then, we have moved towards our contributions. To begin with, we have proposed a biologically inspired CNN with a first layer ℓ^0, mimicking the role of the LGN. This layer contained a single filter Ψ^0 that has turned out to be rotational invariant and to approximate a LoG. We have proved the rotation invariance of Ψ^0 and we have studied the statistical distribution of the subsequent layer ℓ^1 with respect to the recordings of the neurons of a macaque's V1. Thus, we have introduced an LGN-CNN architecture with a transition kernel K^1, mimicking the role of the long-range horizontal connectivity of V1. We have studied the emergence of a rotational symmetric filter in ℓ^0 and of Gabor filters in ℓ^1. Then, thanks to the approximation of ℓ^1 filters with a Gabor function, we have re-parameterized the

transition kernel and we have studied its properties. We have first analyzed the association fields induced by the connectivity kernel and, then, the vector fields with the solution of the heat equation.

In the last chapters we have presented two applications on learned filters. We have described an algorithm for the Retinex effects via learned kernels, through the application of an inverse operator. Thus, we have applied our algorithm to two different images using different kernels and, in particular, the filter Ψ^0, comparing the corresponding Retinex effects. We have also studied the information transmission efficiency of the filter Ψ^0. Then, we have moved towards the problem of grouping. We have first analyzed the grouping obtained with a modeled kernel generated from a bank of Gabor filters. Then, we have put forward the results obtained with different banks of learned filters, showing a good flexibility due to the presence of different kinds of filters.

In the future, there are several possible research paths. Indeed, we could face the theoretical problem regarding the rotation symmetry of the first convolutional layer for a general LGN-CNN architecture. We could also introduce an autoencoder associated to this architecture, which can reconstruct perceived images with the Retinex effect. Furthermore, since the learned connectivity kernel K^1 describes the strength of interactions between filters shifted by up to 6 pixels w.r.t. one another in each direction, we could introduce a wider connectivity, modelled by taking further propagation steps. Thanks to the linearity, this composition would be equivalent to propagate the long-range kernel $2K^1 + K^1 * K^1$, obtained via "self-replication" from K^1 through convolution against itself. Indeed, the update rule becomes:

$$\frac{1}{2}\Big(\tilde{h}^1 + K^1 * \tilde{h}^1\Big) = \frac{1}{4}\Big(h^1 + 2K^1 * h^1 + K^1 * K^1 * h^1\Big).$$

Therefore, we could consider larger natural images and examine wider horizontal connectivity obtained via multiple diffusion steps – as well as to compare the propagated long-range connectivity kernels with theoretical kernels.

As other future perspectives, we would like to extend our study by considering a wider range of features; therefore, we could find a finer characterization of the geometry of first layer filters, including more complex types of receptive profiles.

Finally, we could extend our model to other convolutional layers, comparing their properties with higher cortices of the visual system.

Bibliography

Abbasi-Sureshjani, S., Favali, M., Citti, G., Sarti, A., and ter Haar Romeny, B. M. (2018). Curvature integration in a 5d kernel for extracting vessel connections in retinal images. *IEEE Trans Image Process*, 27:606–621.

Alvarez, L., Lions, P.-L., and Morel, J.-M. (1992). Image selective smoothing and edge detection by nonlinear diffusion. ii. *SIAM Journal on Numerical Analysis*, 29(3):845–866.

Antolík, J., Hofer, S. B., Bednar, J. A., and Mrsic-Flogel, T. D. (2016). Model constrained by visual hierarchy improves prediction of neural responses to natural scenes. *PLOS Computational Biology*, 12(6):1–22.

Barbieri, D., Citti, G., and Sarti, A. (2014a). How uncertainty bounds the shape index of simple cells. *The Journal of Mathematical Neuroscience*, 4(1):5.

Barbieri, D., Cocci, G., Citti, G., and Sarti, A. (2014b). A cortical-inspired geometry for contour perception and motion integration. *J Math Imaging Vis*, 49(3):511–529.

Barnard, E. and Casasent, D. (1991). Invariance and neural nets. *IEEE Trans Neural Netw*, 2(5):498–508.

Baspinar, E. (2021). Multi-frequency image completion via a biologically-inspired sub-riemannian model with frequency and phase. *Journal of Imaging*, 7(12).

Bekkers, E. J., Duits, R., Mashtakov, A., and Sachkov, Y. (2017). Vessel tracking via sub-riemannian geodesics on the projective line bundle. In Nielsen, F.

and Barbaresco, F., editors, *Geometric Science of Information*, pages 773–781. Springer International Publishing.

Bennequin, D. (2014). Remarks on invariance in the primary visual systems of mammals.

Bertalmío, M., Calatroni, L., Franceschi, V., Franceschiello, B., and Prandi, D. (2019). A cortical-inspired model for orientation-dependent contrast perception: A link with wilson-cowan equations. In Lellmann, J., Burger, M., and Modersitzki, J., editors, *Scale Space and Variational Methods in Computer Vision*, pages 472–484, Cham. Springer International Publishing.

Bertalmío, M., Calatroni, L., Franceschi, V., Franceschiello, B., and Prandi, D. (2021). Cortical-inspired wilson–cowan-type equations for orientation-dependent contrast perception modelling. *Journal of Mathematical Imaging and Vision*, 63(2):263–281.

Bertoni, F., Citti, G., and Sarti, A. (2022). Lgn-cnn: A biologically inspired cnn architecture. *Neural Networks*, 145:42–55.

Bertoni, F., Montobbio, N., Sarti, A., and Citti, G. (2021). Emergence of lie symmetries in functional architectures learned by cnns. *Frontiers in Computational Neuroscience*, 15.

Boscain, U., Gauthier, J., and Prandi, D. (2018). Image inpainting via a control-theoretical model of human vision. In *2018 IEEE 14th International Conference on Control and Automation (ICCA)*, pages 963–968.

Boscain, Ugo, Chertovskih, Roman, Gauthier, Jean-Paul, Prandi, Dario, and Remizov, Alexey (2018). Cortical-inspired image reconstruction via sub-riemannian geometry and hypoelliptic diffusion. *ESAIM: ProcS*, 64:37–53.

Bosking, W., Zhang, Y., Schoenfield, B., and Fitzpatrick, D. (1997). Orientation selectivity and the arrangement of horizontal connections in tree shrew striate cortex. *J Neurosci*, 17:2112–2127.

Brainard, D. H. and Wandell, B. A. (1986). Analysis of the retinex theory of color vision. *J Opt Soc Am A*, 3(10):1651–1661.

Bressloff, P. and Cowan, J. D. (2002). The visual cortex as a crystal. *Physica D Nonlinear Phenomena*, 173:226–258.

Bressloff, P. C. and Cowan, J. D. (2003). The functional geometry of local and long-range connections in a model of V1. *J Physiol Paris*, 97(2-3):221–236.

Bressloff, P. C., Cowan, J. D., Golubitsky, M., Thomas, P. J., and Wiener, M. C. (2002). What Geometric Visual Hallucinations Tell Us about the Visual Cortex. *Neural Computation*, 14(3):473–491.

Brezis, H. (2010). *Functional Analysis, Sobolev Spaces and Partial Differential Equations*. Universitext. Springer New York.

Carandini, M. and Heeger, D. J. (2011). Normalization as a canonical neural computation. *Nat Rev Neurosci*, 13(1):51–62.

Citti, G. and Sarti, A. (2006). A cortical based model of perceptual completion in the roto-translation space. *Journal of Mathematical Imaging and Vision archive*, 24:307–326.

Citti, G. and Sarti, A. (2013). A gauge field model of modal completion. *Journal of Mathematical Imaging and Vision*, 52.

Coates, A., Lee, H., and Ng, A. (2011). An analysis of single-layer networks in unsupervised feature learning. pages 1–9.

Cohen, T., Geiger, M., and Weiler, M. (2020). A general theory of equivariant cnns on homogeneous spaces.

Cohen, T. S. and Welling, M. (2016). Group equivariant convolutional networks.

Cucker, F. and Smale, S. (2001). On the mathematical foundations of learning. *BULLETIN*, 39.

Daugman, J. G. (1985). Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *J Opt Soc Am A2*, 2:1160–1169.

DeAngelis, G. C., Ohzawa, I., and Freeman, R. D. (1995). Receptive-field dynamics in the central visual pathways. *Trends in Neurosciences*, 18(10):451–458.

Delsolneux, A., Moisan, L., and Morel, J.-M. (2008). *From Gestalt Theory to Image Analysis: A Probabilistic Approach*, volume 34.

Deng, L. (2012). The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142.

Dieleman, S., Fauw, J., and Kavukcuoglu, K. (2016). Exploiting cyclic symmetry in convolutional neural networks.

Dieleman, S., Willett, K. W., and Dambre, J. (2015). Rotation-invariant convolutional neural networks for galaxy morphology prediction. *Monthly Notices of the Royal Astronomical Society*, 450(2):1441–1459.

Dobbins, A., Zucker, S. W., and Cynader, M. S. (1987). Endstopped neurons in the visual cortex as a substrate for calculating curvature. *Nature*, 329(6138):438–441.

Duits, R. and Franken, E. (2010a). Left-invariant parabolic evolutions on se(2) and contour enhancement via invertible orientation scores part i: Linear left-invariant diffusion equations on se(2). *Quarterly of Applied Mathematics*, 68.

Duits, R. and Franken, E. (2010b). Left-invariant parabolic evolutions on se(2) and contour enhancement via invertible orientation scores part ii: Nonlinear left-invariant diffusions on invertible orientation scores. *Quarterly of Applied Mathematics*, 68(2):293–331.

E. R. Kandel, J. H. S. and Jessell., T. M. (1993). Principles of neural science, third edition. *Human Psychopharmacology: Clinical and Experimental*, 8(4):294–294.

Enroth-Cugell, C. and Robson, J. G. (1966). The contrast sensitivity of retinal ganglion cells of the cat. *The Journal of Physiology*, 187(3):517–552.

Ermentrout, G. B. and Cowan, J. D. (1980). Large scale spatially organized activity in neural nets. *SIAM Journal on Applied Mathematics*, 38(1):1–21.

Espinosa, J. S. and Stryker, M. P. (2012). Development and plasticity of the primary visual cortex. *Neuron*, 75(2):230–249.

Fasel, B. and Gatica-Perez, D. (2006). Rotation-invariant neoperceptron. volume 3, pages 336 – 339.

Favali, M. (2017). *Formal models of visual perception based on cortical architectures*. PhD thesis.

Favali, M., Citti, G., and Sarti, A. (2017). Local and Global Gestalt Laws: A Neurally Based Spectral Approach. *Neural Computation*, 29(2):394–422.

Ferraro, M. and Caelli, T. (1994). Lie transformation groups, integral transforms, and invariant pattern recognition. *Spatial vision*, 8(1):33—44.

Field, D. J., Hayes, A., and Hess, R. F. (1993). Contour integration by the human visual system: evidence for a local association field. *Vision Res*, 33:173–193.

Galyaev, I. and Mashtakov, A. (2021). Liouville integrability in a four-dimensional model of the visual cortex. *Journal of Imaging*, 7:277.

Gens, R. and Domingos, P. M. (2014). Deep symmetry networks. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Gidas, B., Ni, W., and Nirenberg, L. (1981). Symmetry of positive solutions of nonlinear elliptic equations in r.

Girosi, F., Jones, M., and Poggio, T. (1995). Regularization Theory and Neural Networks Architectures. *Neural Computation*, 7(2):219–269.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition.

Higham, C. F. and Higham, D. J. (2018). Deep learning: An introduction for applied mathematicians.

Hoffman, W. (1989). The visual cortex is a contact bundle. *Appl Math Comput*, 32:137–167.

Hubel, D. H. (1987). *Eye, brain, and vision*. New York, WH Freeman (Scientific American Library).

Hubel, D. H. and Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat visual cortex. *J Physiol (London)*, 160:106–154.

Hubel, D. H. and Wiesel, T. N. (1977). Ferrier lecture: Functional architecture of macaque monkey visual cortex. *Proceedings of the Royal Society of London. Series B, Biological Sciences*, 198(1130):1–59.

Im, M. and Fried, S. I. (2015). Indirect activation elicits strong correlations between light and electrical responses in on but not off retinal ganglion cells. *The Journal of Physiology*, 593(16):3577–3596.

Ivakhnenko, A., Ivakhnenko, A., Lapa, V., LAPA, V., Lapa, V., and McDonough, R. (1967). *Cybernetics and Forecasting Techniques*. Modern analytic and computational methods in science and mathematics. American Elsevier Publishing Company.

Jumakulyyev, I. and Schultz, T. (2021). *Fourth-Order Anisotropic Diffusion for Inpainting and Image Compression*, pages 99–124.

Kanizsa, G., Legrenzi, P., and Bozzi, P. (1979). Organization in vision: Essays on gestalt perception.

Kimmel, R., Elad, M., Shaked, D., Keshet, R., and Sobel, I. (2003). A variational framework for retinex. *International Journal of Computer Vision*, 52(1):7–23.

Koenderink, J. J. and van Doom, A. J. (1987). Representation of local geometry in the visual system. *Biol. Cybern.*, 55(6):367–375.

Land, E. H. (1964). The retinex. *American Scientist*, 52(2):247–264.

Land, E. H. and McCann, J. J. (1971). Lightness and retinex theory. *J Opt Soc Am*, 61(1):1–11.

Laptev, D., Savinov, N., Buhmann, J. M., and Pollefeys, M. (2016). Ti-pooling: Transformation-invariant pooling for feature learning in convolutional neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 289–297.

Lawrence, S., Giles, C. L., Tsoi, A. C., and Back, A. D. (1997). Face recognition: a convolutional neural-network approach. *IEEE Transactions on Neural Networks*, 8:98–113.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, volume 86, pages 2278–2324.

LeCun, Y. and Cortes, C. (2010). MNIST handwritten digit database.

Lee, T. S. (1996). Image representation using 2d Gabor wavelets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18.

Lei, L., Zhou, Y., and Li, J. (2007). An investigation of retinex algorithms for image enhancement. *Journal of Electronics (China)*, 24(5):696–700.

Liang, M. and Hu, X. (2015). Recurrent convolutional neural network for object recognition. *CVPR*.

Limare, N., Petro, A.-B., Sbert, C., and Morel, J.-M. (2011). Retinex poisson equation: a model for color perception. *Image Processing On Line*, 1.

Lopes, O. (1996). Radial symmetry of minimizers for some translation and rotation invariant functionals. *Journal of Differential Equations*, 124(2):378–388.

Marcos, D., Volpi, M., and Tuia, D. (2016). Learning rotation invariant convolutional filters for texture classification. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2012–2017.

Mikusiński, J. (1978). *The Bochner Integral*. Lehrbücher und Monographien aus dem Gebiete der exakten Wissenschaften: Mathematische Reihe. Academic Press.

Montobbio, N., Bonnasse-Gahot, L., Citti, G., and Sarti, A. (2019a). Kercnns: biologically inspired lateral connections for classification of corrupted images. *CoRR*, abs/1910.08336.

Montobbio, N., Citti, G., and Sarti, A. (2019b). From receptive profiles to a metric model of v1. *Journal of Computational Neuroscience*, 46(3):257–277.

Montobbio, N., Sarti, A., and Citti, G. (2020). A metric model for the functional architecture of the visual cortex. *Journal on Applied Mathematics*, 80:1057–1081.

Morel, J. M., Petro, A. B., and Sbert, C. (2010). A pde formalization of retinex theory. *IEEE Transactions on Image Processing*, 19(11):2825–2837.

Mumford, D. (1994). *Elastica and Computer Vision*, pages 491–506. Springer New York, New York, NY.

Mumford, D. and Shah, J. (1989). Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on Pure and Applied Mathematics*, 42:577–685.

Nolte, J. and Sundsten, J. (2002). *The Human Brain: An Introduction to Its Functional Anatomy*. Number No. 798 in Human Brain. Mosby.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch. In *NIPS-W*.

Perona, P. and Freeman, W. (1998). A factorization approach to grouping. In Burkhardt, H. and Neumann, B., editors, *Computer Vision — ECCV'98*, pages 655–670, Berlin, Heidelberg. Springer Berlin Heidelberg.

Petitot, J. (2003). The neurogeometry of pinwheels as a sub-riemannian contact structure. *J Physiol Paris*, 97(2-3):265–309.

Petitot, J. (2008). *Neurogéométrie de la vision - Modèles mathématiques et physiques des architectures fonctionnelles*. Éditions de l'École Polytechnique.

Petitot, J. (2017). *Elements of Neurogeometry: Functional Architectures of Vision*.

Petitot, J. and Tondut, Y. (1999). Vers une neuro-géométrie. Fibrations corticales, structures de contact et contours subjectifs modaux. In *Mathématiques , Informatique et Sciences Humaines*, volume 145, pages 5–101. CAMS, EHESS.

Pham, Q.-C. and Bennequin, D. (2012). Affine invariance of human hand movements: a direct test. *arXiv: Other Quantitative Biology*.

Pregowska, A., Casti, A., Kaplan, E., Wajnryb, E., and Szczepanski, J. (2019). Information processing in the lgn: a comparison of neural codes and cell types. *Biol Cybern*, 113(4):453–464.

Provenzi, E., Carli, L. D., Rizzi, A., and Marini, D. (2005). Mathematical definition and analysis of the retinex algorithm. *J. Opt. Soc. Am. A*, 22(12):2613–2621.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788.

Reinagel, P. and Reid, R. C. (2000). Temporal coding of visual information in the thalamus. *J Neurosci*, 20(14):5392–5400.

Ren, S., He, K., Girshick, R., and Sun, J. (2017). Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Trans Pattern Anal Mach Intell*, 39(6):1137–1149.

Ringach, D. L. (2002). Spatial structure and symmetry of simple-cell receptive fields in macaque primary visual cortex. *J Neurophysiol*, 88(1):455–463.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychol Rev*, 65(6):386–408.

Rosenblatt, F. (1962). *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Cornell Aeronautical Laboratory. Report no. VG-1196-G-8. Spartan Books.

Sanguinetti, G., Citti, G., and Sarti, A. (2010). A model of natural image edge co-occurrence in the rototranslation group. *J Vis*, 10.

Sarti, A. and Citti, G. (2011). On the origin and nature of neurogeometry. *La Nuova Critica*.

Sarti, A. and Citti, G. (2015). The constitution of visual perceptual units in the functional architecture of V1. *Journal of Computational Neuroscience*, 38:285–300.

Sarti, A., Citti, G., and Manfredini, M. (2003). From neural oscillations to variational problems in the visual cortex. *Journal of Physiology-Paris*, 97(2):379–385. Neurogeometry and visual perception.

Sarti, A., Citti, G., and Petitot, J. (2008). The symplectic structure of the primary visual cortex. *Biol. Cybern.*, 98(1):33–48.

Semeniuta, S., Severyn, A., and Barth, E. (2016). Recurrent dropout without memory loss. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1757–1766, Osaka, Japan. The COLING 2016 Organizing Committee.

Serre, T., Wolf, L., Bileschi, S., Riesenhuber, M., and Poggio, T. (2007). Robust object recognition with cortex-like mechanisms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):411–426.

Sherstinsky, A. (2020). Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306.

Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition.

Solomon, S. G., Lee, B. B., and Sun, H. (2006). Suppressive surrounds and contrast gain in magnocellular-pathway retinal ganglion cells of macaque. *J Neurosci*, 26(34):8715–8726.

Spoerer, C., McClure, P., and Kriegeskorte, N. (2017). Recurrent convolutional neural networks: a better model of biological object recognition. *Frontiers in psychology*, 8:1551.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Stevens, J.-L. R., Law, J. S., Antolík, J., and Bednar, J. A. (2013). Mechanisms for stable, robust, and adaptive development of orientation maps in the primary visual cortex. *Journal of Neuroscience*, 33(40):15747–15766.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2014). Going deeper with convolutions. cite arxiv:1409.4842.

Ts'o, D. Y., Gilbert, C. D., and Wiesel, T. N. (1986). Relationships between horizontal interactions and functional architecture in cat striate cortex as revealed by cross-correlation analysis. *J Neurosci*, 6(4):1160–1170.

Uglesich, R., Casti, A., Hayot, F., and Kaplan, E. (2009). Stimulus size dependence of information transfer from retina to thalamus. *Front Syst Neurosci*, 3:10.

Valberg, A. and Seim, T. (2013). Neurophysiological correlates of color vision: A model. *Psychology & Neuroscience*, 6:213–218.

Weiss, Y. (1999). Segmentation using eigenvectors: a unifying view. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 975–982 vol.2.

Williams, L. R. and Jacobs, D. W. (1997). Stochastic Completion Fields: A Neural Model of Illusory Contour Shape and Salience. *Neural Computation*, 9(4):837–858.

Wilson, H. R. and Cowan, J. D. (1972). Excitatory and inhibitory interactions in localized populations of model neurons. *Biophys J*, 12(1):1–24.

Wu, F., Hu, P., and Kong, D. (2015). Flip-rotate-pooling convolution and split dropout on convolution neural networks for image classification.

Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.

Yamins, D., Hong, H., Cadieu, C., and Dicarlo, J. (2015). Hierarchical modular optimization of convolutional networks achieves representations similar to macaque it and human ventral stream. *Advances in Neural Information Processing Systems*.

Yamins, D. L. K. and DiCarlo, J. J. (2016). Using goal-driven deep learning models to understand sensory cortex. *Nature Neuroscience*, 19:356 – 365.

Yeonan-Kim, J. and Bertalmío, M. (2017). Analysis of retinal and cortical components of Retinex algorithms. *Journal of Electronic Imaging*, 26:031208.

Yosida, K. (1995). *Functional Analysis*. Springer-Verlag Berlin Heidelberg.

Yu, S., Xie, J., Hao, J., Zheng, Y., Zhang, J., Hu, Y., Liu, J., and Zhao, Y. (2021). 3d vessel reconstruction in oct-angiography via depth map estimation. In *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pages 1609–1613.

Zaghloul, K. A., Boahen, K., and Demb, J. B. (2003). Different circuits for on and off retinal ganglion cells cause different contrast sensitivities. *Journal of Neuroscience*, 23(7):2645–2654.

Zhang, J., Dashtbozorg, B., Bekkers, E., Pluim, J. P. W., Duits, R., and ter Haar Romeny, B. M. (2016). Robust retinal vessel segmentation via locally adaptive derivative frames in orientation scores. *IEEE Transactions on Medical Imaging*, 35(12):2631–2644.

Zhang, Y., Lee, T. S., Li, M., Liu, F., and Tang, S. (2019). Convolutional neural network models of v1 responses to complex patterns. *J Comput Neurosci*, 46(1):33–54.