A Complex Wavelet Approach for Shift-Invariant Convolutional Neural Networks

Hubert Leterme

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

École doctorale : MSTII - Mathématiques, Sciences et technologies de l'information, Informatique
Spécialité : Mathématiques et Informatique
Unité de recherche : Laboratoire Jean Kuntzmann

Ondelettes Complexes pour des Réseaux de Neurones Convolutifs Invariants par Translation

A Complex Wavelet Approach for Shift-Invariant Convolutional Neural Networks

Présentée par :

Hubert LETERME

Direction de thèse :

Valérie PERRIER Professeure des Universités, GRENOBLE INP - UGA	Directrice de thèse
Karteek ALAHARI Chargé de recherche HDR, INRIA CENTRE GRENOBLE-RHONE-ALPES	Co-directeur de thèse
Kévin POLISANO Chargé de recherche, CNRS	Co-encadrant de thèse

Rapporteurs :

Nelly PUSTELNIK
Chargée de recherche HDR, CNRS

François MALGOUYRES
Professeur des universités, UNIVERSITÉ TOULOUSE III - PAUL SABATIER

Thèse soutenue publiquement le **14 juin 2023**, devant le jury composé de :

Valérie PERRIER Professeure des Universités, GRENOBLE INP - UGA	Directrice de thèse
Nelly PUSTELNIK Chargée de recherche HDR, CNRS	Rapporteure
François MALGOUYRES Professeur des universités, UNIVERSITÉ TOULOUSE III - PAUL SABATIER	Rapporteur
Massih-Reza AMINI Professeur des Universités, UNIVERSITÉ GRENOBLE ALPES	Président
Joan BRUNA Professeur associé, NEW YORK UNIVERSITY	Examinateur
Gabriel PEYRÉ Directeur de recherche, CNRS	Examinateur

Invités :

Karteek ALAHARI
Chargé de recherche HDR, INRIA CENTRE GRENOBLE-RHONE-ALPES

Kévin POLISANO
Chargé de recherche, CNRS

Abstract

Despite significant advancements in computer vision over the past decade, convolutional neural networks (CNNs) still suffer from a lack of mathematical understanding. In particular, stability properties with respect to small transformations such as translations, rotations, scaling or deformations are only partially understood. While there is a broad literature on this topic, some gaps remain, specifically with regard to the combined effect of convolution and max pooling layers in producing near shift-invariant feature representations. This property is of utmost importance for classification, since two shifted versions of a single input image are expected to receive the same label.

It is well-known that subsampled convolutions with band-pass filters are prone to producing unstable image representations when inputs are shifted by a few pixels. The first contribution of this thesis consists in proving that a nonlinear max pooling operator can partially restore shift invariance. By applying results from the wavelet theory, and adopting a probabilistic point of view, we reveal a similarity between the max pooling of real-valued convolutions, as implemented in conventional architectures, and the modulus of complex-valued convolutions, for which a measure of shift invariance is established.

However, for specific filter frequencies, this similarity is lost, and CNNs become unstable to translations. This phenomenon, known as aliasing, can be avoided by employing additional low-pass filters in strategic locations of the network architecture, as several authors have done in recent years. While their methods effectively increase both shift invariance and prediction accuracy, they come at the cost of significant loss of high-frequency information. As a second contribution, we present a novel antialiasing method which, unlike previous methods, preserves this information. Relying on our theoretical study, the key idea is to exploit the properties of complex convolutions to guarantee near-shift invariance for any filter frequency. By adding an imaginary part to high-frequency kernels and replacing the max pooling layer with a simple modulus operator, we empirically evidence an increase in the network's stability and a lower error rate compared to previous approaches based on low-pass filtering.

In conclusion, the aim of this thesis is twofold: improving the mathematical understanding of CNNs from the perspective of shift invariance, and improving the tradeoff between stability and information preserving, based on our theoretical contribution which is grounded in wavelet theory. Our findings thus have the potential to positively impact various applications of computer vision, especially in fields that require theoretical guarantees.

Résumé

Malgré des progrès spectaculaires en vision par ordinateur au cours de la dernière décennie, les réseaux de neurones convolutifs (CNN) souffrent toujours d'un faible niveau de compréhension mathématique. En particulier, les propriétés de stabilité vis-à-vis de petites transformations (translations, rotations, mises à l'échelle, déformations) ne sont que partiellement comprises. Bien qu'il existe une vaste littérature sur ce sujet, certaines lacunes subsistent, notamment concernant l'effet combiné des couches de convolution et de max pooling dans la génération de représentations quasi-invariantes. Cette propriété est primordiale pour la classification, puisqu'il est attendu que deux versions translatées d'une même image soient classifiées de manière identique.

Les convolutions sous-échantillonnées avec des filtres passe-bande sont connues pour produire des représentations instables lorsque les images en entrée sont translatées de quelques pixels. La première contribution de cette thèse consiste à prouver qu'un opérateur non linéaire de max pooling est susceptible de partiellement restaurer l'invariance par translation. En appliquant des résultats issus de la théorie des ondelettes, et en adoptant un point de vue probabiliste, nous révélons une similitude entre le max pooling de convolutions à valeurs réelles, tel qu'implémenté dans les architectures conventionnelles, et le module de convolutions à valeurs complexes, pour lequel une mesure d'invariance par translation est établie.

Cependant, pour certaines fréquences de filtre, une telle similitude ne se vérifie pas et les CNN deviennent instables face aux petites translations. Ce phénomène, connu sous le nom d'aliasing, peut être évité en appliquant des filtres passe-bas supplémentaires à des endroits stratégiques du réseau, comme plusieurs auteurs l'ont proposé au cours des dernières années. Ces méthodes, bien qu'elles améliorent sensiblement l'invariance par translation et la qualité des prédictions, impliquent une perte importante d'informations de haute fréquence. Comme seconde contribution, nous présentons une nouvelle méthode d'antialiasing qui, contrairement aux précédentes, préserve cette information. En s'appuyant sur notre étude théorique, l'idée clé est d'exploiter les propriétés des convolutions complexes pour garantir une quasi-invariance par translation quelle que soit la fréquence du filtre. En ajoutant une partie imaginaire aux filtres de haute fréquence et en remplaçant l'opérateur de max pooling par un simple module, nous mettons empiriquement en évidence une augmentation de la stabilité du réseau et un taux d'erreur plus faible par rapport aux approches précédentes basées sur des filtres passe-bas.

En conclusion, l'objectif de cette thèse est double : améliorer la compréhension mathématique des CNN en termes d'invariance par translation, et améliorer le compromis entre stabilité et préservation de l'information, sur la base de notre contribution théorique fondée sur la théorie des ondelettes. Ces travaux ont donc le potentiel d'impacter positivement diverses applications de la vision par ordinateur, en particulier dans les domaines nécessitant des garanties théoriques.

Remerciements

Tout d'abord, je remercie chaleureusement ma directrice de thèse, Valérie, ainsi que mon co-directeur, Karteek, pour m'avoir accompagné avec bienveillance et exigence tout au long de ces années de doctorat. Un grand merci également à Kévin pour nos longues séances de réflexion et de discussion, tant sur le plan scientifique que philosophique ! Vos conseils éclairés, votre disponibilité et votre complémentarité ont été des atouts essentiels qui m'ont permis de mener à bien ce projet.

Je tiens à remercier Nelly Pustelnik et François Malgouyres d'avoir accepté de rapporter cette thèse, pour leurs retours pertinents et les discussions constructives qui ont suivi. Je souhaite par ailleurs remercier les autres membres du jury, à commencer par Massih-Reza Amini pour avoir accepté de le présider. Je n'oublie pas que tu as été mon premier point de contact lorsque j'ai décidé de reprendre des études universitaires. Mes sincères remerciements vont également à Gabriel Peyré pour son accompagnement lors des comités de suivi individuels et ses précieux conseils, ainsi qu'à Joan Bruna, dont les travaux inspirants ont guidé ma recherche.

Cette thèse est le fruit d'un long processus de reconversion professionnelle, après avoir passé plusieurs années en tant qu'ingénieur et analyste des données. Je tiens à exprimer ma reconnaissance envers Laurent et Asma, qui ont joué un rôle essentiel en me montrant que cette reconversion était possible et en m'aidant à cheminer en ce sens.

Un tel projet n'aurait assurément pas pu être réalisé sans l'ambiance chaleureuse qui règne au laboratoire. Je remercie ainsi mes collègues et ami-e-s de l'Imag et de l'Inria pour tous ces instants de discussion et de détente qui ont probablement eu raison de la machine à café ! En particulier, merci à mes chers co-bureau : Anatole, Carlos et Yu-Guan ; à mes compagnons du Diderot : Manon, Nils, Sélim, Margaux, Dima, Sylvain, Sergeï, David, Alexandre, Kajal, Chi, Yunjiao ; sans oublier bien-sûr Gilles, Thibault, JB, Flora, Waïss, Victor, Yassine, Benji, Amine, Gabriel, Juliette, Zhiqi, et toutes les personnes que j'ai pu oublier. Merci également au personnel administratif du laboratoire, avec une mention spéciale pour Laurence.

Une autre composante essentielle de ma vie grenobloise a été le Chœur Universitaire, qui m'a offert une véritable bouffée d'oxygène, des moments riches en émotion et de belles amitiés. Je souhaite donc remercier toutes les personnes avec qui j'ai partagé cette aventure musicale. Par ailleurs, ces années de préparation au doctorat ont été marquées par de longues périodes d'isolement en raison de la pandémie de Covid-19. Heureusement, j'ai pu compter, tant dans ces moments difficiles que lors de retrouvailles, sur la présence indéfectible de mes amis de longue date. Un immense merci à François, Alexandre, Benjamin, Sandra, Anne-Laure, Maxence, Pierre-Louis, Thibaut, Camille, Arthur, Sylvain, Thibaut, et bien d'autres, pour tous ces moments partagés, dans la vie réelle comme par écran interposé. J'ai une pensée particulière pour Carolina, qui nous a quittés bien trop tôt.

Pour finir, j'exprime ma profonde gratitude envers ma famille, qui m'a soutenu de façon inconditionnelle pendant toutes ces années. Envers mes parents, Annie et Emmanuel, qui ont été présents à chaque étape de mon parcours professionnel et ont souvent été en première ligne lors de mes moments de doute et de découragement. Le rôle que vous avez joué est inestimable. Envers mes frères et sœurs, Anne-Claire, Pierre-Hugues et Bénédicte, et leurs moitiés. Envers mes oncles et tantes, en particulier Brigitte et Gabriel, ainsi que mes cousins et cousines. Envers mes formidables mamies, Henriette et Agnès, à qui je dédie cette thèse.

Acknowledgments

This work has been partially supported by the LabEx PERSYVAL-Lab (ANR-11-LABX-0025-01) funded by the French program Investissement d'avenir, as well as the ANR grant MIAI (ANR-19-P3IA-0003). Most of the computations presented in this thesis were performed using the GRICAD infrastructure (https://gricad.univ-grenoble-alpes.fr), which is supported by Grenoble research communities.

Contents

List of Figures

List of Tables

List of Symbols

$\{a\,..\,b\} \subset \mathbb{N}$	Set of integers ranging from a to b
$x \in \mathbb{R}$	Scalar
$\boldsymbol{x} \in \mathbb{R}^D$	D-dimensional vector
$x_d \in \mathbb{R}$	d-th element of vector $\boldsymbol{x}$, for any $d \in \{0\,..\,D-1\}$
$\boldsymbol{X} \in \mathbb{R}^{N \times D}$	Matrix of size $N \times D$
$\boldsymbol{x}_n \in \mathbb{R}^D$	n-th row vector of $\boldsymbol{X}$, for any $n \in \{0\,..\,N-1\}$
$x_{nd} \in \mathbb{R}$	Element of coordinates (n, d) in matrix $\boldsymbol{X}$
$\varGamma : E \to \mathbb{R}$	Scalar-valued function
$\boldsymbol{\varGamma} : E \to \mathbb{R}^Q$	Vector-valued function
$\varGamma_q : E \to \mathbb{R}$	Function computing the q-th scalar output of $\boldsymbol{\varGamma}$
$\operatorname{argmax}(\boldsymbol{y}) \in \{0\,..\,Q-1\}$	Index of the (first) maximal element of $\boldsymbol{y} \in \mathbb{R}^Q$
$\|p\| \cdot \in \mathbb{R}_+$	l^p-norm, with $p > 0$
$\mathsf{X} : \Omega \to E$	Random variable with outcomes in E
$\mathbf{X} : \Omega \to \mathbb{R}^D$	Random vector with outcomes in $\mathbb{R}^D$
$f_{\mathsf{X}} : E \to \mathbb{R}_+$	Probability density function of X
$f_{(\mathsf{X}\,\mid\,\mathsf{Y}=y)} : E \to \mathbb{R}_+$	Cond. probability density of X given $\mathsf{Y} = y$
$\mathbb{P}\{\mathsf{X} = x\} \in [0,\,1]$	Probability of the event $\mathsf{X} = x$, with $x \in E$
$\mathbb{E}[\mathsf{X}] \in E$	Expected value of X
$\boldsymbol{x}^\top \boldsymbol{y}$ or $\langle \boldsymbol{x},\,\boldsymbol{y} \rangle \in \mathbb{R}$	Euclidean inner product between vectors $\boldsymbol{x}$ and $\boldsymbol{y}$
$\nabla_{\boldsymbol{V}} \tilde{\mathcal{L}} \in \mathbb{R}^{Q \times D}$	Gradient of $\tilde{\mathcal{L}}$ with respect to $\boldsymbol{V} \in \mathbb{R}^{Q \times D}$
$\overline{\boldsymbol{v}} \in \mathbb{R}^d$	Flipped version of $\boldsymbol{v} \in \mathbb{R}^d$
$\boldsymbol{x} * \overline{\boldsymbol{v}} \in \mathbb{R}^D$	Vector convolution between $\boldsymbol{x} \in \mathbb{R}^D$ and $\overline{\boldsymbol{v}} \in \mathbb{R}^d$
$\boldsymbol{y} + b \in \mathbb{R}^D$	Elementwise vector-scalar sum, $\boldsymbol{y} \in \mathbb{R}^D$, $b \in \mathbb{R}$
$\mathrm{x} \in l^2_{\mathbb{R}}(\mathbb{Z})$ or $l^2_{\mathbb{C}}(\mathbb{Z})$	Real- or complex-valued 1D sequence, with $\|\mathrm{x}\|_2^2 < \infty$

$\mathrm{x}[n] \in \mathbb{R}$ or $\mathbb{C}$	Element of $\mathrm{x} \in l^2_{\mathbb{R}}(\mathbb{Z})$ or $l^2_{\mathbb{C}}(\mathbb{Z})$ at position $n \in \mathbb{Z}$
$\overline{\mathrm{w}} \in l^2_{\mathbb{C}}(\mathbb{Z})$	Flipped version of $\mathrm{w} \in l^2_{\mathbb{C}}(\mathbb{Z})$
$\mathrm{x} * \overline{\mathrm{w}} \in l^2_{\mathbb{C}}(\mathbb{Z})$	Standard convolution between x and $\overline{\mathrm{w}} \in l^2_{\mathbb{C}}(\mathbb{Z})$
$\widehat{\mathrm{x}} \in L^2_{\mathbb{C}}([-\pi, \pi])$	Discrete-time Fourier transform of $\mathrm{x} \in l^2_{\mathbb{C}}(\mathbb{Z})$
$\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ or $l^2_{\mathbb{C}}(\mathbb{Z}^2)$	Real- or complex-valued 2D sequence, with $\|\mathrm{X}\|_2^2 < \infty$
$\mathrm{X}[\boldsymbol{n}] \in \mathbb{R}$ or $\mathbb{C}$	Element of $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ or $l^2_{\mathbb{C}}(\mathbb{Z}^2)$ at position $\boldsymbol{n} \in \mathbb{Z}^2$
$\overline{\mathrm{W}} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$	Flipped version of $\mathrm{W} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$
$\mathrm{X} * \overline{\mathrm{W}} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$	Standard convolution between X and $\overline{\mathrm{W}} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$
$\mathrm{X} \star \mathrm{W} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$	Cross-correlation between X and W, equal to $\mathrm{X} * \overline{\mathrm{W}}$
$\mathrm{X} \downarrow m \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ or $l^2_{\mathbb{C}}(\mathbb{Z}^2)$	Subsampling operation on $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ or $l^2_{\mathbb{C}}(\mathbb{Z}^2)$
$\widehat{\mathrm{X}} \in L^2_{\mathbb{C}}([-\pi, \pi]^2)$	Discrete-time Fourier transform of $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$
$\mathbf{X} \in \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^K$	Multichannel sequence, with $K \in \mathbb{N} \setminus \{0\}$
$\mathrm{X}_k \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$	k-th channel of $\mathbf{X}$, for any $k \in \{0\,..\,K-1\}$
$\mathit{\Gamma} : l^2_{\mathbb{R}}(\mathbb{Z}^d) \to l^2_{\mathbb{R}}(\mathbb{Z}^d)$	Functions defined on d-dimensional sequences
$\boldsymbol{\mathit{\Gamma}} : (l^2_{\mathbb{R}}(\mathbb{Z}^d))^K \to (l^2_{\mathbb{R}}(\mathbb{Z}^d))^L$	Functions defined on multichannel sequences
$\mathit{\Gamma}_l : (l^2_{\mathbb{R}}(\mathbb{Z}^d))^K \to l^2_{\mathbb{R}}(\mathbb{Z}^d)$	Functions computing the l-th output of $\boldsymbol{\mathit{\Gamma}}$
$F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ or $L^2_{\mathbb{C}}(\mathbb{R}^2)$	Real or complex 2D function, with $\|F\|_{L^2}^2 < \infty$
$\mathit{\Phi} \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ or $L^2_{\mathbb{C}}(\mathbb{R}^2)$	2D scaling function
$\mathit{\Psi} \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ or $L^2_{\mathbb{C}}(\mathbb{R}^2)$	2D wavelet
$\overline{\mathit{\Psi}} \in L^2_{\mathbb{C}}(\mathbb{R}^2)$	Flipped version of $\mathit{\Psi} \in L^2_{\mathbb{C}}(\mathbb{R}^2)$
$F * \overline{\mathit{\Psi}} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$	Standard convolution between F and $\overline{\mathit{\Psi}} \in L^2_{\mathbb{C}}(\mathbb{R}^2)$
$\widehat{F} \in L^2_{\mathbb{C}}(\mathbb{R}^2)$	Fourier transform of $F \in L^2_{\mathbb{C}}(\mathbb{R}^2)$
$\delta_{\boldsymbol{np}} \in \{0\,..\,1\}$	Kronecker delta symbol
$\mathbb{1} : \mathbb{Z}^2 \to \{0\,..\,1\}$	Characteristic function on $\mathbb{Z}^2$
$\nabla \boldsymbol{\tau} : \mathbb{R}^2 \to \mathbb{R}^{2\times 2}$	Jacobian of operator $\boldsymbol{\tau} : \mathbb{R}^2 \to \mathbb{R}^2$

Chapter 1

Introduction

It has been over a decade since convolutional neural networks (CNNs) overtook other machine learning frameworks in large visual recognition tasks, when Krizhevsky et al. (2017) won the 2012 edition of the ILSVRC challenge on image classification (Russakovsky et al., 2015). CNNs then became increasingly popular for many computer vision tasks, including classification (Szegedy et al., 2015; He et al., 2016), object detection (Girshick, 2015; Ren et al., 2015), generative models (Goodfellow et al., 2014), pose estimation (Toshev and Szegedy, 2014), *etc.*

CNNs rely on convolutions and nonlinear pooling operations to transform input images into high-level feature vectors, which are in turn processed for the task at hand. In the context of image classification, which is the main focus of this thesis, the feature vectors are fed into a linear classifier. This type of architecture was popularized by LeCun et al. (1989) as a subfamily of feedforward neural networks. Their work introduced, through the use of convolution layers, prior assumptions about the network's desired behavior. Specifically, nearby pixels in an image are more strongly correlated, and the same weights are used for different parts of the image. The purpose of employing such *inductive bias* is to avoid having to learn these properties from data. The model therefore becomes more specialized, making it more efficient in processing images with reduced complexity, while improving its generalization to unseen data.

In order to achieve high classification accuracy, a convolutional network is expected to retain discriminative image components while reducing intra-class variability (LeCun et al., 1998; Bruna and Mallat, 2013). A key property that is often desired in CNNs is their ability to remain invariant to small input transformations, such as translations, rotations, distortions, or scaling (Liao and Peng, 2010; Sifre and Mallat, 2013; Bruna and Mallat, 2013; Bietti and Mairal, 2017; Wiatowski and Bölcskei, 2018). Since perfect invariance is seldom achieved, we shall also use the term *stability* to refer to their near-invariance behavior. This thesis primarily targets translations, also called shifts. Furthermore, we focus on a configuration that is commonly observed in CNNs when trained on image datasets: many convolution kernels in the first layer resemble band-pass oriented waveforms, also known as *Gabor-like filters* (Yosinski et al., 2014; Rai and Rivas, 2020). This phenomenon is illustrated in Figure 1.1 in the case of AlexNet: approximately half of the kernels exhibit oscillating patterns at various frequencies and orientations.

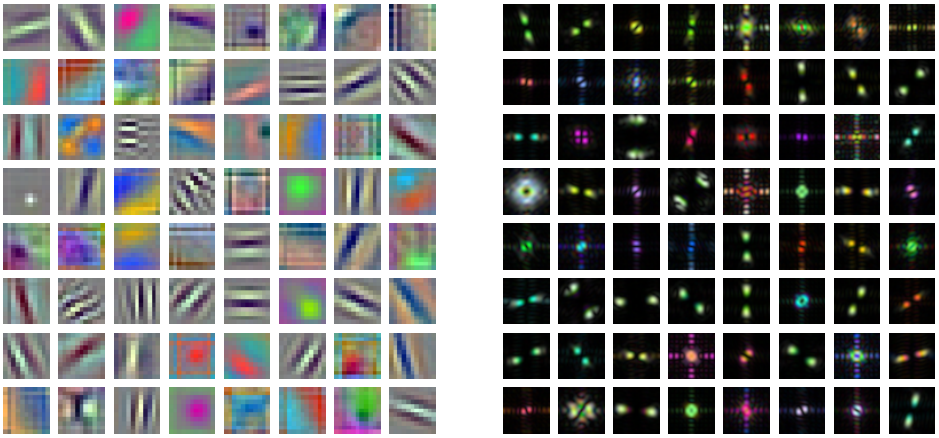


Figure 1.1. Spatial (left) and Fourier (right) representations of convolution kernels in the first layer of AlexNet, after training with ImageNet ILSVRC 2012-2017 (Russakovsky et al., 2015). Each kernel connects the 3 RGB input channels to one of the 64 output channels.

1.1 Themes and Challenges

This thesis is articulated around three main themes: (1) identifying and analyzing the invariance properties of CNNs; (2) introducing inductive bias in the network structure through oriented band-pass filters to reduce model complexity; (3) improving the shift invariance of CNNs to enhance their performance. This section presents a brief overview of the literature on these topics, and the unsolved questions that we tackled. Then, a summary of our main contributions is provided in Section 1.2.

1.1.1 Studying Invariance Properties in CNNs

Analyzing the invariance properties of CNNs is critical as it enables to identify their shortcomings and provides an opportunity to enhance their performance. In recent years, several works focused on this topic. Most notably, Bruna and Mallat (2013) developed a family CNN-like architectures, named *wavelet scattering networks* (ScatterNets), based on a succession of complex convolutions with wavelet filters followed by nonlinear modulus pooling. They established stability properties of these models with respect to translations and distortions. Further studies of this family of CNNs have been conducted by Mallat (2012), Mallat and Waldspurger (2015), Mallat (2016), Waldspurger (2016), Gama et al. (2019), Czaja and W. Li (2019), Perlmutter et al. (2020), Zarka et al. (2020), and D. Zou and Lerman (2020), sometimes using modified versions of the architecture. We refer the reader to Section 3.5.2 for more details on this topic. As deep learning architectures with well-established mathematical properties, ScatterNets are sometimes used as explanatory models for standard, freely-trained networks. However, whether their properties are transferable to a broader class of models is unclear, because the former rely on complex-valued convolutions whereas the latter exclusively employ real-valued kernels. Moreover, the modulus operator is used as an activation and pooling layer in ScatterNets, whereas standard CNNs implement pointwise nonlinear operators such as ReLU and spatial pooling layers such as max pooling. This limitation has been pointed out by Tygert et al. (2016) as an argument in favor of complex-valued CNNs.

Another work established asymptotic stability properties for a wide range of deep learning architectures in the continuous domain (Wiatowski and Bölcskei, 2018). However, these results do not fully extend to the discrete framework due to aliasing effects. Aliasing occurs when a signal is sampled at a rate that is too low to accurately capture its high-frequency components. Consequently, subsampled convolutions with band-pass real-valued filters can introduce aliasing artifacts, which results in instability to translations (Azulay and Weiss, 2019; R. Zhang, 2019). This limitation can penalize the network's accuracy and generalization capability. This kind of filter is very common in CNNs when trained on image datasets (Yosinski et al., 2014; Rai and Rivas, 2020): many convolution kernels exhibit Gabor-like patterns with well-defined frequencies and orientations, as illustrated in Figure 1.1. Such filters produce sparse image representations that are able to discriminate features by scale and orientation, detecting basic geometric shapes such as edges or textures.

Another line of work is focused on modeling and studying CNNs from the point of view of convolutional kernel networks (Bietti and Mairal, 2019a,b; Scetbon and Harchaoui, 2020; Bietti, 2022). Kernel representations do not seem to suffer from aliasing effects; this can be explained by the Gaussian pooling layers that have been employed instead of max pooling: by discarding high-frequency information, shift invariance is preserved. Finally, some papers studied stability of CNNs in a broader sense, measured in terms of Lipschitz continuity (Szegedy et al., 2014; Virmaux and Scaman, 2018; Balan et al., 2018; D. Zou et al., 2020; Pérez et al., 2020). However, the Lipschitz bounds, which have been obtained theoretically, are generally several orders of magnitude higher than empirical results. This discrepancy may be due to the fact that these bounds were obtained for generic situations and represent overly conservative worst-case scenarios, rather than typical real-world situations. Furthermore, the specific case of convolutions with band-pass Gabor-like filters have been overlooked, except for Pérez et al. (2020).

In summary, we identified the following blind spots in the literature, regarding the topic of studying shift invariance in CNNs.

- The effect of the max pooling operator on network stability under small input shifts has not been investigated, particularly when used in combination with Gabor-like convolutions.

- While the shift invariance of CNNs tends to increase with network depth in the continuous framework, in the discrete case, the presence of subsampled convolutions with oriented band-pass filters can lead to aliasing artifacts. To our knowledge, the literature lacks theoretical studies that take these aliasing effects into account.

- Although extensive studies have been conducted on complex-valued convolutions followed by modulus, a link is missing to extend these results to standard CNNs, which implement real-valued convolutions and spatial pooling operators.

We have tackled these points in this thesis, as the main focus of our theoretical study, which is detailed in Chapter 4. More specifically, we established a probabilistic measure of shift invariance for max pooling outputs. Based on our findings, an approach to enhance stability was identified, which we experimentally tested in Chapter 5.

1.1.2 Introducing Gabor Inductive Bias to the Network

By design, CNNs are a subcategory of feedforward neural networks with inductive bias, leveraging some fundamental properties of images. Namely, it is assumed that (1) two pixels are more likely to be correlated when they are spatially close to each other (local connectivity); (2) all image areas should be processed equally by the network (weight sharing). In contrast, the Gabor-like properties observed in trained CNNs are not explicitly built into the network architecture but are instead entirely learned from data. As noted by Yosinski et al. (2014), this is a widespread phenomenon in CNNs and has been observed for various datasets and training objectives. A visual example is provided in Figure 1.1. This suggests that Gabor-like convolutions fundamentally act as a generic feature extractor for images. Therefore, integrating this property directly into the network architecture could provide an additional inductive bias to CNNs, thereby reducing their complexity by eliminating the need to learn this property from data.

In this purpose, several recent studies have proposed using Gabor filters in CNNs to replicate the behavior of freely-trained networks (Sarwar et al., 2017; Alekseev and Bobe, 2019; Pérez et al., 2020). In the last two papers, the parameters characterizing the Gabor filters are learned by the network. These studies however have certain limitations:

- The discrete Gabor transform requires manual tuning of several parameters such as subsampling factor, bandwidth, frequencies and rotation angles. While these parameters can be manually chosen based on prior knowledge or directly learned from data, it could be interesting to consider a Gabor-like transform having the same desired properties with fewer degrees of freedom.

- The previous papers did not include a quantitative evaluation of the degree of similarity between Gabor-controlled models and their baselines, particularly in terms of filter orientation, frequency, bandwidth, and number of Gabor-like filters.

- Whether learned from scratch or directly embedded in the network architecture, real-valued Gabor-like filters are inherently unstable to translations, as discussed in Section 1.1.1. However, theoretical considerations regarding complex Gabor-like filters offer insights into improving shift invariance, a topic not addressed in the aforementioned papers.

In Chapter 5, we introduce a *mathematical twin* of existing architectures, offering a greater control over their behavior. The above issues are directly addressed, paving the way for improving shift invariance in CNNs, as discussed in the next section.

1.1.3 Improving Shift Invariance in CNNs

Designing models to be nearly shift invariant introduces additional inductive bias to the architecture. This is advantageous because it avoids the burden of learning this property from data and improves network efficiency. At an identical level of complexity, a shift-invariant model can therefore produce better predictions, as detailed below.

Instability to translations has been addressed by several authors in recent years. As mentioned in Section 1.1.1, subsampled convolutions with oriented band-pass filters, also known as Gabor-like filters, have been identified as a significant cause of this problem. This phenomenon is directly related to the Nyquist-Shannon sampling theorem (Shannon,

1949), which implies that high-frequency signals must be blurred before subsampling, in order to avoid artifacts in reconstruction. Following this idea, some work (R. Zhang, 2019; X. Zou et al., 2023) proposed to introduce antialiasing layers based on low-pass filtering, called *blur pooling* (BlurPool), within CNNs. This approach simultaneously improved shift invariance and accuracy of various conventional architectures. However, increasing stability through low-pass filtering results in a significant loss of information. Although X. Zou et al. (2023) successfully mitigated this effect, it required additional computational resources.

Following a different approach, Chaman and Dokmanic (2021) reached perfect shift invariance without removing aliasing effects, using a strategy called *adaptive polyphase sampling* (APS). The idea is to select, for each input image and convolution layer, the subsampling grid that yields the highest l^p-norm in the resulting output, for a given $p \in \mathbb{N} \setminus \{0\}$. However, although shift invariance is satisfied for integer-pixel translations, this may not be the case for fractional-pixel translations requiring interpolation. In addition, since antialiasing was not performed, the feature maps produced by this method may include artifacts that were not present in the original signal. This may negatively impact the network's performance, compared to previous methods. In fact, combining this approach with blur pooling yielded the highest classification scores on ResNet trained with ImageNet, indicating that antialiasing is still beneficial even when perfect shift invariance has been achieved. Finally, APS necessitates m^2 times more computations and memory capacity than the conventional subsampling strategy, where $m \in \mathbb{N} \setminus \{0\}$ denotes the subsampling factor.

The main shortcomings of the previous approaches are summarized as follows.

- Blur pooling improves shift invariance at the cost of a loss of high-frequency information (R. Zhang, 2019). This weakness can be alleviated with an adaptive blur pooling filter, at the expense of additional computational resources and trainable parameters (X. Zou et al., 2023).

- Perfect shift invariance can be achieved for integer-pixel translations, using APS (Chaman and Dokmanic, 2021). Yet, aliasing effects may persist, leading to instabilities with respect to fractional-pixel translations and suboptimal accuracy. It also requires additional computational resources.

- Combining APS with blur-pooling-based antialiasing yields the best performances, but high-frequency information may still be lost in the process.

It is widely assumed that preserving high-frequency components in the output feature maps can enhance feature discrimination (Mallat, 2012). It could therefore be beneficial to develop an antialiasing method that, unlike blur pooling, retains high-frequency information without incurring additional computational costs. This is the primary focus of Chapter 5, in which we introduced complex-valued convolutions with Gabor-like filters to achieve our goal, based on the theoretical study from Chapter 4. We conducted our experiments on the mathematical twin briefly introduced in Section 1.1.2, in which the Gabor-like characteristics are imposed for a predetermined number of convolution kernels, rather than letting the network learn them from data.

1.2 Contributions

Chapters 2 and 3 provide a comprehensive background on CNNs and wavelet analysis, respectively. They discuss the current state of knowledge regarding stability properties in CNNs, and highlight the novelty of our findings in this context. Chapters 4 and 5 are devoted to the contributions of this thesis: Chapter 4 is mainly theoretical and tackles the questions raised in Section 1.1.1, whereas Chapter 5 is an experimental study combining the themes discussed in Sections 1.1.2 and 1.1.3. In the following, we present a brief overview of these contributions.

1.2.1 Studying Shift Invariance of Max Pooling Feature Maps

The main goal of Chapter 4 is to establish a probabilistic measure of shift invariance for max pooling outputs. Specifically, we consider a common scenario in which max pooling follows a convolution operator with a real-valued Gabor-like filter (see Figure 1.1), which is unstable to translation, as mentioned in Section 1.1.1 and detailed in Section 3.2.6. In this work, we show that, under specific conditions, max pooling can improve shift invariance. To address this topic, we adopt a four-step approach.

(1) Section 4.2 establishes a stability metric for shift invariance of a *complex-modulus* operator, conceptually defined by

$$\mathbb{C}\text{Mod} \quad : \quad \mathbb{C}\text{Conv} \to \text{Sub} \to \text{Modulus}, \tag{1.1}$$

where $\mathbb{C}$Conv refers to a convolution operator with a complex-valued Gabor-like kernel, whose real and imaginary parts approximately form a 2D Hilbert transform pair (Havlicek et al., 1997), and Sub denotes a subsampling operator. The claim that such an operator is nearly shift invariant is hinted by Kingsbury and Magarey (1998) but not formally proven.

(2) In Section 4.3, we introduce the operator of interest, *real-max-pooling*:

$$\mathbb{R}\text{Max} \quad : \quad \text{Conv} \to \text{Sub} \to \text{MaxPool}, \tag{1.2}$$

where Conv refers to a convolution operator with real-valued Gabor-like kernels, as commonly observed in standard CNNs after training on image datasets such as ImageNet. We then prove that, under additional conditions on the filter's frequency and orientation, $\mathbb{C}$Mod and $\mathbb{R}$Max produce similar outputs:

$$\mathbb{R}\text{Max} \approx \mathbb{C}\text{Mod}. \tag{1.3}$$

This result was hinted by Waldspurger (2015, pp. 190–191) for operators defined on the continuous domain $\mathbb{R}^2$, instead of $\mathbb{Z}^2$. However, because max pooling operates on a discrete grid, there are pathological situations where (1.3) is not satisfied. To overcome this difficulty, we adopt a probabilistic framework and bound the expected value of the normalized mean squared error between $\mathbb{R}$Max and $\mathbb{C}$Mod outputs. In particular, we show that the obtained estimation strongly depends on the convolution filter's frequency and orientation.

(3) From this result, we deduce in Section 4.4 a measure of shift invariance for $\mathbb{R}$Max operators, which benefits from the stability of $\mathbb{C}$Mod. Therefore, some filters, depending on their frequency and orientation, are more likely than others to produce stable image representations.

(4) Finally, Section 4.5 extends the results to operators defined on RGB input images, such as implemented in conventional CNN architectures.

In Section 4.6, we experimentally validate our theory by considering a deterministic feature extractor based on the dual-tree wavelet packet transform (DT-$\mathbb{C}$WPT). As discussed in Section 1.2.2, it possesses characteristics comparable to those of trained convolution layers in CNNs. We demonstrate a strong correlation between shift invariance of $\mathbb{R}$Max on the one hand, and similarity between $\mathbb{C}$Mod and $\mathbb{R}$Max on the other hand. We therefore establish a domain of validity for shift invariance of $\mathbb{R}$Max operators.

1.2.2 Improving Shift Invariance with Complex Gabor-like Convolutions

As explained earlier, Chapter 4 shows that the near-shift invariance property of $\mathbb{C}$Mod can be partially extended to $\mathbb{R}$Max, due to the proximity between the two operators. In Chapter 5, we build on these findings to design an antialiasing method in which $\mathbb{C}$Mod is used as a stable proxy for $\mathbb{R}$Max. The goal is to increase shift invariance and prediction accuracy in convolutional neural networks, as discussed in Section 1.1.3. More specifically, the proposed method consists in replacing the first layers of a CNN:

$$\text{Conv} \to \text{Bias} \to \text{Sub} \to \text{ReLU} \to \text{MaxPool}, \tag{1.4}$$

equivalently written as

$$\mathbb{R}\text{Max} \to \text{Bias} \to \text{ReLU}, \tag{1.5}$$

by the following combination:

$$\mathbb{C}\text{Mod} \to \text{Bias} \to \text{ReLU}, \tag{1.6}$$

where $\mathbb{C}$Mod and $\mathbb{R}$Max satisfy (1.1) and (1.2), respectively (note that, in (1.4), Bias and ReLU can be moved after MaxPool with no impact on the output). In compliance with the theoretical study from Chapter 4, the $\mathbb{R}$Max-$\mathbb{C}$Mod substitution is only applied to the channels associated with Gabor-like convolution kernels. This property can be enforced by introducing inductive bias to the original model, prior to antialiasing (see Section 1.1.2). Specifically, a predefined number of convolution kernels are guided to adopt Gabor-like structures, rather than letting the network learn them from data. To achieve this, our models implement the dual-tree wavelet packet transform (DT-$\mathbb{C}$WPT) (Bayram and I. W. Selesnick, 2008), an efficient algorithm based on separable filter banks. Unlike the Gabor transform, it only requires tuning one hyperparameter, the decomposition depth, to obtain a set of filters with predefined frequencies, orientations and bandwidth while also providing perfect reconstruction properties with limited redundancy. Empirical measurements reveal that the kernels provided by DT-$\mathbb{C}$WPT share many similarities with those observed in freely-trained CNNs. Throughout the chapter, we refer to this constrained model as a *mathematical twin*, because it employs a well-defined mathematical operator

to mimic the behavior of the freely-trained model. In this context, replacing $\mathbb{R}$Max by $\mathbb{C}$Mod is straightforward, since the complex-valued filters are provided by DT-$\mathbb{C}$WPT. To assess the resemblance between the freely-trained model and its mathematical twin, we experimentally measure and compare the properties of their convolution kernels.

Our $\mathbb{C}$Mod-based antialiasing method, implemented on AlexNet and ResNet, achieves superior accuracy on ImageNet and CIFAR-10 classification tasks, compared to prior methods based on low-pass filtering (R. Zhang, 2019; X. Zou et al., 2023). Arguably, our approach's emphasis on retaining high-frequency details contributes to a better balance between shift invariance and information preservation, resulting in improved performance. Furthermore, it has a lower computational cost and memory footprint than concurrent work, making it a promising solution for practical implementation.

1.3 Publications and Software

This thesis is based on the following papers:

- H. Leterme, K. Polisano, V. Perrier, and K. Alahari (2023). "From CNNs to Shift-Invariant Twin Models Based on Complex Wavelets". arXiv: 2212.00394, under review. This paper is an experimental study presenting the antialiasing method motivated in Section 1.1.3. It is the central theme of Chapter 5.
- H. Leterme, K. Polisano, V. Perrier, and K. Alahari (2022). "On the Shift Invariance of Max Pooling Feature Maps in Convolutional Neural Networks". arXiv: 2209.11740, under review. This paper, for which an expanded version is presented in Chapter 4, tackles the issues raised in Section 1.1.1.
- H. Leterme, K. Polisano, V. Perrier, and K. Alahari (2021). "Modélisation Parcimonieuse de CNNs Avec Des Paquets d'Ondelettes Dual-Tree". In: *ORASIS*. This conference paper (in French) presents a first version of the mathematical twin based on DT-$\mathbb{C}$WPT, addressing the questions discussed in Section 1.1.2. A refined version of the model is presented as part of Chapter 5.

As a related contribution, we released the following implementation on GitHub:

- H. Leterme (2023). *WCNN, a Python Library for Shift-Invariant Twin Models Based on Complex Wavelets*. URL: https://github.com/hubert-leterme/wcnn. This library provides PyTorch pretrained models for the mathematical twins presented in Chapter 5, in both $\mathbb{R}$Max and $\mathbb{C}$Mod (antialiased) versions. It also comes with training and evaluation scripts, and provides visualization tools to analyze convolution kernels, characteristic frequencies, bandwidth, *etc.*

Chapter 2

Background on Deep Learning for Classification

THIS CHAPTER presents an overview of deep learning and its applications to computer vision, with a focus on convolutional neural networks (CNNs). For the sake of consistency with the main topic of this thesis, the subject is tackled in the context of classification; applications of deep learning to other tasks are only briefly mentioned.The content of this chapter is inspired by Bishop and Mitchell (2014, Chapters 4–7) and LeCun et al. (2015).

2.1 The Linear Classification Problem

We start our journey by considering a simple classification task. Given an input space $\mathbb{R}^D$ with $D \in \mathbb{N} \setminus \{0\}$, and an integer $Q \in \mathbb{N}$, it consists in assigning a label $q \in \{0\,..\,Q-1\}$ to any D-dimensional input vector $\boldsymbol{x} \in \mathbb{R}^D$, or equivalently, partitioning $\mathbb{R}^D$ into Q *decision regions* $\{\mathcal{Z}_q\}_{q\in\{0..Q-1\}}$, each of which corresponding to a specific label, or class.

In a supervised setting, we consider a dataset of $N \in \mathbb{N}\setminus\{0\}$ labeled examples $(\boldsymbol{X}, \boldsymbol{q})$, with $\boldsymbol{X} := (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{N-1})^\top \in \mathbb{R}^{N\times D}$ and $\boldsymbol{q} \in \{0\,..\,Q-1\}^N$, referred to as a *training set*. The values q_n are called *ground truth* labels. The training procedure will then attempt to find an input space partition $\{\mathcal{Z}_q\}_{q\in\{0..Q-1\}}$ which provides a good classification accuracy on unseen examples. Let $(\boldsymbol{X}', \boldsymbol{q}')$ denote a *test set* of $N' \in \mathbb{N}\setminus\{0\}$ such examples, with $\boldsymbol{X}' := (\boldsymbol{x}'_0, \ldots, \boldsymbol{x}'_{N'-1})^\top \in \mathbb{R}^{N'\times D}$ and $\boldsymbol{q}' \in \{0\,..\,Q-1\}^{N'}$ (in general, $N' \ll N$). Ideally, we would like to get $\boldsymbol{x}'_n \in \mathcal{Z}_{q'_n}$ for any $n \in \{0\,..\,N'-1\}$, which corresponds to a 100% test accuracy. This, of course, only makes sense if we assume the training and test sets to share similar structures. For instance, we can assume the training and test examples to be drawn from the same conditional distribution given each label.

A naive training objective would be to perfectly fit the training examples to their ground truth labels, *i.e.*, $\boldsymbol{x}_n \in \mathcal{Z}_{q_n}$ for any $n \in \{0\,..\,N-1\}$. This is always possible provided the set of admissible decision regions has sufficient complexity. However, this does not guarantee that new data will be correctly classified. In the absence of additional constraints, the training procedure may learn either on features which are irrelevant for classification, or on random noise. Such a phenomenon, known as *overfitting*, can be handled with two possible approaches: (1) imposing additional constraints on the class of admissible decision regions; (2) penalizing unnecessarily complex solutions by adding a

regularization term to the objective function.

In what follows, we will restrict to the class of linear models for classification. In this framework, decision regions are separated by affine hyperplanes, referred to as *decision boundaries*. While this may seem too restrictive, we will see that more complex decision boundaries can be derived by preprocessing input data through nonlinear *feature extractors*.

2.1.1 Problem Formulation

A linear classifier is parameterized by a *weight matrix* $\boldsymbol{V} := (\boldsymbol{v}_0, \dots, \boldsymbol{v}_{Q-1})^\top \in \mathbb{R}^{Q\times D}$ and a *bias vector* $\boldsymbol{b} \in \mathbb{R}^Q$. Given an input $\boldsymbol{x} \in \mathbb{R}^D$, we first compute

$$\boldsymbol{y} := \boldsymbol{\Lambda}(\boldsymbol{V}\boldsymbol{x} + \boldsymbol{b}), \tag{2.1}$$

where $\boldsymbol{\Lambda} : \mathbb{R}^Q \to \mathbb{R}^Q$ denotes a (generally nonlinear) *activation function*. Then, the label $q \in \{0\,..\,Q-1\}$ assigned to $\boldsymbol{x}$ satisfies $y_q \geq y_{q'}$ for any $q' \neq q$. In other words, q is the output of

$$\varphi : (\boldsymbol{V}, \boldsymbol{b}, \boldsymbol{x}) \mapsto \operatorname{argmax} \boldsymbol{\Lambda}(\boldsymbol{V}\boldsymbol{x} + \boldsymbol{b}), \tag{2.2}$$

where $\operatorname{argmax} \boldsymbol{y}$ denotes the index of the maximal element in $\boldsymbol{y}$. Under specific conditions on $\boldsymbol{\Lambda}$ including monotonicity of its components, the decision boundary between any two regions $\mathcal{Z}_q$ and $\mathcal{Z}_{q'} \subset \mathbb{R}^D$ is a $(D-1)$-dimensional hyperplane defined by

$$(\boldsymbol{v}_{q'} - \boldsymbol{v}_q)^\top \boldsymbol{x} + (b_{q'} - b_q) = 0, \tag{2.3}$$

where $\boldsymbol{v}_q \in \mathbb{R}^D$ and $b_q \in \mathbb{R}$ respectively denote the q-th row vector of $\boldsymbol{V}$ and the q-th element of $\boldsymbol{b}$. For this reason, φ is called a *generalized linear model* (Nelder and Wedderburn, 1972).

Let $(\boldsymbol{X}, \boldsymbol{q})$ denote a training set. We consider a class of objective functions defined by

$$\mathcal{L} : (\boldsymbol{V}, \boldsymbol{b}, \boldsymbol{x}, q) \mapsto \mathcal{E}(\boldsymbol{V}\boldsymbol{x} + \boldsymbol{b}, q) + \lambda\,\mathcal{R}(\boldsymbol{V}), \tag{2.4}$$

where $\mathcal{E} : \mathbb{R}^Q \times \{0\,..\,Q-1\} \to \mathbb{R}_+$ denotes an *error function*, $\lambda > 0$ denotes a *regularization hyperparameter*, and $\mathcal{R} : \mathbb{R}^{Q\times D} \to \mathbb{R}_+$ denotes a *regularization function*. For a suitable choice of λ, the regularization term $\lambda\,\mathcal{R}(\boldsymbol{V})$ is intended to improve the classifier's generalization capabilities, for instance by reducing overfitting.

Then, evaluating the objective function for each training example and averaging over the whole dataset yields the *training loss*:

$$\tilde{\mathcal{L}} : (\boldsymbol{V}, \boldsymbol{b}, \boldsymbol{X}, \boldsymbol{q}) \mapsto \frac{1}{N}\sum_{n=0}^{N-1} \mathcal{L}(\boldsymbol{V}, \boldsymbol{b}, \boldsymbol{x}_n, q_n). \tag{2.5}$$

The training procedure consists in minimizing $\tilde{\mathcal{L}}(\boldsymbol{V}, \boldsymbol{b}, \boldsymbol{X}, \boldsymbol{q})$ with respect to the weight $\boldsymbol{V}$ and bias $\boldsymbol{b}$. In Section 2.1.3, we present and discuss several such objective functions.

Remark 2.1. In machine learning literature, the weight matrix is often denoted by $\boldsymbol{W}$. In this thesis however, this denomination is reserved for complex-valued weight matrices.

Remark 2.2. In many cases, the problem will be more constrained, and $\boldsymbol{V}$ and $\boldsymbol{b}$ will not be fully trainable. Instead, we will consider parameters $\boldsymbol{v}$ and b from which $\boldsymbol{V}$ and $\boldsymbol{b}$ can be recovered:

$$\boldsymbol{V} := \Xi(\boldsymbol{v}); \qquad \boldsymbol{b} := \Pi(b), \tag{2.6}$$

for certain linear functions Ξ and Π to be defined. The nature of $\boldsymbol{v}$ and b—a one-dimensional vector, a convolution kernel, a scalar—will be clarified when necessary. Then, we derive from (2.2) and (2.4), respectively, a generalized linear model and an objective function parameterized by $\boldsymbol{v}$ and b:

$$\varphi_0 : (\boldsymbol{v},\, b,\, \boldsymbol{x}) \mapsto \varphi(\Xi(\boldsymbol{v}),\, \Pi(b),\, \boldsymbol{x}); \tag{2.7}$$

$$\mathcal{L}_0 : (\boldsymbol{v},\, b,\, \boldsymbol{x},\, q) \mapsto \mathcal{L}(\Xi(\boldsymbol{v}),\, \Pi(b),\, \boldsymbol{x},\, q). \tag{2.8}$$

In this context, the training procedure consists in minimizing

$$\tilde{\mathcal{L}}_0(\boldsymbol{v},\, b,\, \boldsymbol{X},\, \boldsymbol{q}) := \tilde{\mathcal{L}}(\Xi(\boldsymbol{v}),\, \Pi(b),\, \boldsymbol{X},\, \boldsymbol{q}) \tag{2.9}$$

$$= \frac{1}{N} \sum_{n=0}^{N-1} \mathcal{L}_0(\boldsymbol{v},\, b,\, \boldsymbol{x}_n,\, q_n) \tag{2.10}$$

with respect to $\boldsymbol{v}$ and b.

2.1.2 The Two-Class Problem

A particular case of linear model is when $Q = 2$. Following Remark 2.2, we consider a simpler setting parameterized by a weight vector $\boldsymbol{v} \in \mathbb{R}^D$ and bias scalar value $b \in \mathbb{R}$. The linear mappings introduced in (2.6) are then defined by

$$\Xi : \boldsymbol{v} \mapsto \begin{bmatrix} \boldsymbol{v}^\top \\ -\boldsymbol{v}^\top \end{bmatrix}; \qquad \Pi : b \mapsto \begin{pmatrix} b \\ -b \end{pmatrix}. \tag{2.11}$$

Besides, the activation $\boldsymbol{\Lambda}(\boldsymbol{y})$ is obtained by applying a one-dimensional activation function $\Lambda_0 : \mathbb{R} \to \mathbb{R}$ to each member of $\boldsymbol{y}$. Then, the generalized linear model introduced in (2.7) becomes:

$$\varphi_0(\boldsymbol{v},\, b,\, \boldsymbol{x}) = \begin{cases} 0 & \text{if } \Lambda_0(\boldsymbol{v}^\top \boldsymbol{x} + b) \geq \Lambda_0(-\boldsymbol{v}^\top \boldsymbol{x} - b); \\ 1 & \text{otherwise.} \end{cases} \tag{2.12}$$

For training, the error function introduced in (2.4) is defined by

$$\mathcal{E} : (\boldsymbol{y},\, q) \mapsto \mathcal{E}_0(y_1,\, q), \tag{2.13}$$

for some one-dimensional error function $\mathcal{E}_0 : (\mathbb{R},\, \{0\,..\,Q-1\}) \mapsto \mathbb{R}_+$ to be specified. The objective function $\mathcal{L}_0$ as introduced in (2.8) then becomes:

$$\mathcal{L}_0(\boldsymbol{v},\, b,\, \boldsymbol{x},\, q) = \mathcal{E}_0\left(\boldsymbol{v}^\top \boldsymbol{x} + b,\, q\right) + \lambda\, \mathcal{R}_0(\boldsymbol{v}), \tag{2.14}$$

where we have defined $\mathcal{R}_0 : \boldsymbol{v} \mapsto \mathcal{R}(\Xi(\boldsymbol{v}))$.

2.1.3 Examples of Linear Classifiers

We now present some typical examples of classifiers and discuss their properties. By default, they apply to the general multilabel setting. However, some of them are specifically designed for the two-class problem. Even though extensions to $Q > 2$ are generally possible, for example by adopting *one-versus-one* or *one-versus-rest* strategies, they come with fundamental limitations such as ambiguous decision regions or imbalanced training sets (Bishop and Mitchell, 2014, pp. 338-339).

A Naive Classifier based on Least Squares. By analogy with linear regression, it could be tempting to use an objective function based on the sum-of-squares error function. In this setting, the activation function $\boldsymbol{\Lambda}$ introduced in (2.4) is the identity, and the error function $\mathcal{E}$ is defined, for any $\boldsymbol{y} \in \mathbb{R}^Q$ and any $q \in \{0\,..\,Q-1\}$, by

$$\mathcal{E}(\boldsymbol{y},\, q) := \left\| \boldsymbol{y} - \tilde{\boldsymbol{y}}_q \right\|_2^2, \tag{2.15}$$

where $\tilde{\boldsymbol{y}}_q \in \{0,\, 1\}^Q$ is a *binary target vector*, defined by $\tilde{y}_{qq'} := \delta_{qq'}$.

This approach is convenient since it provides a closed-form solution for the optimal parameters $\boldsymbol{V}$ and $\boldsymbol{b}$. However, it suffers from two major drawbacks: (1) lack of robustness to outliers; (2) failure to correctly classify training examples when $Q > 2$, even when the classes are well separated. The second point comes from the implicit assumption that the target vector $\tilde{\boldsymbol{y}}_q$ is drawn from a Gaussian conditional distribution given $\boldsymbol{x}$ (Bishop and Mitchell, 2014, p.186), which is reasonable for linear regression but not for classification.

The Perceptron. The perceptron algorithm (Rosenblatt, 1958) plays an important role in the history of machine learning since it is one the first to be intended for practical implementation on a physical machine. It is designed as a simplified model of a biological neural network, such as described by McCulloch and Pitts (1943).

The perceptron is fundamentally a two-class model; generalizations to $Q > 2$ are not straightforward. In this framework, the one-dimensional activation function Λ_0 such as introduced in Section 2.1.2 is a *step function*:

$$\Lambda_0 : y \mapsto \begin{cases} +1 & \text{if } y \geq 0; \\ -1 & \text{if } y < 0, \end{cases} \tag{2.16}$$

whereas the one-dimensional error function $\mathcal{E}_0$ is defined as

$$\mathcal{E}_0 : (y,\, q) \mapsto \max(-y\,\check{q},\, 0), \qquad \text{where} \qquad \check{q} := \begin{cases} +1 & \text{if } q = 0; \\ -1 & \text{if } q = 1. \end{cases} \tag{2.17}$$

Therefore, $\mathcal{E}_0(\boldsymbol{v}^\top \boldsymbol{x} + b,\, q) > 0$ for misclassified examples only, except those lying on the decision boundary. Besides, the perceptron does not use regularization; thus $\lambda = 0$. Then, the training loss $\tilde{\mathcal{L}}_0$ defined in (2.14) is minimized with respect to $\boldsymbol{v}$ and b, following the stochastic gradient descent procedure (see Section 2.2.2).

An important theoretical result is the *perceptron convergence theorem*. It states that, if the training set is linearly separable, then $\tilde{\mathcal{L}}_0(\boldsymbol{v},\, b,\, \boldsymbol{x},\, q)$ will reach 0 after a finite number of steps. However, in the opposite case, the algorithm will never converge, which restricts

its practical utility to strictly linearly separable training sets. This fundamental limitation was pointed out in a book by Minsky and Papert (1988), which was first published in 1969 and resulted in the emerging machine learning community looking away from neural networks until the mid-eighties. Even if the training set is linearly separable, the number of steps before reaching convergence can be important, which makes it difficult to distinguish between a nonseparable dataset and a dataset which is simply slow to converge. Another major issue is about the number of exact solutions. Depending on the choice of parameter initialization as well as the order in which datapoints are fed into the network, the perceptron algorithm may converge to many different valid states. However, not all of them perform equally well on unseen data.

Support Vector Machines. Support vector machines (SVMs) (Boser et al., 1992; Cortes and Vapnik, 1995) have been very popular in the past decades for solving classification tasks, including face detection (Osuna et al., 1997), but also in other contexts such as regression (Smola and Schölkopf, 2004). Similar to the perceptron, they are specifically designed for the two-class problem.

The main idea is to find a decision boundary maximizing the *margin*. In the linearly-separable setting, the margin is defined as the Euclidean distance between the decision boundary and the closest datapoints from both classes. In a more general case, some datapoints are allowed to lie inside the margin or to be misclassified. The penalty for violating this constraint is proportional to the distance from those misplaced datapoints to the margin boundary. The datapoints lying on or inside the margin, or being misclassified, are referred to as *support vectors*.

The SVM model can be expressed in the framework presented in Section 2.1.2. Similar to the perceptron, the activation function Λ_0 is chosen to be the step function as defined in (2.16). The error function $\mathcal{E}_0$ is the *hinge error function*:

$$\mathcal{E}_0 : (y,\, q) \mapsto \max(1 - y\,\check{q},\, 0), \tag{2.18}$$

where the target value $\check{q}$ has been introduced in (2.17). In addition, regularization is performed using the squared l^2-norm:

$$\mathcal{R}_0(\boldsymbol{v}) := \|\boldsymbol{v}\|_2^2\,. \tag{2.19}$$

The tradeoff between the margin size and tolerance with respect to faulty datapoints is controlled by a regularization hyperparameter λ such as introduced in (2.14).

The Logistic Regression. Since the step function is used for activation in the two previous classifiers, predictions are made in a binary fashion. With logistic regression, we shall see that a linear classifier can also provide probabilistic outputs.

In its two-class formulation, logistic regression uses the *logistic sigmoid function* as an activation function, which outputs confidence values in $[0,\, 1]$:

$$\Lambda_0 : y \mapsto \frac{1}{1 + \mathrm{e}^{-y}}. \tag{2.20}$$

While this function—and a closely-related cousin named the *probit*—was originally introduced as a regression model for population growth (Pearl and Reed, 1920) and bioassay

(Berkson, 1944), its theoretical and practical interest for classification was unraveled in the seventies (McFadden, 1973; McKelvey and Zavoina, 1975). More details on the history of logistic regression can be found in a review by Cramer (2002).

We now examine the general multiclass problem and explain that, under specific assumptions on the input data distribution, the output of logistic regression can be interpreted as a probability distribution over the set of labels. Let us assume that the training examples $(\boldsymbol{x}_n, q_n)$ are drawn from N independent and identically distributed (i.i.d.) random variables $(\mathbf{X}_n, \mathsf{Q}_n) \sim (\mathbf{X}, \mathsf{Q})$. Moreover, we assume that the conditional distributions of $\mathbf{X}$ given $\mathsf{Q} = q$ for any class $q \in \{0\,..\,Q-1\}$ constitute an *exponential family of distributions* (Bishop and Mitchell, 2014, p. 203), *e.g.*, a family of multivariate Gaussian distributions with a shared covariance matrix $\boldsymbol{\Sigma}$.

One of the main motivations for using logistic regression lies in the following result concerning posterior probabilities. Under the exponential family hypothesis, there exists $\boldsymbol{V} \in \mathbb{R}^{Q\times D}$ and $\boldsymbol{b} \in \mathbb{R}^Q$ such that, for any $\boldsymbol{x} \in \mathbb{R}^D$ and $q \in \{0\,..\,Q-1\}$,

$$\mathbb{P}\{\mathsf{Q} = q \mid \mathbf{X} = \boldsymbol{x}\} = \Lambda_q(\boldsymbol{V}\boldsymbol{x} + \boldsymbol{b}), \tag{2.21}$$

where $\Lambda_q : \mathbb{R}^Q \to [0,\,1]$ is defined by

$$\Lambda_q : \boldsymbol{y} \mapsto \frac{\exp(y_q)}{\sum_{q'=0}^{Q-1} \exp(y_{q'})} \tag{2.22}$$

denotes the *normalized exponential*, which is a multiclass generalization of the logistic sigmoid function.

Expression (2.22) defines the q-th component of a *softmax* function $\boldsymbol{\Lambda} : \mathbb{R}^Q \to [0,\,1]^Q$, which can be used as a nonlinear activation function in the general framework introduced in Section 2.1.1. According to (2.21), the output can be interpreted as a posterior probability distribution over the set of labels. The linear model (2.2) then assigns, for any input $\boldsymbol{x} \in \mathbb{R}^D$, the most probable label $q \in \{0\,..\,Q-1\}$, conditionally to $\mathbf{X} = \boldsymbol{x}$.

The model should accurately reflect the empirical distribution of the training data. The training procedure therefore consists in maximizing the joint posterior probability $\mathbb{P}\{\mathsf{Q}_0 = q_0,\,\ldots,\,\mathsf{Q}_{N-1} = q_{N-1} \mid \mathbf{X}_0 = \boldsymbol{x}_0,\,\ldots,\,\mathbf{X}_{N-1} = \boldsymbol{x}_{N-1}\}$ with respect to $\boldsymbol{V}$ and $\boldsymbol{b}$. Under the i.i.d. hypothesis, this is equivalent to minimizing the *average negative log-likelihood*, also called *cross-entropy loss*, over the training examples $(\boldsymbol{x},\,q)$. In this context, the error function $\mathcal{E}$, introduced in (2.4), is defined by

$$\mathcal{E}(\boldsymbol{y},\,q) := -\ln \Lambda_q(\boldsymbol{y}), \qquad \text{with} \qquad \boldsymbol{y} := \boldsymbol{V}\boldsymbol{x} + \boldsymbol{b}. \tag{2.23}$$

As seen above, performing logistic regression is justified if the input data are drawn from an exponential family of distributions. Moreover, the softmax output provides a probabilistic estimation of the prediction accuracy. Logistic regression is widely used as the final classifier in modern neural network architectures, introduced in Section 2.3.

2.1.4 Beyond Linear Models

The above models are linear classifiers, which means that the decision boundaries are constrained to be affine hyperplanes. Although SVMs and logistic regression do not require strict linear separability, such decision boundaries often do not reflect the complexity of

underlying data structures. To get an intuition for this, consider a classification task for animal species (LeCun et al., 2015). In the input space, also called *pixel space*, two images of the same animal in two different poses or background environment may be very far away from each other. Conversely, images of two different yet visually similar animals may be close to each other in the pixel space. In this context, a linear decision boundary may fail to discriminate the latter two images while identifying the former as belonging to the same species.

A common way to overcome this limitation is to preprocess input data through a nonlinear operator $\boldsymbol{\Gamma} : \mathbb{R}^D \to \mathbb{R}^{D'}$, and apply the linear classifier on its output. Equation (2.1) is then replaced by

$$\boldsymbol{y} := \boldsymbol{\Lambda}(\boldsymbol{V}\,\boldsymbol{\Gamma}(\boldsymbol{x}) + \boldsymbol{b}), \tag{2.24}$$

with $\boldsymbol{V} \in \mathbb{R}^{Q \times D'}$. Such an operator is called a *feature extractor*, and the corresponding output space $\mathbb{R}^{D'}$ is referred to as a *feature space*. Whereas the decision boundaries remain linear in the feature space, they may correspond to nonlinear decision boundaries in the input space.

Designing a good feature extractor is far from being straightforward. This task often requires high domain expertise and thus prior knowledge on the underlying data structure. In this section, we review several popular approaches for feature extraction. They fall into three different categories: (1) generic, nonparametric feature extractors (kernel methods); (2) specific, handcrafted feature extractors (local descriptors, wavelet scattering transform); (3) generic feature extractors with trainable parameters (multilayer perceptron).

Kernel Methods. This concept was first introduced and applied to the perceptron by Aizerman (1964), and later rediscovered with the emergence of SVMs (Boser et al. 1992). Kernel methods use generic feature extractors, for which the feature space $\mathbb{R}^{D'}$ can be of very high—or even infinite—dimension, thus allowing very complex feature representations. This is made computationally tractable by taking advantage of the *kernel trick* (Aizerman, 1964). In the context of SVMs, the inner product $\boldsymbol{v}^\top \boldsymbol{\Gamma}(\boldsymbol{x})$, which is required in (2.12) and (2.14), can be computed by performing $N_{\text{supp}} \ll N$ evaluations on the subset of support vectors $\{\boldsymbol{x}_n\}_{n \in \mathcal{S}}$, with $\mathcal{S} \subset \{0\,..\,N-1\}$. More precisely, there exists a vector of *Lagrange multipliers*, denoted by $\boldsymbol{a} \in \mathbb{R}_+^N$, such that, for any $\boldsymbol{x} \in \mathbb{R}^D$,

$$\boldsymbol{v}^\top \boldsymbol{\Gamma}(\boldsymbol{x}) = \sum_{n \in \mathcal{S}} a_n \check{q}_n \, \mathcal{K}_{\boldsymbol{\Gamma}}(\boldsymbol{x}_n, \boldsymbol{x})\,, \tag{2.25}$$

where $\check{\boldsymbol{q}} \in \{-1, +1\}^N$ denotes the vector of target values such as introduced in (2.17). Moreover, in the above expression we have introduced

$$\mathcal{K}_{\boldsymbol{\Gamma}} : (\boldsymbol{x}, \boldsymbol{x}') \mapsto \boldsymbol{\Gamma}(\boldsymbol{x})^\top \boldsymbol{\Gamma}(\boldsymbol{x}'), \tag{2.26}$$

which is referred to as a *kernel function*. Therefore, $\boldsymbol{v}$ and $\boldsymbol{\Gamma}(\boldsymbol{x})$ do not need to be explicitly computed. It turns out that, for any *positive semidefinite* kernel function $\mathcal{K} : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$, the existence of $\boldsymbol{\Gamma}$ such that $\mathcal{K} = \mathcal{K}_{\boldsymbol{\Gamma}}$ as defined in (2.26) is guaranteed (Bishop and Mitchell, 2014, p.295). Therefore, an explicit formulation of $\boldsymbol{\Gamma}$ can be avoided.

A popular example is the *Gaussian kernel*, defined by

$$\mathcal{K} : (\boldsymbol{x}, \boldsymbol{x}') \mapsto \exp\left(\frac{-\,\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2}{2\sigma^2}\right), \tag{2.27}$$

for which the implicit feature extractor $\boldsymbol{\Gamma}$ maps any input vector to an infinite-dimensional space.

Local Descriptors. An important family of hand-crafted feature extractors are the *local descriptors*. Their primary purpose is to extract a set of local features from images that can be used to perform matching between several inputs. For instance, two pictures of the same building seen from various angles and weather conditions are expected to produce nearly identical sets of local features.

One of the first feature descriptors of this kind was developed by Harris and Stephens (1988). Despite its success, the Harris detector is very unstable with respect to changes of scale, which makes it poorly suited for matching, and many other vision tasks including classification. This weakness led to the development of scale- and rotation-invariant descriptors which are also robust to distortions, cluttered environments or changes of luminosity. A popular example is the *scale-invariant feature transform* (SIFT) descriptor (Lowe, 2004), which gave birth to several extensions including PCA-SIFT (Ke and Sukthankar, 2004), and *gradient location and orientation histogram* (GLOH) (Mikolajczyk and Schmid, 2005).

The Wavelet Scattering Transform. A recent example of specific, handcrafted feature extractor is the *wavelet scattering transform*, introduced by Mallat (2012). This nonlinear transform is designed to produce near-translation-invariant signal representations which are stable to deformations and preserve high-frequency information. The key idea is that two slightly shifted or deformed signals are generally expected to be of the same nature, and should therefore be assigned the same label. On the other hand, this transform is designed to extract discriminative features such as directional structures at various scales.

The wavelet scattering transform gave birth to a whole new family of classifiers called *wavelet scattering networks* (ScatterNets) (Bruna and Mallat, 2013). They achieved good performance on textures or small image datasets such as handwritten digits, as well as tasks with limited labeled data. Furthermore, they brought some theoretical insights to the field of deep learning, due to their resemblance with standard convolutional neural networks (see Section 2.3).

A detailed description of the wavelet scattering transform and an overview on its various applications are provided in Section 3.4.

Multilayer Perceptrons. A powerful alternative to the above approaches is to design parametric feature extractors with a generic learning procedure. This is where deep learning comes into play. The next section focuses on an important class of models, called *multilayer perceptrons* or *feedforward neural networks*, containing such parametric feature extractors. In each layer, the network captures more and more discriminative features while becoming less sensitive to intra-class variations. Seen from another angle, the input space is progressively distorted until becoming suitable for linear decision boundaries. As we will see, convolutional neural networks constitute a subclass of feedforward neural networks which are well suited for image classification.

2.2 Multilayer Perceptrons

Multilayer perceptrons (MLPs) were popularized by Rumelhart et al. (1986). Although the concept of multilayer networks was not new at the time, the authors contributed to design an efficient training procedure called *gradient backpropagation*. Consequently, such models became well suited for practical implementations.

This section provides a detailed description of MLPs as well as the related training procedure, based on stochastic gradient descent and backpropagation. The general framework introduced in Section 2.1 holds, except that (2.1) is replaced by (2.24), to account for the nonlinear feature extractor $\boldsymbol{\Gamma}$.

2.2.1 Parametric Feature Extractor

The feature extractor $\boldsymbol{\Gamma}$ is built as a succession of operators following the prototype (2.24). We consider $P \in \mathbb{N}$ as the number of *hidden layers*. Then for any $p \in \{0\,..\,P-1\}$, let $D_p \in \mathbb{N} \setminus \{0\}$ denote the number of input features in the $(p+1)$-th layer, and $D' := D_P$ denote the size of output vectors. Note that $D := D_0$ represents the size of input vectors $\boldsymbol{x} \in \mathbb{R}^D$. Then, we define

$$\boldsymbol{\Gamma} := \boldsymbol{\Gamma}^{(P)}, \qquad \text{with} \qquad \boldsymbol{\Gamma} : \mathbb{R}^D \to \mathbb{R}^{D'}, \tag{2.28}$$

in a recursive manner. First, $\boldsymbol{\Gamma}^{(0)} : \mathbb{R}^D \to \mathbb{R}^D$ is defined as the identity. Then, for any $p \in \{0\,..\,P-1\}$, $\boldsymbol{\Gamma}^{(p+1)} : \mathbb{R}^D \to \mathbb{R}^{D_{p+1}}$ satisfies

$$\boldsymbol{\Gamma}^{(p+1)} : \boldsymbol{x} \mapsto \boldsymbol{\Lambda}^{(p)}(\boldsymbol{V}^{(p)}\boldsymbol{\Gamma}^{(p)}(\boldsymbol{x}) + \boldsymbol{b}^{(p)}), \tag{2.29}$$

where $\boldsymbol{\Lambda}^{(p)} : \mathbb{R}^{D_{p+1}} \to \mathbb{R}^{D_{p+1}}$ denotes a nonlinear activation function. Moreover, $\boldsymbol{V}^{(p)} \in \mathbb{R}^{D_{p+1} \times D_p}$ and $\boldsymbol{b}^{(p)} \in \mathbb{R}^{D_{p+1}}$ denote a trainable weight matrix and bias vector, respectively.

Note that the name "multilayer perceptron" should be seen as a tribute to Rosenblatt's perceptron rather than a proper description of the model. In fact, $\boldsymbol{\Lambda}^{(p)}$ is generally chosen to be a sigmoid logistic function (Rumelhart et al., 1986), a scaled hyperbolic tangent (LeCun et al., 1989), or more recently, a rectified linear unit (ReLU) (Glorot et al., 2011). Besides, the final classifier is generally chosen to be a logistic regression model, thus providing a probabilistic estimation for the assigned labels.

2.2.2 Training Procedure

During training, the parameters of the feature extractor $\boldsymbol{\Gamma}$ are updated simultaneously with those of the final classifier. For the sake of consistency, we therefore denote by

$$\boldsymbol{V}^{(P)} := \boldsymbol{V} \qquad \text{and} \qquad \boldsymbol{b}^{(P)} := \boldsymbol{b} \tag{2.30}$$

the weight tensor and bias vector in the final classifier. Then, the objective function introduced in (2.4) takes additional arguments:

$$\mathcal{L} : \Big((\boldsymbol{V}^{(p)}, \boldsymbol{b}^{(p)})_{p\in\{0..P\}}, \boldsymbol{x}, q\Big) \mapsto \mathcal{E}\left(\boldsymbol{V}^{(P)}\boldsymbol{\Lambda}^{(P-1)}\left(\boldsymbol{V}^{(P-1)}\boldsymbol{\Lambda}^{(P-2)}(\dots) + \boldsymbol{b}^{(P-1)}\right) + \boldsymbol{b}^{(P)}, q\right), \tag{2.31}$$

and the corresponding training loss $\tilde{\mathcal{L}}$, introduced in (2.5), is modified accordingly. Note that the regularization term is discarded for the sake of simplicity, even though regularization is possible in MLPs.

The training procedure is generally performed using stochastic gradient descent (SGD). At each step, the algorithm randomly selects a subset $(\boldsymbol{X}^{\mathrm{b}}, \boldsymbol{q}^{\mathrm{b}})$, called *minibatch*, of the training set $(\boldsymbol{X}, \boldsymbol{q})$. Then, using backpropagation, the gradient of the training loss

$$\tilde{\mathcal{L}}^{\mathrm{b}} := \tilde{\mathcal{L}}\left((\boldsymbol{V}^{(p)}, \boldsymbol{b}^{(p)})_{p\in\{0..P\}}, \boldsymbol{X}^{\mathrm{b}}, \boldsymbol{q}^{\mathrm{b}}\right) \tag{2.32}$$

is recursively computed with respect to $\boldsymbol{V}^{(p)}$ and $\boldsymbol{b}^{(p)}$, for each $p \in \{0\,..\,P\}$. Finally, the parameters are updated using the following rule:

$$\boldsymbol{V}^{(p)} \leftarrow \boldsymbol{V}^{(p)} - \alpha\,\nabla_{\boldsymbol{V}^{(p)}}\tilde{\mathcal{L}}^{\mathrm{b}}; \qquad \boldsymbol{b}^{(p)} \leftarrow \boldsymbol{b}^{(p)} - \alpha\,\nabla_{\boldsymbol{b}^{(p)}}\tilde{\mathcal{L}}^{\mathrm{b}}, \tag{2.33}$$

where $\alpha > 0$ denotes a *learning rate*.

Despite promising beginnings, multilayer neural networks became overshadowed by other machine learning approaches in the late 1990s and early 2000s, in particular by SVMs and kernel methods. This is partly because stochastic gradient descent was thought to be fatally stuck at local minima and therefore produce suboptimal solutions. However, one subclass of MLPs—namely, *convolutional neural networks* (CNNs)—continued to blossom in the computer vision field, even though its most outstanding successes only came out by the 2010s.

2.3 Convolutional Neural Networks

According to Scherer et al. (2010), CNNs are based on the concepts of simple and complex cells from the mammal visual cortex (Hubel and Wiesel, 1962). A notable precursor is the *neocognitron*, developed by Fukushima and Miyake (1982). However, actual models trained by backpropagation emerged subsequently to Rumelhart et al. (1986). The first one- and two-dimensional applications were developed simultaneously, respectively by Waibel et al. (1989) for phonemes and word recognition, and LeCun et al. (1989) in the context of handwritten digits recognition.

CNNs synthesize three main ideas: *local connections*, *weight sharing* and *pooling*. The two first ones arise from the properties of input images. Features are more likely to be correlated if they are spatially close to each other. Besides, objects or shapes are generally not bound to a specific location. Therefore, the same weights can be repeated across the spatial extent of input images. This allows reducing the number of freely-trained parameters by a large amount (LeCun et al., 1989). Local connections and weight sharing are implemented in *convolution layers*, which are described in Section 2.3.1. Additionally, pooling is another key concept in CNNs, which is developed in Section 2.3.2.

Following this approach, subsequent research led to develop CNNs for face detection and localization (Vaillant et al., 1994), face recognition (Lawrence et al., 1997), document reading (LeCun et al., 1998), video segmentation (Ning et al., 2005), and more recently, pose estimation (Tompson et al., 2015). However, a major breakthrough came with Krizhevsky et al. (2017), who won the 2012 edition of the ILSVRC challenge on ImageNet. This was the first time a CNN architecture reached competitive performance in a classification task on large image datasets. More details on this are provided in Section 2.3.3.

2.3.1 Convolution Layers

Input data usually take the form of multiple arrays (multichannel inputs), either 1D (*e.g.*, audio signals or language sequences), 2D (*e.g.*, images or audio spectrograms) or 3D (*e.g.*, videos or volumetric images). In this section, we describe a typical convolution layer for 1D signals and show that it fits in the MLP framework as described in Section 2.2. Generalization to higher dimensions is quite straightforward.

A one-dimensional *convolution layer* is parameterized by a *convolution kernel* $\boldsymbol{v} \in \mathbb{R}^d$ for a given $d \in \mathbb{N} \setminus \{0\}$ ($d \ll D$) and a bias $b \in \mathbb{R}$. Then, we consider

$$\Xi : \mathbb{R}^d \to \mathbb{R}^{D \times D} \qquad \text{and} \qquad \Pi : \mathbb{R} \to \mathbb{R}^D \tag{2.34}$$

as in Remark 2.2 (Section 2.1.1). They are defined by

$$\Xi : \boldsymbol{v} \mapsto \begin{pmatrix} v_{\lfloor d/2 \rfloor} & \dots & v_{d-1} & 0 & 0 & & \dots & & 0 \\ v_{\lfloor d/2 \rfloor - 1} & \dots & v_{d-2} & v_{d-1} & 0 & & \dots & & 0 \\ \vdots & & & & \vdots & & & & \vdots \\ 0 & \dots & 0 & v_0 & \dots & v_{d-1} & 0 & \dots & 0 \\ \vdots & & & & \vdots & & & & \vdots \\ 0 & & \dots & & 0 & v_0 & v_1 & \dots & v_{\lfloor d/2 \rfloor + 1} \\ 0 & & \dots & & 0 & 0 & v_0 & \dots & v_{\lfloor d/2 \rfloor} \end{pmatrix} \tag{2.35}$$

and

$$\Pi : b \mapsto (b, \dots, b)^\top. \tag{2.36}$$

Then, for any input $\boldsymbol{x} \in \mathbb{R}^D$, the convolution layer, followed by a nonlinear activation function $\boldsymbol{\Lambda} : \mathbb{R}^D \to \mathbb{R}^D$, produces an output $\boldsymbol{y} \in \mathbb{R}^D$ satisfying (2.1):

$$\boldsymbol{y} := \boldsymbol{\Lambda}(\boldsymbol{V}\boldsymbol{x} + \boldsymbol{b}), \tag{2.37}$$

with

$$\boldsymbol{V} := \Xi(\boldsymbol{v}) \in \mathbb{R}^{D \times D} \qquad \text{and} \qquad \boldsymbol{b} := \Pi(b) \in \mathbb{R}^D. \tag{2.38}$$

$\boldsymbol{V}$ is called a *Toeplitz matrix*. The notions of local connection and weight sharing emerge from this structure.

We denote by $\overline{\boldsymbol{v}} \in \mathbb{R}^d$ the "flipped vector", defined by $\overline{v}_i := v_{d-1-i}$. The matrix-vector product $\boldsymbol{V}\boldsymbol{x}$ can then be written in the form of a convolution product:

$$\boldsymbol{V}\boldsymbol{x} = \boldsymbol{x} * \overline{\boldsymbol{v}}, \tag{2.39}$$

with

$$(\boldsymbol{x} * \overline{\boldsymbol{v}})_j := \sum_{i=0}^{d-1} x_i \, v_{i-j+\lfloor d/2 \rfloor} \quad \forall j \in \{0 \,..\, D-1\}, \tag{2.40}$$

where we have defined $v_{i'} := 0$ if $i' \notin \{0 \,..\, d-1\}$.

If several pairs of convolution layer and activation function are stacked on top of each other, we obtain a multilayer feature extractor satisfying (2.28) and (2.29). The final

classifier remains unchanged, *i.e.*, all the belements of $\boldsymbol{V}^{(P)}$ are freely trained. The last layer is therefore qualified as *fully connected*.

In the multilayer setting, the above quantities are superscripted by p, for any hidden layer $p \in \{0\,..\,P-1\}$. Then, the partial feature extractor (2.29) becomes

$$\boldsymbol{\Gamma}^{(p+1)} : \boldsymbol{x} \mapsto \boldsymbol{\Lambda}^{(p)}\left(\boldsymbol{\Gamma}^{(p)}(\boldsymbol{x}) * \overline{\boldsymbol{v}}^{(p)} + b^{(p)}\right), \tag{2.41}$$

where the vector-scalar sum $\boldsymbol{y}' := \boldsymbol{y} + b$ is defined by $y'_i := y_i + b$ for any i.

Remark 2.3. To avoid the cumbersome definition of convolutions between fixed-length vectors (2.40), we cast $\boldsymbol{x}$ and $\boldsymbol{v}$ into finitely-supported sequences, denoted by roman, non-bold letters x and $\mathrm{v} \in l^2_{\mathbb{R}}(\mathbb{Z})$, where $l^2_{\mathbb{R}}(\mathbb{Z})$ denotes the space of real-valued square-summable sequences. Indexing is made between square brackets. The support of v is chosen to be centered around 0, such that $\overline{\mathrm{v}}[n] := \mathrm{v}[-n]$ for any $n \in \mathbb{Z}$. Then, $\boldsymbol{x} * \overline{\boldsymbol{v}}$ can be retrieved from the standard convolution between x and $\overline{\mathrm{v}}$, defined by

$$(\mathrm{x} * \overline{\mathrm{v}})[n] := \sum_{p\in\mathbb{Z}} \mathrm{x}[p]\,\mathrm{v}[p-n] \qquad \forall n \in \mathbb{Z}. \tag{2.42}$$

Then, (2.41) can be rewritten

$$\Gamma^{(p+1)} : \mathrm{x} \mapsto \Lambda^{(p)}\left(\Gamma^{(p)}(\mathrm{x}) * \overline{\mathrm{v}}^{(p)} + b^{(p)}\right), \tag{2.43}$$

where $\Gamma^{(p)} : l^2_{\mathbb{R}}(\mathbb{Z}) \to l^2_{\mathbb{R}}(\mathbb{Z})$ and $\Lambda^{(p)} : l^2_{\mathbb{R}}(\mathbb{Z}) \to l^2_{\mathbb{R}}(\mathbb{Z})$ are defined on a space of infinite sequences, rather than fixed-length vectors as before.

Two-Dimensional CNNs. In computer vision, input datapoints and convolution kernels are 2D sequences, denoted by roman, non-bold capital letters: $\mathrm{X}, \mathrm{V} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$. Inputs X and their corresponding outputs

$$(\mathrm{X} * \overline{\mathrm{V}})[\boldsymbol{n}] := \sum_{\boldsymbol{p}\in\mathbb{Z}^2} \mathrm{X}[\boldsymbol{p}]\,\mathrm{V}[\boldsymbol{p}-\boldsymbol{n}] \qquad \forall \boldsymbol{n} \in \mathbb{Z}^2, \tag{2.44}$$

are referred to as *feature maps*.

Subsampling. In CNNs, convolutions are sometimes performed with subsampling, also called *stride*. Considering $\mathrm{X}' := \mathrm{X} * \overline{\mathrm{V}} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ and a subsampling factor $m \in \mathbb{N} \setminus \{0\}$, we write $(\mathrm{X}' \downarrow m) \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ such that, for any $\boldsymbol{n} \in \mathbb{Z}^2$,

$$(\mathrm{X}' \downarrow m)[\boldsymbol{n}] := \mathrm{X}'[m\boldsymbol{n}]. \tag{2.45}$$

The support of the output feature maps is then roughly divided by m along each axis. In (2.35), subsampling is equivalent to keep one row out of m and discard all others.

Multichannel Convolution Layers. In many applications, convolution layers take multiple feature maps as input and output:

$$\mathbf{X} \in \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^{K_{\text{inp}}} \qquad \text{and} \qquad \mathbf{X}' \in \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^{K_{\text{out}}}, \tag{2.46}$$

where K_{inp} and $K_{\text{out}} \in \mathbb{N} \setminus \{0\}$ denote, for a specific layer, the number of input and output channels, respectively. Such objects are referred to as *multichannel tensors*. Besides, the layer is parameterized by a multichannel weight tensor and a bias vector

$$\mathbf{V} \in \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^{K_{\text{out}} \times K_{\text{inp}}} \qquad \text{and} \qquad \boldsymbol{b} \in \mathbb{R}^{K_{\text{out}}} \tag{2.47}$$

such that, for any output channel $l \in \{0 \,..\, K_{\text{out}} - 1\}$,

$$\mathrm{X}'_l := \sum_{k=0}^{K-1} (\mathrm{X}_k * \overline{\mathrm{V}}_{lk}) \downarrow m + b_l, \tag{2.48}$$

where we have introduced a bias notation, defined such that $(\mathrm{Y} + b)[\boldsymbol{n}] = \mathrm{Y}[\boldsymbol{n}] + b$ for any $\boldsymbol{n} \in \mathbb{Z}^2$.

For any $p \in \{0 \,..\, P-1\}$, let $K_p \in \mathbb{N} \setminus \{0\}$ denote the number of input channels in the $(p+1)$-th layer, and $L := K_P$ denote the number of output channels in the last layer. Note that $K := K_0$ denotes the number of input channels in the initial layer, *e.g.*, $K = 3$ for RGB input images. Each layer is associated with a weight tensor $\mathbf{V}^{(p)}$ and a bias vector $\boldsymbol{b}^{(p)}$, satisfying (2.47) with $K_{\text{inp}} \leftarrow K_p$ and $K_{\text{out}} \leftarrow K_{p+1}$. Then, the feature extractor mapping any input $\mathbf{X} \in \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^K$ to the output of the last convolution layer, denoted by

$$\boldsymbol{\Gamma}^{(P)} : \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^K \to \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^L, \tag{2.49}$$

is defined, like in Section 2.2.1, in a recursive manner. First, we define $\boldsymbol{\Gamma}^{(0)}$ as the identity. Then, for any $p \in \{0 \,..\, P-1\}$, the feature extractor $\boldsymbol{\Gamma}^{(p+1)} : \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^K \to \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^{K_{p+1}}$ related to the $p+1$ first layers is defined, for any channel $l \in \{0 \,..\, K_{p+1} - 1\}$, by

$$\Gamma^{(p+1)}_l : \mathbf{X} \mapsto \Lambda^{(p)} \left(\sum_{k=0}^{K_p - 1} \left(\Gamma^{(p)}_k(\mathbf{X}) * \overline{\mathrm{V}}^{(p)}_l \right) \downarrow m_p + b^{(p)}_l \right), \tag{2.50}$$

where $\Gamma^{(p)}_k : \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^K \to l^2_{\mathbb{R}}(\mathbb{Z}^2)$ denotes the k-th output component of $\boldsymbol{\Gamma}^{(p)}$, and $\Lambda^{(p)} : l^2_{\mathbb{R}}(\mathbb{Z}^2) \to l^2_{\mathbb{R}}(\mathbb{Z}^2)$ denotes a nonlinear activation function. Expression (2.50) is a two-dimensional, multichannel extension of (2.43).

Before the final classifier, an average value is computed over each feature map. We denote by

$$\boldsymbol{\mu} : \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^L \to \mathbb{R}^L \tag{2.51}$$

the corresponding operator, referred to as *adaptive average pooling*. We then get a variation of (2.28) for the global feature extractor:

$$\boldsymbol{\Gamma} := \boldsymbol{\mu} \circ \boldsymbol{\Gamma}^{(P)}, \qquad \text{with} \qquad \boldsymbol{\Gamma} : \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^K \to \mathbb{R}^L. \tag{2.52}$$

Finally, the feature vector $\boldsymbol{\Gamma}(\mathbf{X})$ serves as input for a final classifier, similar to (2.24):

$$\boldsymbol{y} := \boldsymbol{\Lambda}(\boldsymbol{V}\, \boldsymbol{\Gamma}(\mathbf{X}) + \boldsymbol{b}), \tag{2.53}$$

where $\boldsymbol{V} \in \mathbb{R}^{Q \times L}$ and $\boldsymbol{b} \in \mathbb{R}^Q$ are the trainable parameters of the final fully-connected layer, and $\boldsymbol{\Lambda}$ is the softmax activation function such as defined in (2.22).

2.3.2 Pooling Layers

Another key component of CNNs is pooling layers, which are usually placed between convolution layers, after the activation function. Their role is to aggregate information spread across a group of neighboring pixels into one single feature. Pooling layers therefore reduce dimensionality while decreasing the variability of signal representations.

A pooling layer operates on patches that are shifted along both dimensions by more than one pixel; it is therefore a downsampling operator. More formally, it is represented as a function $\Omega : l^2_{\mathbb{R}}(\mathbb{Z}) \to l^2_{\mathbb{R}}(\mathbb{Z}^2)$ satisfying

$$\Omega(\mathrm{X}) := \Omega_0(\mathrm{X}) \downarrow m', \tag{2.54}$$

where Ω_0 depends on the type of pooling, and m' denotes the subsampling factor (generally equal to 2). Early networks such as LeNet (LeCun et al., 1989) typically used *average pooling*, computed over a sliding grid of size $(2q+1) \times (2q+1)$ for a given $q \in \mathbb{N} \setminus \{0\}$ (generally equal to 1). Specifically, for any $\boldsymbol{n} \in \mathbb{Z}^2$,

$$\Omega_0^{\mathrm{avg}}(\mathrm{X})[\boldsymbol{n}] := \frac{1}{(2q+1)^2} \sum_{\|\boldsymbol{p}\|_\infty \leq q} \mathrm{X}[\boldsymbol{n}+\boldsymbol{p}]. \tag{2.55}$$

In the rest of the thesis, we refer to q as the grid's *half-size*. However, a nonlinear function, which became known as *max pooling*, was proposed by Yamaguchi et al. (1990) in the context of speech recognition, and by Riesenhuber and Poggio (1999) as a model for the cortex visual processing system. For any $\boldsymbol{n} \in \mathbb{Z}^2$,

$$\Omega_0^{\max}(\mathrm{X})[\boldsymbol{n}] := \max_{\|\boldsymbol{p}\|_\infty \leq q} \mathrm{X}[\boldsymbol{n}+\boldsymbol{p}]. \tag{2.56}$$

Max pooling has been widely adopted in CNN architectures since, as suggested by empirical evidence (Scherer et al., 2010), it leads to improved performance over linear pooling operators. Then, for any $p \in \{0 \,..\, P-1\}$, (2.50) becomes

$$\Gamma_l^{(p+1)} : \mathbf{X} \mapsto (\Omega \circ \Lambda^{(p)}) \left(\sum_{k=0}^{K_p - 1} \left(\Gamma_k^{(p)}(\mathbf{X}) * \overline{\mathrm{V}}_l^{(p)} \right) \downarrow m_p + b_l^{(p)} \right). \tag{2.57}$$

2.3.3 Advances in CNNs for Image Classification

The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) was organized annually from 2010 to 2017, and served as a benchmark for advances in large scale object classification and detection (Russakovsky et al., 2015). The first two editions were dominated by SVMs, used in combination with feature extractors such as SIFT or the Fisher kernel, as well as dimensionality reduction and data compression techniques (Y. Lin et al., 2011; Perronnin et al., 2010; Sanchez and Perronnin, 2011). The 2012 edition however was a turning point in the history of image classification, since CNNs came into play and won the challenge by a comfortable margin.

AlexNet: a Breakthrough in Computer Vision. To train their model, widely known as AlexNet, the laureates (Krizhevsky et al., 2017) took advantage of a parallel GPU implementation and introduced *dropout layers* before the final classifier. During

the training phase, each element of the input feature vector $\boldsymbol{\Gamma}(\mathbf{X})$, such as used in (2.53), is set to zero with a 50%-probability. Dropout serves as a regularization technique, therefore improving the network's generalization performances, at the cost of a doubled training time.

In the subsequent years, CNNs completely took over the competition. Zeiler and Fergus (2014) won the 2013 edition, using a visualization technique as a diagnosis tool to optimize Krizhevsky's architecture.

Inception Modules and 1×1 Convolution. The winners of the 2014 edition (Szegedy et al., 2015) beat the state of the art at that time, with a model named GoogLeNet.[1] It was built upon a stack of *inception* modules, which contain parallel convolutions layers (2.48) with various kernel sizes, in a similar way as multiresolution transforms (see Section 3.2). Moreover, dimensionality reduction is performed using convolution layers with kernels of size 1×1 (M. Lin et al., 2014). For such a layer, (2.50) becomes

$$\Gamma_l^{(p+1)} : \mathbf{X} \mapsto \Lambda^{(p)}\left(\sum_{k=0}^{K_p-1}\left(a_{lk}^{(p)}\,\Gamma_k^{(p)}(\mathbf{X})\right) + b_l^{(p)}\right), \tag{2.58}$$

with trainable weights $\boldsymbol{A}^{(p)} \in \mathbb{R}^{K_{p+1}\times K_p}$. Therefore, a *$1 \times 1$ convolution layer* performs linear combinations of feature maps with trainable weights. This trick allowed to increase the network width and depth at a reasonable cost. As a matter of fact, the number of trainable parameters was 12 times less than in AlexNet.

The same year, the VGG team, winner of the single-object localization track, ranked second in object classification with a very deep model using small 3×3 convolution kernels (Simonyan and Zisserman, 2015).

Residual Networks. Another major innovation came with the 2015 edition of ILSVRC. He et al. (2016) proposed a class of models called *residual networks*, also known as ResNets. This architecture introduces *skip connections* between groups of convolution layers. In its simplest form, a skip connection modifies the feature extractor at a given layer p, described in (2.50), as follows. We consider a situation where $K_p = K_{p+1}$. We also assume that $m_p = 1$ (subsampling factor). We then get, for any $l \in \{0\,..\,K_p - 1\}$,

$$\Gamma_l^{(p+1)} : \mathbf{X} \mapsto \Gamma_l^{(p)}(\mathbf{X}) + \Lambda^{(p)}\left(\sum_{k=0}^{K_p-1}\left(\Gamma_k^{(p)}(\mathbf{X}) * \overline{\mathrm{V}}_l^{(p)}\right)\downarrow m_p + b_l^{(p)}\right). \tag{2.59}$$

We can see that, if the convolution kernels and bias are set to 0, we get $\boldsymbol{\Gamma}^{(p+1)}(\mathbf{X}) = \boldsymbol{\Gamma}^{(p)}(\mathbf{X})$, and the $(p+1)$-th layer is a simple identity. For this reason, the features extracted at this layer (right of the $+$ sign) are qualified as *residual*. This approach allowed the authors to build very deep models with no drop in accuracy, as often observed in non-residual architectures.

Besides, ResNets perform a linear transform called *batch normalization* (Ioffe and Szegedy, 2015) after each convolution layer. This step ensures that intermediate feature maps have approximately zero mean and unit variance. Consequently, the network is less

[1] The name derives from LeCun's original model, which is usually referred to as LeNet.

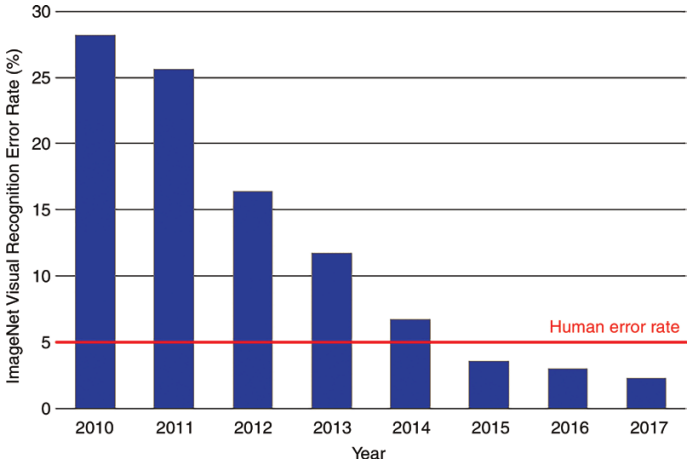


Figure 2.1. Top-5 error rates on the ILSVRC classification challenge. From Langlotz et al. (2019).

sensitive to parameter initialization, resulting in a faster training convergence. More on this is provided in Section 5.4.5.

The last two editions of ILSVRC awarded improved ResNet-like models; respectively, ResNeXt (S. Xie et al., 2017) and SENet (Hu et al., 2018). The evolution of the top-5 error rates from 2010 to 2017 is displayed in Figure 2.1. Outside ILSVRC, some very successful models were developed, mainly based on the above principles, with some novelty. A few examples among these are Inception-ResNet (Szegedy et al., 2017), DenseNet (Huang et al., 2017), WideResNet (Zagoruyko and Komodakis, 2017), and EfficientNet (Tan and Le, 2019).

Light-Weight CNNs. Despite impressive results, CNNs are very resource demanding, in terms of data transfer, memory usage as well as computing power. In some contexts such as mobile and embedded vision applications, the need for light weight yet competitive networks is growing. For this reason, efforts were made to design models to meet this demand. Iandola et al. (2016) developed SqueezeNet, in which the number of trainable parameters is optimized with extensive use of 1×1 convolution layers such as described in (2.58). They reached AlexNet accuracy with significantly fewer parameters. Later, Howard et al. (2017) proposed MobileNet. Following ideas developed in a previous network called Xception (Chollet, 2017), the authors built a very light architecture based on *depthwise-separable* convolution layers. In this context, $K_p = K_{p+1}$, and $\mathrm{V}_{lk} = 0$ if $l \neq k$, where the weight tensor $\mathbf{V}$ has been introduced in (2.47). We denote, for any $l \in \{0\,..\,K_p - 1\}$, $\widetilde{\mathrm{V}}_l := \mathrm{V}_{ll}$. Then, (2.50) becomes

$$\Gamma_l^{(p+1)} : \mathbf{X} \mapsto \Lambda^{(p)}\left(\left(\Gamma_l^{(p)}(\mathbf{X}) * \overline{\widetilde{\mathrm{V}}}_l^{(p)}\right) \downarrow m_p + b_l^{(p)}\right). \tag{2.60}$$

Besides, as in GoogLeNet/Inception networks (Szegedy et al., 2015), the number of channels is increased or reduced by using 1×1 convolution layers (2.58). More recent work, such as ShuffleNet (J. Zhang and J. Zhang, 2018) and MobileNetV2 (Sandler et al., 2018), proposed improvements over similar principles.

Complex-Valued CNNs. In parallel, a family of *complex-valued* CNNs (ℂVCNNs) saw a significant growth of interest in the recent years. The idea of complex-valued neural networks, which is the focus of a comprehensive survey (C. Lee et al., 2022), dates back from the early nineteens, when Kim and Guest (1990), Leung and Haykin (1991), and Georgiou and Koutsougeras (1992) adapted the backpropagation algorithm to the complex domain. First applied to recurrent neural networks (Kataoka et al., 1998),[2] the concept was more recently adapted to the CNN framework (see below). These networks implement convolution layers with complex-valued kernels, as well as complex extensions of the traditional activation and pooling layers. They are well suited for tasks requiring the phase information to be propagated through the entire network, as done in the context of magnetic resonance imaging (Dedmari et al., 2018), polarimetric imaging (Z. Zhang et al., 2017) or audio signals (Trabelsi et al., 2018). However, for image recognition tasks, ℂVCNNs do not seem to perform better than standard CNNs, with equal number of trainable parameters (Trabelsi et al., 2018).

Also based on complex convolutions, wavelet scattering networks (ScatterNets) (Bruna and Mallat, 2013), for which a description is provided in Section 3.4, are more adapted to image recognition. However, unlike ℂVCNNs, ScatterNets do not propagate the phase information and use the modulus operator as an activation function instead. Therefore, their properties and domains of application are significantly different from the above models. Mathematical motivations for employing complex-valued convolutions in deep learning architectures have been introduced by Tygert et al. (2016).

2.3.4 Beyond CNNs

The last few years have seen a shift of paradigm in computer vision, inspired by models developed in other research areas. Attention mechanisms have been widely used among the natural language processing (NLP) community, in particular since Bahdanau et al. (2016). The main idea is to model dependencies between input features in a non-local way, in contrast to the CNN strategy. When combined with RNNs, attention allows focusing on specific parts of the source, thus avoiding heavy computations on the whole signal. Going further, Vaswani et al. (2017) claimed that attention mechanisms alone suffice to produce state-of-the-art results on machine translation tasks, while significantly improving training time and scalability. They designed a new type of feedforward neural network called *transformer*, where hidden layers are based on non-local *self-attention* modules. This was a breakthrough in the field of natural language processing, giving birth to powerful algorithms such as BERT (Devlin et al., 2019) or GPT-3 (Brown et al., 2020). In its simplest form, a self-attention module takes as input a matrix $\boldsymbol{X} \in \mathbb{R}^{N\times D}$, whose row vectors $\boldsymbol{x}_n \in \mathbb{R}^D$ constitute a set of local *embeddings* converted from an input sequence or image. From $\boldsymbol{X}$, three intermediate matrices, referred to as *keys*, *queries* and *values*, are generated through linear transformations with trainable weight matrices $\boldsymbol{V}_\mathrm{k}$, $\boldsymbol{V}_\mathrm{q}$ and $\boldsymbol{V}_\mathrm{v}$. They satisfy, respectively,

$$\boldsymbol{K} := \boldsymbol{X}\boldsymbol{V}_\mathrm{k} \in \mathbb{R}^{N\times D_\mathrm{k}}; \qquad \boldsymbol{Q} := \boldsymbol{X}\boldsymbol{V}_\mathrm{q} \in \mathbb{R}^{N\times D_\mathrm{k}}; \qquad \boldsymbol{U} := \boldsymbol{X}\boldsymbol{V}_\mathrm{v} \in \mathbb{R}^{N\times D'}, \tag{2.61}$$

[2] Recurrent neural networks (RNNs) form another family of neural networks, initially developed by Rumelhart et al. (1986), and later improved by Hochreiter and Schmidhuber (1997) (*long short-term memory*, or LSTM). RNNs are designed for processing input sequences with variable lengths, and have been widely used in tasks such as handwriting or speech recognition.

with D_{k}, $D' \in \mathbb{N} \setminus \{0\}$. Then, a weighted average is performed between the N row vectors $\boldsymbol{u}_n \in \mathbb{R}^{D'}$ of $\boldsymbol{U}$, using an *attention matrix*

$$\boldsymbol{A} := \text{Softmax}(\boldsymbol{Q}\boldsymbol{K}^\top / \sqrt{D_{\text{k}}}) \in \mathbb{R}^{N \times N}, \tag{2.62}$$

where the softmax operator, described in (2.22), is applied to each row vector. The module's output is then defined by

$$\boldsymbol{Y} := \boldsymbol{A}\boldsymbol{U} \in \mathbb{R}^{N \times D'}. \tag{2.63}$$

The attention matrix, which contains inner products between keys and queries, models dependencies between any two embeddings $\boldsymbol{x}_n$, $\boldsymbol{x}_{n'} \in \mathbb{R}^D$, with $n, n' \in \{0 \,..\, N-1\}$.

From the computer vision perspective, among the many attempts to combine CNNs with self-attention or other types of non-local operations, we can cite X. Wang et al. (2018), and Carion et al. (2020). Meanwhile, Ramachandran et al. (2019) designed a ResNet-like model in which all convolution layers are replaced by self-attention modules. They outperformed the baseline on ImageNet-1K classification task with increased computational efficiency and 29% fewer trainable parameters. However, it was not until recently that Dosovitskiy et al. (2021) transposed the actual transformer model from NLP to computer vision. When pretrained on large image datasets, the corresponding network, referred to as *vision transformer* (ViT), achieves excellent results on ImageNet-1K and small image datasets with significantly fewer computational resources than CNNs. As of today, transformers in computer vision are a rapidly growing area of research.

2.4 Feature Extraction Properties in CNNs

It is widely assumed that a good feature extractor must retain discriminant image components while decreasing intra-class variability (LeCun et al., 1998; Bruna and Mallat, 2013). In particular, information about feature frequencies and orientations should be captured by the operator (LeCun et al., 1998; Liao and Peng, 2010; Bruna and Mallat, 2013). On the other hand, extracted features should remain consistent under transformations including small shifts, rotations, deformations or scaling (Liao and Peng, 2010; Sifre and Mallat, 2013; Bruna and Mallat, 2013; Bietti and Mairal, 2017; Wiatowski and Bölcskei, 2018). We refer the reader to Section 3.4.1 for a formal definition of stability under deformations. This requirement is motivated by the principle that similarly-appearing input images should yield similar classifications. This section reviews current knowledge about feature extraction properties in the early layers of CNNs, and whether the above requirements are satisfied.

2.4.1 Gabor-Like Patterns in Trained Convolution Kernels

When studying the response of primary visual cortex of cats and monkeys at various points along the visual pathway, Hubel and Wiesel (1959, 1962, 1968) identified two types of cells, respectively qualified as *simple* and *complex*. Each cell responds to input light signals, with intensity varying with their position in the field of view. Later, Marĉelja (1980), Daugman (1980), and Jones and Palmer (1987) discovered that simple cells are responsive to input signals with specific positions, orientations and frequencies. In other words, their receptive fields belong to a class of 2D filters analogous to Gabor filters.

Speculating that the visual cortex generates sparse representations of input signals, Olshausen and Field (1996) proposed a sparse coding algorithm for natural images, in an attempt to reproduce the natural behavior of simple cells with an artificial neural network. The objective was to find a set of atoms—also called basis functions—from which input images could be recovered with minimal information loss and maximal sparsity. After the training phase, the learned atoms took the appearance of oriented waveforms at different frequencies and positions, resulting in receptive fields similar to those found in the primary visual cortex of mammals. More details on this are provided in Section 3.1.3. Building on this, further work proposed to learn sparse representations using an overcomplete basis set (Olshausen and Field, 1997), an energy-based encoder-decoder (Ranzato et al., 2006), or supervised dictionary learning (Mairal et al., 2008b). However, the learned atoms may take on more complex or specialized structures that are tailored to the type of signal being modeled, rather than simply consisting of simple oscillating or oriented patterns. On the other hand, Bell and Sejnowski (1997) proposed an unsupervised learning algorithm based on information maximization, producing visual filters which, again, are localized and oriented.

It turns out that the first layer of CNNs trained on image datasets such as ImageNet (Russakovsky et al., 2015) exhibits a similar behavior: their receptive fields often take the appearance of Gabor-like structures with well-defined frequencies and orientations (Yosinski et al., 2014; Rai and Rivas, 2020). Therefore, they produce responses to input signals that are analogous to simple cells in the visual cortex. This was first evidenced by H. Lee et al. (2009) in the context of the *convolutional deep belief network*, a generative model built after G. E. Hinton et al. (2006). This phenomenon can be evidenced by visualizing the learned convolution kernels, as done by Krizhevsky et al. (2017) for AlexNet. Subsequent architectures such as ResNet are no exception. Figure 1.1 displays the convolution kernels of AlexNet's first layer after training with ImageNet. The model is provided by the Torchvision package belonging to PyTorch's ecosystem (Paszke et al., 2017).

These observations, conducted in both natural and artificial contexts, suggest that a wavelet or Gabor transform may be a good candidate to produce sparse signal representations which are useful for pattern recognition and classification. Chapter 3 provides theoretical insights to support this hypothesis. On the other hand, does it grant stability to deformations, introduced before as a fundamental requirement? To address this question, we need to delve deeper into the properties of wavelet transforms. Section 2.4.2 provides a partial answer—in general, subsampled convolution layers are unstable to translations. However, the relation between convolution and max pooling layers remains poorly understood. In Chapter 4, which is one of the main contributions of this thesis, we show that, under specific conditions, max pooling can partially restore shift invariance. More specifically, we unravel a connection between the operator "Conv $\rightarrow$ MaxPool" on the one hand, and "ComplexConv $\rightarrow$ Modulus" on the other hand, which produces stable image representations.

2.4.2 CNNs are Generally Not Shift Invariant

It is generally assumed that CNNs are translation invariant. Namely, if we consider two input images $\mathbf{X}$ and $\mathbf{X}'$ that are slightly shifted with respect to each other, we expect the corresponding feature vectors $\boldsymbol{\Gamma}(\mathbf{X})$ and $\boldsymbol{\Gamma}(\mathbf{X}') \in \mathbb{R}^L$, where $\boldsymbol{\Gamma}$ has been defined in (2.52), to be nearly identical. This however is not true. This misconception takes its roots

in the equivariance property of convolutions in the continuous domain (*i.e.*, convolutions commute with translations). In this context, shifting an input will result in an equally shifted output. Near shift invariance is then obtained by decreasing the resolution of feature maps, as done in the pooling layers. In particular, Wiatowski and Bölcskei (2018) showed that a wide variety of models becomes progressively more shift invariant with increasing network depth.

However, one key element is often omitted from studies on CNNs: practical implementations are based on *discrete* feature maps and parameters. As mentioned by Azulay and Weiss (2019) and R. Zhang (2019), both convolution and pooling layers may greatly diverge from shift invariance. This is due to a phenomenon called *aliasing*, which is caused by subsampling. To get an intuition on this behavior, an example is provided in Section 3.2.6. Therefore, feature vectors in standard CNNs are not shift invariant, which could penalize the network's accuracy and generalization capability.

To overcome this, R. Zhang (2019) proposed an antialiased version of CNNs based on low-pass filtering, called *blur pooling* (BlurPool). This operator, defined by

$$\Omega_0^{\text{blur}}(\mathrm{X}) := \mathrm{X} * \overline{\mathrm{B}}, \tag{2.64}$$

where $\mathrm{B} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ denotes a low-pass *blurring* filter, is used in two situations.

(1) Max pooling layers (MaxPool, decomposed into Max $\to$ Sub)[3] are replaced by max-blur pooling (Max $\to$ Blur $\to$ Sub). In this context, Ω_0^{max}, such as introduced in (2.56), is replaced by
$$\Omega_0^{\text{maxbl}} := \Omega_0^{\text{blur}} \circ \Omega_0^{\text{max}}. \tag{2.65}$$

(2) Convolution layers followed by ReLU (Conv $\to$ Sub $\to$ ReLU) are blurred before subsampling (Conv $\to$ ReLU $\to$ Blur $\to$ Sub). Note that ReLU is computed before blurring; otherwise the network would simply perform on low-resolution images.

This approach follows a well-known practice in signal processing, which involves low-pass filtering a high-frequency signal before subsampling, in order to avoid artifacts in reconstruction. It improved shift invariance as well as accuracy of various networks including AlexNet, ResNet, DenseNet and MobileNet. Interesting results were also obtained on corrupted datasets such as ImageNet-C (Hendrycks and Dietterich, 2019), as well as tiny image datasets such as CIFAR-10 (Krizhevsky and G. Hinton, 2009). However, this was achieved with a significant loss of information. The question of designing a non-destructive antialiasing method was tackled by X. Zou et al. (2023). They proposed an adaptive antialiasing approach, called *adaptive blur pooling*, which predicts separate filter weights for each spatial location and output channel. This allows preserving—to some extent—high-frequency information, resulting in improved accuracy. Yet, this approach remains fundamentally based on low-pass filtering, simply avoiding antialiasing where high-frequency information is needed. High-frequency features therefore remain unstable to translations. Furthermore, the two above approaches come at the cost of increased computational cost, memory consumption and, for the latter, higher number of trainable parameters. In Chapter 5, we propose an alternative antialiasing approach based on complex-valued convolutions, extracting high-frequency features that are stable to translations. We observed improved accuracy, with significant advantages in terms of computational efficiency and

[3]Sub and Conv stand for "convolution" and "subsampling," respectively.

memory usage. Our approach is actually rooted in the theoretical study presented in Chapter 4.

Other works addressed the topic of improving translation invariance in CNNs. In particular, Chaman and Dokmanic (2021) reached perfect shift invariance by using an adaptive, input-dependent subsampling grid, whereas the previous models based on blur pooling rely on fixed grids. This idea was harnessed by J. Xu et al. (2021) to get shift equivariance in generative models. Although this method satisfied shift invariance for integer-pixel translations, it did not address the problem of shift instability for fractional-pixel translations. Besides, it may still lead to aliasing artifacts that could negatively impact network accuracy and generalization performance. Combining this approach with blur pooling actually produced the highest classification scores on ResNet trained with ImageNet. This indicates that antialiasing remains beneficial, even after achieving perfect shift invariance on the pixel grid. Another aspect of shift invariance in CNNs is related to boundary effects. The fact that CNNs can encode the absolute position of an object in the image by exploiting boundary effects was discovered independently by Islam et al. (2020), and Kayhan and Gemert (2020). The study of this phenomenon is outside the scope of this thesis. Shift invariance was further studied by J. Lee et al. (2020), from both local and global points of view. A visualization technique based on cosine similarities was used. The authors showed that Zhang's antialiasing method improved local shift invariance but not global one, which is rather affected by boundary effects.

To complete this discussion, Section 3.5.2 provides a comprehensive overview of theoretical studies on CNNs, bridging the gap between deep learning and wavelet analysis.

2.4.3 Focus on the Max Pooling Layer

Scherer et al. (2010) compared the performances of CNNs with average (subsampled low-pass filtering) versus nonlinear max pooling (see Section 2.3.2), and found that the latter outperforms the former. According to the authors, the goal of pooling layers is to improve invariance by decreasing the size of feature maps. Overlapping pooling windows may also help achieving this purpose. However, as explained in Section 2.4.2, max pooling in particular is not shift invariant, due to aliasing. As explained before, a deeper analysis of the properties of this operator, when used in combination with Gabor-like convolutions, is presented in Chapter 4 as one of the main contributions of this thesis.

2.4.4 Concluding Remarks

To summarize, CNNs are a powerful tool for many computer vision tasks including image classification. They contain heavily-parameterized feature extractors that exploit the geometry of images. In particular, the learned kernels in initial convolution layers spontaneously adopt Gabor-like structures, which extract discriminative feature at various frequencies and orientations. The properties of convolutions with oriented band-pass filters are examined in Chapter 3, from the point of view of sparse image representations. Despite their strengths, discrete subsampled convolutions with band-pass filters are unstable to translations, a topic which is specifically addressed in this thesis.

Chapter 3

Background on Wavelet Analysis

We have seen in Chapter 2 that CNNs are intended to transform input images into "useful" representations for classification. We have also identified some desired properties for such representations. In particular, the first convolution layer, when trained on image datasets, spontaneously transform inputs into wavelet or Gabor-like representations, thus achieving frequency and directional selectivity. In this chapter, we cover theoretical aspects of multiresolution analysis, and discrete wavelet transforms in particular, with a focus on two-dimensional applications. We shall see that there are very good reasons for why such image transformations are observed in CNNs, and why they have been used in many other domains outside machine learning. Finally, we will present some applications in which wavelet transforms have been *explicitly* used in the context of image classification.

3.1 Sparse Representations of Images

One possible angle of approach for wavelet analysis is to consider the problem of sparse coding. More specifically, we seek a family of elementary atoms, or basis functions, such that any input signal from a given class can be reconstructed using a limited number of atoms. Moreover, we want the signal reconstruction to be as close as possible from the original signal. A sparse coding effectively captures the most prominent features in a given class of signal (*e.g.*, pikes, edges, corners, textures, *etc.*). It converts raw input data into richer signal representations which, in turn, can be used as input for deeper feature extractors as in CNNs, before feeding a linear classifier.

The present dissertation is mainly focused on two-dimensional images, which contain localized and oriented features at various scales. In their simplest form, they can be modeled as piecewise smooth signals with one-dimensional discontinuities (Vetterli, 2001). In Section 3.1.1, we introduce a mathematical formulation of the sparse coding problem. Then, in Section 3.1.2, we present two approaches to solve it and introduce wavelet bases as a good candidate for nonlinear sparse coding in an orthonormal basis. Finally, in Section 3.1.3, the concept of dictionary learning is outlined, which involves learning the basis functions from the data rather than relying on handcrafted ones, thus eliminating the requirement for expert knowledge. Sections 3.1 and 3.2 are principally based on Vetterli (2001) and Mallat (2009).

3.1.1 General Problem Formulation

Let $\mathcal{I} \subset l^2_{\mathbb{R}}(\mathbb{Z}^2)$ denote a set of images. Given a set of indices $\mathcal{B}$, we consider a dictionary of (possibly complex) two-dimensional sequences, called *basis functions*, denoted by $\mathrm{E}_n \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ for any $n \in \mathcal{B}$. We denote by $\mathcal{J} := \operatorname{span}(\mathrm{E}_n)_{n\in\mathcal{B}}$ the subspace of $l^2_{\mathbb{C}}(\mathbb{Z}^2)$ spanned by the basis functions, and assume that $\mathcal{I} \subset \mathcal{J}$. We now consider a mapping (or feature extractor, as seen in Chapter 2), denoted by

$$\boldsymbol{\Gamma} : \mathcal{I} \to \mathbb{C}^{\mathcal{B}}, \tag{3.1}$$

and the corresponding projection ("image reconstruction") on $\mathcal{J}$:

$$\Pi : \mathrm{X} \mapsto \sum_{n\in\mathcal{B}} \Gamma_n(\mathrm{X})\, \mathrm{E}_n, \tag{3.2}$$

where we have denoted $\mathbf{E} := (\mathrm{E}_n)_{n\in\mathcal{B}}$ and $\boldsymbol{\Gamma} := (\Gamma_n)_{n\in\mathcal{B}}$. The goal is to describe input images in terms of basis function coefficients, which can be seen as the "building blocks" of our image set. By rearranging information in a certain way, such image representation, when chosen properly, may be much more useful than a simple pixel description.

One possible way to select a "proper" feature extractor is by minimizing a certain error function $\mathcal{E} : (\mathcal{I},\, \mathbb{C}^{\mathcal{B}},\, \mathcal{J}^{\mathcal{B}}) \to \mathbb{R}_+$. For any $\mathrm{X} \in \mathcal{I}$,

$$\boldsymbol{\Gamma}(\mathrm{X}) := \operatorname*{argmin}_{\boldsymbol{y}\in\mathbb{C}^{\mathcal{B}}} \mathcal{E}(\mathrm{X},\, \boldsymbol{y},\, \mathbf{E}). \tag{3.3}$$

For instance, one may want to minimize the Euclidean distance between an input image and its projection on $\mathcal{J}$. In this case, $\mathcal{E}$ is defined as the mean squared error (MSE):

$$\mathcal{E} : (\mathrm{X}, \boldsymbol{y}, \mathbf{E}) \mapsto \left\| \mathrm{X} - \sum_{n\in\mathcal{B}} y_n\, \mathrm{E}_n \right\|_2^2. \tag{3.4}$$

If, moreover, $(\mathrm{E}_n)_{n\in\mathcal{B}}$ is an orthonormal family, then we get

$$\Gamma_n : \mathrm{X} \mapsto \langle \mathrm{X},\, \mathrm{E}_n \rangle \qquad \forall n \in \mathcal{B}, \tag{3.5}$$

and Π is simply the orthonormal projection on $\mathcal{J}$.

In many tasks such as compression, denoising, pattern recognition or inverse problems, it can be interesting to get a sparse description of input images. In other words, we want information to be captured by a small number $M \ll \operatorname{card}(\mathcal{B})$ of nonzero coefficients $\Gamma_n(\mathrm{X})$. The corresponding projection $\Pi(\mathrm{X})$ is called an *M-term approximation*. Assuming $(\mathrm{E}_n)_{n\in\mathcal{B}}$ is an orthonormal basis, a possible approach is to perform linear projection on a fixed subspace spanned by a carefully-chosen subset of $(\mathrm{E}_n)_{n\in\mathcal{B}}$ indexed by $\mathcal{B}_M \subset \mathcal{B}$, such that $M := \operatorname{card}(\mathcal{B}_M) \ll \operatorname{card}(\mathcal{B})$. In this case, (3.5) becomes

$$\Gamma_n : \mathrm{X} \mapsto \begin{cases} \langle \mathrm{X},\, \mathrm{E}_n \rangle & \text{if } n \in \mathcal{B}_M; \\ 0 & \text{elsewhere.} \end{cases} \tag{3.6}$$

Alternatively, a sparse representation can be obtained by selecting a set of support basis functions (*i.e.*, associated with nonzero coefficients) in an adaptive, nonlinear fashion. In this context, the minimization problem (3.3) becomes

$$\boldsymbol{\Gamma}(\mathrm{X}) := \operatorname*{argmin}_{\|\boldsymbol{y}\|_0 \le M} \mathcal{E}(\mathrm{X},\, \boldsymbol{y},\, \mathbf{E}), \tag{3.7}$$

where $\|\boldsymbol{y}\|_0$ denotes the number of nonzero coefficients ("l^0-norm"). In the general case where $(\mathrm{E}_n)_{n\in\mathcal{B}}$ is an over-complete (redundant) dictionary, this problem is NP-hard. However, there exists greedy algorithms which compute non-optimal yet efficient approximations, such as matching pursuit (Mallat and Z. Zhang, 1993). In the more restrictive scenario where $(\mathrm{E}_n)_{n\in\mathcal{B}}$ is an orthonormal basis, we can show that the feature extractor $\boldsymbol{\Gamma}$ satisfying (3.7) is defined by

$$\Gamma_n : \mathrm{X} \mapsto \begin{cases} \langle \mathrm{X}, \mathrm{E}_n \rangle & \text{for the } M \text{ largest values of } |\langle \mathrm{X}, \mathrm{E}_n \rangle|; \\ 0 & \text{elsewhere,} \end{cases} \tag{3.8}$$

which is different from the linear feature extractors defined in (3.5) and (3.6). Therefore, Π linearly projects X on an adaptive, input-dependent subspace of $\mathcal{I}$. As such, Π is a nonlinear approximation function.

We now address the problem of choosing an orthonormal basis $(\mathrm{E}_n)_{n\in\mathcal{B}}$ which yields sparse image representations with good reconstruction properties. Then, we extend the problem to more generic dictionaries $(\mathrm{E}_n)_{n\in\mathcal{B}}$, which are possibly redundant and non-orthogonal, and introduce the concept of *dictionary learning* (Mairal et al., 2008b).

3.1.2 Sparse Coding in Orthonormal Bases

Let $(\mathrm{E}_n)_{n\in\mathcal{B}}$ denote an orthonormal family of $l^2_{\mathbb{C}}(\mathbb{Z}^2)$, and $\mathcal{J} := \operatorname{span}(\mathrm{E}_n)_{n\in\mathcal{B}}$. For any $\mathrm{X} \in \mathcal{I}$, the reconstruction error $\mathcal{E}(\mathrm{X}, \boldsymbol{\Gamma}(\mathrm{X}), \mathbf{E})$ is denoted by $\mathcal{E}^{\mathrm{l}}_M(\mathrm{X})$ for linear approximations (3.6), and $\mathcal{E}^{\mathrm{nl}}_M(\mathrm{X})$ in the nonlinear setting (3.8). On the one hand, we have:

$$\mathcal{E}^{\mathrm{l}}_M(\mathrm{X}) = \sum_{n \notin \mathcal{B}_M} |\langle \mathrm{X}, \mathrm{E}_n \rangle|^2 . \tag{3.9}$$

On the other hand, the nonlinear feature extractor $\boldsymbol{\Gamma}$ satisfying (3.7) selects the M largest inner products and sets all others to 0, as written in (3.8). For a given image $\mathrm{X} \in \mathcal{I}$, we denote by $\mathcal{B}^{\mathrm{nl}}_M(\mathrm{X}) \subset \mathcal{B}$ the set of M selected basis function, such that $\Gamma_n(\mathrm{X}) = 0$ for any $n \notin \mathcal{B}^{\mathrm{nl}}_M(\mathrm{X})$. Then, the reconstruction error is equal to the residual energy:

$$\mathcal{E}^{\mathrm{nl}}_M(\mathrm{X}) = \sum_{n \notin \mathcal{B}^{\mathrm{nl}}_M(\mathrm{X})} |\langle \mathrm{X}, \mathrm{E}_n \rangle|^2 , \tag{3.10}$$

which strongly depends on the choice of orthonormal basis. The "optimal basis", in turn, depends on the class of input images and its distribution. Note that, by design, $\mathcal{E}^{\mathrm{nl}}_M(\mathrm{X}) \leq \mathcal{E}^{\mathrm{l}}_M(\mathrm{X})$ for any $\mathrm{X} \in \mathcal{I}$.

In this section, we explore several such bases and introduce wavelet bases and their properties. We assume that input images $\mathrm{X} \in \mathcal{I}$ are supported in a fixed square grid of size $N \times N$, denoted by $\mathcal{B}_N \subset \mathbb{Z}^2$. We also denote by $\mathcal{I}_N \subset l^2_{\mathbb{C}}(\mathbb{Z}^2)$ the subspace of (complex-valued) images supported on $\mathcal{B}_N$. We therefore have $\mathcal{I} \subset \mathcal{I}_N$. To alleviate notations, in this section we discard all indices outside $\mathcal{B}_N$.

Pixel Basis. For two-dimensional images, the trivial basis of $\mathcal{I}_N$ is made of N^2 "pixel atoms", indexed by $\mathcal{B} := \mathcal{B}_N$. More precisely, for any $\boldsymbol{n} \in \mathcal{B}_N$,

$$\mathrm{E}_{\boldsymbol{n}}[\boldsymbol{p}] := \delta_{\boldsymbol{np}} \qquad \forall \boldsymbol{p} \in \mathcal{B}_N. \tag{3.11}$$

Then, we simply have $\langle \mathrm{X}, \mathrm{E}_{\boldsymbol{n}} \rangle = \mathrm{X}[\boldsymbol{n}]$. The sparse coding $\boldsymbol{\Gamma}(\mathrm{X})$ therefore consists in keeping the M highest-energy pixels of X and setting all the others to 0. We can easily see that, if energy is spread across the image, then the residual energy (3.10) will be very high if $M \ll N^2$. Therefore, the pixel basis is ill-suited for sparse coding of most signal families.

Fourier Basis. At the opposite end of the scope, we can encode images in the frequency domain. To this end, we consider the *Fourier basis* which, as before, is indexed by $\mathcal{B} := \mathcal{B}_N$. For any $\boldsymbol{n} \in \mathcal{B}_N$,

$$\mathrm{E}_{\boldsymbol{n}}[\boldsymbol{p}] := \frac{1}{N} \exp\left(\frac{2i\pi \langle \boldsymbol{n}, \boldsymbol{p} \rangle}{N}\right) \qquad \forall \boldsymbol{p} \in \mathcal{B}_N. \tag{3.12}$$

Then, for any $\boldsymbol{n} \in \mathcal{B}_N$, $\langle \mathrm{X}, \mathrm{E}_{\boldsymbol{n}} \rangle$ is obtained by computing the *discrete Fourier transform* of X at discrete frequency $\boldsymbol{n}$. The sparse coding $\boldsymbol{\Gamma}(\mathrm{X})$ therefore consists in keeping the M largest Fourier coefficients of X (in terms of modulus), and setting all others to 0.

If input images are realizations of a smooth stationary process, then the discrete Fourier transform provides a powerful framework to encode most of their energy over a small number of coefficients. Moreover, the discrete Fourier transform is computed with an efficient algorithm called the *fast Fourier transform* (FFT), with complexity $O(N^2 \log(N))$. The situation is comparable to 1D signals such as time-invariant musical sequences. For example, a three-note chord played on a piano (*e.g.*, C major triad) will be mainly encoded into coefficients associated with frequencies around 523 Hz, 659 Hz and 784 Hz, as well as their harmonics. Setting to 0 the frequencies lying in between will not change much the reconstructed signal.

However, the discrete Fourier transform is ill-suited to encode localized events, because the corresponding basis functions themselves are nonlocal. Therefore, encoding a single event requires many nonzero coefficients. This scenario is very common in full musical sequences, but also in 2D images, which generally contain local discontinuities.

Wavelet Bases. We have seen that the pixel basis is well-suited to encode punctual events whereas the Fourier basis does exactly the opposite: encoding regular signals without any sharp transition (*e.g.*, textures, time-invariant musical sequences, *etc.*). Therefore, neither of them seem appropriate for images, which, as mentioned in the introduction of Section 3.1, can be modeled as piecewise smooth signals with local irregularities. What is actually needed is something in between: a basis able to encode both spatial and frequency features. This is where wavelets come into play. The atoms of a wavelet basis are made of localized oscillating waveforms, characterized by a certain frequency and spatial localization. Such a basis spawns sparse representation of piecewise regular signals, as will be explained in Section 3.2.3. Actually, there exists a tradeoff between spatial and frequency resolutions, referred to as the *Heisenberg's uncertainty principle*: any given wavelet basis function satisfies

$$\sigma^2_{\mathrm{spat}} \times \sigma^2_{\mathrm{freq}} \geq \frac{1}{4}, \tag{3.13}$$

where σ^2_{spat} and σ^2_{freq} respectively denote the functions's spatial (or time) and frequency variance in a given direction (Mallat, 2009, p. 44). In a way, the pixel and Fourier bases represent the two extreme configurations, with either perfect spatial or frequency resolution.

Wavelets take their roots in time-frequency analysis. Working on sound waves, and inspired by quantum physics, Gabor (1946) designed a signal decomposition over waveforms which minimize the time-frequency support (3.13). These waveforms are made of a sliding Gaussian window modulated by a complex exponential with varying frequencies:

$$\psi_{u,\,\nu} : t \mapsto \phi(t-u)\,\mathrm{e}^{i\nu t}, \tag{3.14}$$

where u and $\nu \in \mathbb{R}$ respectively denote the time and frequency around which input signals are analyzed, and $\phi \in L^2_{\mathbb{R}}(\mathbb{R})$ denotes a normalized Gaussian window of fixed size. Gabor showed that such a decomposition along both time and frequency axes resemble the human perception of sounds. Forty years later, Grossmann and Morlet (1984) laid the foundation for wavelet analysis, by designing an invertible continuous transform over translations and dilations of a single Gabor waveform $\psi_{0,\,1}$, called *Morlet wavelet* in this context. While the Gabor transform initially relied on a fixed window size, in the *continuous wavelet transform* (CWT), the window size is decreased when frequency increases. This was initially motivated by the need for a proper separation of closely-spaced geophysical waves at high frequencies.

In Section 3.2, we will introduce multiresolution discrete wavelet transforms, which come with a fast decomposition algorithm called the *fast wavelet transform* (FWT). Before that, we introduce an algorithm by Olshausen and Field (1996) to learn optimal basis functions from data, which ultimately led to the concept of dictionary learning. We shall see that wavelet or Gabor-like patterns spontaneously arise when learning basis functions on landscape images, using nonsmooth regularization.

3.1.3 Toward Dictionary Learning

In this section, we no longer restrict to orthonormal bases, but instead seek an overcomplete dictionary. Rather than manually selecting a dictionary and searching for an optimal sparse coding within that family, a data-driven approach can be adopted where the basis functions $(\mathrm{E}_n)_{n\in\mathcal{B}}$ are learned directly from the data, thereby removing the need for handcrafted design. Olshausen and Field (1996) were among the first to propose learning basis functions while enforcing sparsity. For this purpose, they introduced a nonsmooth regularization term to the error function (3.4), for instance the l^1-norm—as independently proposed by Tibshirani (1996) in the context of *lasso regression*:

$$\mathcal{E}_\lambda : (\mathrm{X}, \boldsymbol{y}, \mathbf{E}) \mapsto \mathcal{E}(\mathrm{X}, \boldsymbol{y}, \mathbf{E}) + \lambda \left\|\boldsymbol{y}\right\|_1 . \tag{3.15}$$

Note that the l^1-norm is used in the regularization term, instead of the l^0-"norm" as employed in (3.7). Thanks to this choice, the minimization problem remains convex and is therefore easier to solve. The goal is to minimize $\mathcal{E}_\lambda$ with respect to both $\boldsymbol{y}$ and $\mathbf{E}$. In this problem, the total number of basis functions $\mathrm{card}(\mathcal{B})$ is much greater than the number of support basis functions M, which is controlled by the regularization hyperparameter λ. Again, we restrict to real-valued basis functions.

The algorithm proposed by Olshausen and Field (1996) can be summarized as follows. First, $(\mathrm{E}_n)_{n\in\mathcal{B}}$ is randomly initialized, and normalized such that $\|\mathrm{E}_n\|_2^2 = 1$ for any $n \in \mathcal{B}$. Then, at each iteration, the two following steps are performed on a minibatch $\mathcal{I}' \subset \mathcal{I}$.

(1) Compute, for any $\mathrm{X} \in \mathcal{I}'$, the representation $\boldsymbol{y} := \boldsymbol{\Gamma}(\mathrm{X}) \in \mathbb{R}^{\mathcal{B}}$ minimizing the regularized error function $\mathcal{E}_\lambda$ defined in (3.15).

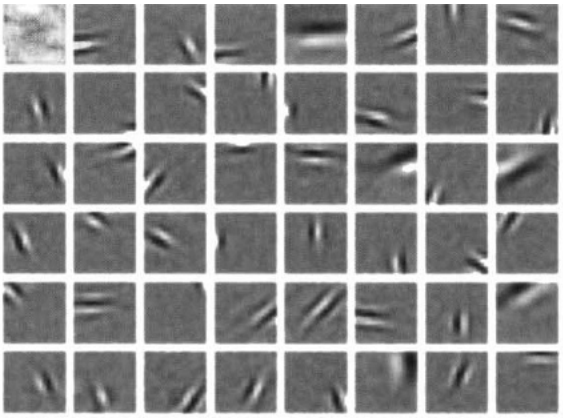


Figure 3.1. Subset of 48 basis functions from $(\mathrm{E}_n)_{n\in\mathcal{B}}$ ($\mathrm{card}(\mathcal{B}) = 192$), learned from 4 000 patches of size 16×16 extracted from natural scenes, using the two-step algorithm presented in Section 3.1.3. Figure from Olshausen and Field (1996).

(2) Compute the gradient of $\mathcal{E}_\lambda(\mathrm{X}, \boldsymbol{\Gamma}(\mathrm{X}), \mathbf{E})$ with respect to E_n, averaged over all images in $\mathrm{X} \in \mathcal{I}'$ in the minibatch. Then, update the basis functions $(\mathrm{E}_n)_{n\in\mathcal{B}}$ with stochastic gradient descent.

Figure 3.1 displays the basis functions obtained using the above algorithm, trained on patches of size 16×16 extracted from landscape images ($\mathrm{card}(\mathcal{B}) = 192$). We notice that $(\mathrm{E}_n)_{n\in\mathcal{B}}$ are localized, and present oriented, band-pass Gabor-like structures. Besides, the corresponding receptive fields resemble those of mammalian visual cortex, as explained in Section 2.4.1. Therefore, wavelet-like basis functions seem to provide a useful framework for image representations, from the perspective of natural vision as well as sparse coding.

Intuitively, an analogy with natural languages may help to understand why redundancy can be exploited to improve sparsity. A natural language is fundamentally redundant, offering many different possibilities to express a single idea, with more or less sparsity. For example, a "road vehicle for freight transportation" is usually referred to as a "truck". The concept could have been accurately described without the latter word, but using it is of great help for efficient communication. Moreover, different languages or dialects may have different degrees of redundancy in a given lexical field, depending on their usage, culture, geographical area, period of history, *etc.* Similarly, in image processing, selecting an appropriate dictionary of basis functions can play an important role in achieving optimal performance for a given task and class of images. The chosen dictionary must indeed effectively encode features which are abundant in this specific class and provide relevant information for the task at hand. For example, a dictionary that has been trained on a dataset of natural landscape images will be better suited for image classification on this category of images, rather than a general-purpose dictionary. In particular, this property has been successfully exploited for various applications such as color image restoration (Mairal et al., 2008a), and more generally led to the development of dictionary learning (Donoho and Elad, 2003; Aharon et al., 2006; Mairal et al., 2008b, 2009; Chabiron et al., 2015). More information on the link between sparsity and redundancy in image processing can be found in a work by Elad et al. (2010). Redundancy will be exploited in Section 3.3 in the context of complex wavelet transforms. More specifically, images will be decomposed in a redundant frame, a concept initially developed by Duffin and Schaeffer (1952).

3.2 Filter Bank Decomposition and Wavelet Bases

In this section, we start by describing a class of orthonormal bases of $l^2_{\mathbb{R}}(\mathbb{Z}^2)$, based on discrete invertible filter banks (FBs) (Vetterli, 1986). Then, we explain the link with wavelet bases in the continuous domain. Mallat (1989) showed that such a FB decomposition encodes a *fast wavelet transform* (FWT), which approximates any function $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ into a wavelet basis, at an arbitrarily high resolution.

3.2.1 Filter Bank Decomposition with Perfect Reconstruction

We could, as in Section 3.1.2, work on the subspace of images supported on $\mathcal{B}_N$ (square grid of size $N \times N$). However, this would require the use of circular convolutions. Instead, we will recursively build orthonormal bases for the entire space $l^2_{\mathbb{R}}(\mathbb{Z}^2)$. As we shall see, decomposing images in such bases is done with standard convolutions, as in CNNs.

Quadrature Mirror Filters. We consider a pair of low- and high-pass 1D orthogonal filters $\mathrm{h}, \mathrm{g} \in l^2_{\mathbb{R}}(\mathbb{Z})$ satisfying the following requirements:

$$\widehat{\mathrm{h}}(0) = \sqrt{2}; \qquad \text{(normalized low-pass filter)} \tag{3.16}$$

$$\forall \omega \in [-\pi, \pi], \left|\widehat{\mathrm{h}}(\omega)\right|^2 + \left|\widehat{\mathrm{h}}(\omega + \pi)\right|^2 = 2; \qquad \text{(orthogonality condition)} \tag{3.17}$$

$$\forall \omega \in [-\pi, \pi], \widehat{\mathrm{g}}(\omega) = \mathrm{e}^{-i\omega}\, \widehat{\mathrm{h}}^*(\omega + \pi), \qquad \text{(quadrature mirror relationship)} \tag{3.18}$$

with the "+" sign being defined modulo (2π), and where we have considered the *discrete-time Fourier transform* $l^2_{\mathbb{C}}(\mathbb{Z}) \to L^2_{\mathbb{C}}([-\pi, \pi])$, defined, for any $\mathrm{x} \in l^2_{\mathbb{C}}(\mathbb{Z})$, by

$$\widehat{\mathrm{x}}(\omega) := \sum_{n \in \mathbb{Z}} \mathrm{x}[n] \mathrm{e}^{-i\omega n} \qquad \forall \omega \in [-\pi, \pi]. \tag{3.19}$$

The filters h and g are referred to as *quadrature mirror filter* (QMFs). Then, we build a separable 2D filter bank (FB), containing a low-pass filter G_0 and three high-pass filters $\mathrm{G}_{1-3} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$, defined by

$$\mathrm{G}_0 := \mathrm{h} \otimes \mathrm{h}; \qquad \mathrm{G}_1 := \mathrm{h} \otimes \mathrm{g}; \qquad \mathrm{G}_2 := \mathrm{g} \otimes \mathrm{h}; \qquad \mathrm{G}_3 := \mathrm{g} \otimes \mathrm{g}, \tag{3.20}$$

where $\otimes$ denotes the tensor (or outer) product.

Filter Bank Decomposition. To start with, we consider the input space $\mathcal{J}_0^{(0)} := l^2_{\mathbb{R}}(\mathbb{Z}^2)$, spanned by the pixel basis $\mathbf{E}_0^{(0)} := \left(\mathrm{E}^{(0)}_{0,\boldsymbol{n}}\right)_{\boldsymbol{n} \in \mathbb{Z}^2}$ defined, similarly to (3.11), by

$$\mathrm{E}^{(0)}_{0,\boldsymbol{n}}[\boldsymbol{p}] := \delta_{\boldsymbol{np}} \qquad \forall \boldsymbol{p} \in \mathbb{Z}^2. \tag{3.21}$$

Considering an input image X, we initialize the decomposition with the feature map of pixel coefficients $\mathrm{D}_0^{(0)} := \mathrm{X}$.

The filter bank decomposition is a recursive algorithm. At any stage $j+1 \in \mathbb{N} \setminus \{0\}$, we consider a subspace $\mathcal{J}_0^{(j)} \subset l^2_{\mathbb{R}}(\mathbb{Z}^2)$, spanned by the basis $\mathbf{E}_0^{(j)}$ generated at previous

stage j. We denote by $\mathrm{D}_0^{(j)}$ the corresponding feature map of coefficients, encoding the linear projection of X on $\mathcal{J}_0^{(j)}$. Then, $\mathcal{J}_0^{(j)}$ is split into four orthogonal subspaces

$$\mathcal{J}_0^{(j)} = \bigoplus_{k=0}^{3} \mathcal{J}_k^{(j+1)}. \tag{3.22}$$

For any $k \in \{0\,..\,3\}$, $\mathcal{J}_k^{(j+1)}$ is spanned by the orthonormal basis $\mathbf{E}_k^{(j+1)}$, defined, for any $\boldsymbol{p} \in \mathbb{Z}^2$, by

$$\mathrm{E}_{k,\boldsymbol{p}}^{(j+1)} = \sum_{\boldsymbol{n}\in\mathbb{Z}^2} \mathrm{G}_k[\boldsymbol{n} - 2\boldsymbol{p}]\, \mathrm{E}_{0,\boldsymbol{n}}^{(j)}, \tag{3.23}$$

where the filters G_k have been introduced in (3.20). $\mathcal{J}_0^{(j+1)}$ is a subspace of low-resolution images whereas $\mathcal{J}_{1-3}^{(j+1)}$ are detail subspaces, respectively containing horizontal, vertical and residual (corners) features at scale j. For each $k \in \{0\,..\,3\}$, the feature map encoding the linear projection on $\mathcal{J}_k^{(j+1)}$, denoted by $\mathrm{D}_k^{(j)}$, is computed with the following subsampled convolution:

$$\mathrm{D}_k^{(j+1)} = (\mathrm{D}_0^{(j)} * \overline{\mathrm{G}_k}) \downarrow 2, \tag{3.24}$$

where the 2D convolution product and subsampling operator have been respectively defined in (2.44) and (2.45). We remind the reader that $\overline{\mathrm{G}_k}[\boldsymbol{n}] := \mathrm{G}_k[-\boldsymbol{n}]$ denotes the "flipped" sequence. Then, the low-resolution subspace $\mathcal{J}_0^{(j+1)}$ is in turn decomposed, following the above procedure. The algorithm stops after reaching the desired number of stages $J > 0$—referred to as *decomposition depth*.

We denote by $\uplus$ the concatenation operator. For any depth $J \in \mathbb{N} \setminus \{0\}$, the family

$$\mathbf{E}^{(J)} := \biguplus_{j=1}^{J} \biguplus_{k=1}^{3} (\mathrm{E}_{k,\boldsymbol{n}}^{(j)})_{\boldsymbol{n}\in\mathbb{Z}^2} \uplus (\mathrm{E}_{0,\boldsymbol{n}}^{(J)})_{\boldsymbol{n}\in\mathbb{Z}^2} \tag{3.25}$$

is an orthonormal basis of $l^2_{\mathbb{R}}(\mathbb{Z}^2)$, which we refer to as a *discrete wavelet basis*. In this context, the sparse coding $\boldsymbol{\Gamma}$ defined in (3.6) (linear setting) or (3.8) (nonlinear setting) is reindexed, and satisfy

$$\Gamma_{k,\boldsymbol{n}}^{(j)}(\mathrm{X}) := \begin{cases} \langle \mathrm{X},\, \mathrm{E}_{k,\boldsymbol{n}}^{(j)}(\mathrm{X}) \rangle = \mathrm{D}_k^{(j)}[\boldsymbol{n}] & \text{if } n \in \mathcal{B}_M \text{ or } \mathcal{B}_M^{\mathrm{nl}}(\mathrm{X}), \text{ respectively;} \\ 0 & \text{elsewhere.} \end{cases} \tag{3.26}$$

A visual representation of discrete wavelet basis functions is provided in Figure 3.2.

Quality of the Approximation. To investigate the potential of filter bank decompositions for approximating input images with a limited number of coefficients, mathematical properties of both images and filters must be characterized. As digital images are discrete representations of continuous light signals, a characterization in the continuous domain is typically more appropriate, and easier to establish. To this end, Section 3.2.2 provides a bond between analog and digital signal processing by linking discrete FB decompositions and continuous wavelet bases. Then, Section 3.2.3 presents the main results on the sparsity of discrete wavelet representations.

3.2.2 Link with Wavelets Defined on the Continuous Domain

In this section, we present Mallat's results (2009, Chap. 7) in the 2D case. In a nutshell, if we consider $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ as the discrete sampling (*e.g.*, digital acquisition) of a continuous image $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ (*e.g.*, the analog light signal captured by the sensor of a digital camera), then the feature maps $\mathrm{D}_k^{(j)}$ introduced in Section 3.2.1 encode the projection of F in an orthonormal wavelet basis. It contains the translated and dilated versions of three mother wavelets Ψ_{1-3} (encoding horizontal, vertical and residual features at various scales), plus the translated versions of a scaling function Φ at the coarsest scale (encoding the remaining low-frequency features).

Multiresolution Approximations. Let $\Phi \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ denote a low-pass *scaling function*, satisfying

$$\|\Phi\|_{L^2} = 1 \qquad \text{and} \qquad \widehat{\Phi}(\mathbf{0}) = 1, \tag{3.27}$$

where we have considered the two-dimensional *Fourier transform* $L^2_{\mathbb{C}}(\mathbb{R}^2) \to L^2_{\mathbb{C}}(\mathbb{R}^2)$, defined, for any $F \in L^2_{\mathbb{C}}(\mathbb{R}^2)$, by

$$\forall \boldsymbol{\xi} \in \mathbb{R}^2,\ \widehat{F}(\boldsymbol{\xi}) := \iint_{\mathbb{R}^2} F(\boldsymbol{x}) \mathrm{e}^{-i\langle \boldsymbol{\xi}, \boldsymbol{x} \rangle}\, \mathrm{d}^2\boldsymbol{x}. \tag{3.28}$$

We consider an initial projection subspace

$$\mathcal{V} := \mathrm{span}(\Phi_{\boldsymbol{n}})_{\boldsymbol{n} \in \mathbb{Z}^2} \subset L^2_{\mathbb{R}}(\mathbb{R}^2), \tag{3.29}$$

spanned by a family of translated versions of Φ defined by

$$\Phi_{\boldsymbol{n}} : \boldsymbol{x} \mapsto \Phi(\boldsymbol{x} - \boldsymbol{n}). \tag{3.30}$$

We assume $(\Phi_{\boldsymbol{n}})_{\boldsymbol{n} \in \mathbb{Z}^2}$ to be an orthonormal basis of $\mathcal{V}$. We now consider, for any $j \in \mathbb{Z}$, an orthonormal family, denoted by $\left(\Phi_{\boldsymbol{n}}^{(j)}\right)_{\boldsymbol{n} \in \mathbb{Z}^2}$ of dilated and translated scaling functions at scale j:

$$\Phi_{\boldsymbol{n}}^{(j)} : \boldsymbol{x} \mapsto \frac{1}{2^j} \Phi\left(\frac{\boldsymbol{x} - 2^j \boldsymbol{n}}{2^j}\right) \qquad \forall \boldsymbol{n} \in \mathbb{Z}^2. \tag{3.31}$$

We also introduce

$$\mathcal{V}^{(j)} := \mathrm{span}\left(\Phi_{\boldsymbol{n}}^{(j)}\right)_{\boldsymbol{n} \in \mathbb{Z}^2} \subset L^2_{\mathbb{R}}(\mathbb{R}^2). \tag{3.32}$$

We assume that, for any $j \in \mathbb{Z}$, $\mathcal{V}^{(j+1)} \subset \mathcal{V}^{(j)}$. If, moreover,

$$\lim_{j \to +\infty} \mathcal{V}^{(j)} = \{0\} \qquad \text{and} \qquad \lim_{j \to -\infty} \mathcal{V}^{(j)} = L^2_{\mathbb{R}}(\mathbb{R}^2), \tag{3.33}$$

then the sequence of subspaces $\left(\mathcal{V}^{(j)}\right)_{j \in \mathbb{Z}}$ is called a *multiresolution analysis* (Mallat, 1989). The linear projection of any continuous 2D signal $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ on $\mathcal{V}^{(j)}$ approximates F at a resolution which is decreased by a factor 2 when j is incremented.

QMFs and Wavelet Bases. Let (h, g) denote a pair of QMFs satisfying (3.16)–(3.18), and G_{0-3} the corresponding 2D filter bank such as defined in (3.20). If

$$\inf_{|\omega|\leq\pi/2} |\widehat{\mathrm{h}}(\omega)| > 0, \tag{3.34}$$

then, Mallat's theorem (1989) implies that there exists a unique multiresolution analysis $(\mathcal{V}^{(j)})_{j\in\mathbb{Z}}$ such that, for any $j \in \mathbb{Z}$ and any $\boldsymbol{p} \in \mathbb{Z}^2$,

$$\Phi_{\boldsymbol{p}}^{(j+1)} := \sum_{\boldsymbol{n}\in\mathbb{Z}^2} \mathrm{G}_0[\boldsymbol{n} - 2\boldsymbol{p}]\, \Phi_{\boldsymbol{n}}^{(j)}, \tag{3.35}$$

where $(\Phi_{\boldsymbol{n}}^{(j)})_{\boldsymbol{n}\in\mathbb{Z}^2}$ denotes the orthonormal basis of $\mathcal{V}^{(j)}$ (3.32). In other words, G_0 contains the coordinates of the dilated and translated scaling function $\Phi_{\boldsymbol{p}}^{(j+1)}$ (3.31) in the approximation space of finer resolution $\mathcal{V}^{(j)}$ (3.32). Recall that, by definition of a multiresolution analysis, $\mathcal{V}^{(j+1)} \subset \mathcal{V}^{(j)}$; therefore, $\Phi_{\boldsymbol{p}}^{(j+1)} \in \mathcal{V}^{(j)}$. Furthermore, we consider, for any $k \in \{1\,..\,3\}$, $j \in \mathbb{Z}$ and $\boldsymbol{p} \in \mathbb{Z}^2$,

$$\Psi_{k,\boldsymbol{p}}^{(j+1)} := \sum_{\boldsymbol{n}\in\mathbb{Z}^2} \mathrm{G}_k[\boldsymbol{n} - 2\boldsymbol{p}]\, \Phi_{\boldsymbol{n}}^{(j)}. \tag{3.36}$$

Then, analogous to (3.31), we get a family of translated and dilated wavelets:

$$\Psi_{k,\boldsymbol{n}}^{(j)} : \boldsymbol{x} \mapsto \frac{1}{2^j}\, \Psi_k\left(\frac{\boldsymbol{x} - 2^j\boldsymbol{n}}{2^j}\right), \tag{3.37}$$

where the mother wavelet $\Psi_k := \Psi_{k,\boldsymbol{0}}^{(0)}$ satisfies

$$\|\Psi_k\|_{L^2} = 1 \qquad \text{and} \qquad \widehat{\Psi}_k(\boldsymbol{0}) = 0. \tag{3.38}$$

Now, for any $k \in \{1\,..\,3\}$ and any $j \in \mathbb{Z}$, we introduce the detail subspace

$$\mathcal{V}_k^{(j)} := \operatorname{span}(\Psi_{k,\boldsymbol{n}}^{(j)})_{\boldsymbol{n}\in\mathbb{Z}^2} \subset L^2_{\mathbb{R}}(\mathbb{R}^2). \tag{3.39}$$

Then, for any $j \in \mathbb{Z}$, $\mathcal{V}_0^{(j)} := \mathcal{V}^{(j)}$ (3.32) can be decomposed into four orthogonal subspaces

$$\mathcal{V}_0^{(j)} = \bigoplus_{k=0}^{3} \mathcal{V}_k^{(j+1)}, \tag{3.40}$$

where $\mathcal{V}_0^{(j+1)} := \mathcal{V}^{(j+1)}$ denotes the subspace of coarser resolution satisfying (3.32) with $j \leftarrow j+1$, and where $\mathcal{V}_{1-3}^{(j+1)}$ denotes the three detail subspaces (3.39), respectively encoding horizontal, vertical and residual features.

Finally, for any depth $J \in \mathbb{N} \setminus \{0\}$, the family

$$\boldsymbol{\Psi}^{(J)} := \biguplus_{j=1}^{J} \biguplus_{k=1}^{3} (\Psi_{k,\boldsymbol{n}}^{(j)})_{\boldsymbol{n}\in\mathbb{Z}^2} \uplus (\Phi_{\boldsymbol{n}}^{(J)})_{\boldsymbol{n}\in\mathbb{Z}^2} \tag{3.41}$$

is an orthonormal basis of $\mathcal{V}$, referred to as a *wavelet basis*. As evidenced in (3.44), it is the continuous counterpart of the discrete basis $\mathbf{E}^{(J)}$ as defined in (3.25). Figure 3.2 provides a visual representation of discrete and continuous basis functions satisfying Section 3.3.3 and (3.36), respectively.

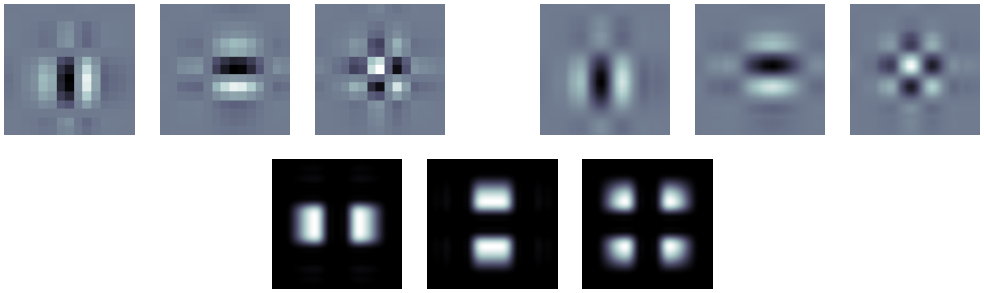


Figure 3.2. Left: discrete basis functions $\mathrm{E}^{(j)}_{k,\mathbf{0}}$ (3.23) with depth $j = 2$ and $k = 1\,..\,3$, computed with Q-shift orthogonal QMFs of length 10 (see Section 3.3.3). Right: continuous basis functions (wavelets) $\Psi^{(j)}_{k,\mathbf{0}}$ (3.36). Bottom: modulus of the discrete-time Fourier transform of $\mathrm{E}^{(j)}_{k,\mathbf{0}}$ (the origin is located at the center).

Remark 3.1. The mother scaling function Φ and wavelets $\Psi_{1-3} \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ are the tensor products of a one-dimensional scaling function ϕ with a wavelet $\psi \in L^2_{\mathbb{R}}(\mathbb{R})$:

$$\Phi := \phi \otimes \phi; \qquad \Psi_1 := \phi \otimes \psi; \qquad \Psi_2 := \psi \otimes \phi; \qquad \Psi_3 := \psi \otimes \psi. \tag{3.42}$$

Any pair of QMFs (h, g) satisfying (3.16)–(3.18) and (3.34) can therefore be associated with a pair of scaling function and wavelet (ϕ, ψ).

Remark 3.2. According to (3.33), we can get an orthonormal basis of the whole space $L^2_{\mathbb{R}}(\mathbb{R}^2)$ if j starts from $-\infty$. Besides, we can get rid of the scaling functions $\left(\Phi^{(J)}_{\boldsymbol{n}}\right)_{\boldsymbol{n}\in\mathbb{Z}^2}$ if J goes to $+\infty$.

The Fast Wavelet Transform (FWT). Let $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ denote a 2D continuous signal. We consider $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ encoding the linear projection of F on $\mathcal{V}$:

$$\forall \boldsymbol{n} \in \mathbb{Z}^2,\ \mathrm{X}[\boldsymbol{n}] := \langle F, \Phi_{\boldsymbol{n}} \rangle\,. \tag{3.43}$$

Then, for any scale $j \in \mathbb{N} \setminus \{0\}$, we can show that the linear projection of F on the subspaces $\mathcal{V}^{(j)}_{0-3}$ are encoded by the FB feature maps introduced in Section 3.2.1:

$$\forall k \in \{0\,..\,3\}\,,\ \forall \boldsymbol{n} \in \mathbb{Z}^2,\ \mathrm{D}^{(j)}_k[\boldsymbol{n}] = \langle F, \Psi^{(j)}_{k,\boldsymbol{n}} \rangle, \tag{3.44}$$

where we have denoted, for the sake of conciseness, $\Psi^{(j)}_{0,\boldsymbol{n}} := \Phi^{(j)}_{\boldsymbol{n}}$. Note that (3.43) and (3.44) can be written as convolution products:

$$\mathrm{X}[\boldsymbol{n}] = (F * \overline{\Phi})(\boldsymbol{n}) \qquad \text{and} \qquad \mathrm{D}^{(j)}_k[\boldsymbol{n}] = (F * \overline{\Psi}^{(j)}_{k,\mathbf{0}})(2^j \boldsymbol{n}), \tag{3.45}$$

where $\overline{\Phi} : \boldsymbol{x} \mapsto \Phi(-\boldsymbol{x})$ denotes a "flipped" function. Therefore, any element of $\mathcal{V}$ can be decomposed in the orthonormal basis $\boldsymbol{\Psi}^{(J)}$ by performing the FB decomposition described in Section 3.2.1. It is performed with separable convolutions in $O(N_{\mathrm{f}} \times N^2)$, where N_{f} denotes the support size of the QMFs h and g, and N denotes the support size of X. As such, it is called *fast wavelet transform* (FWT).

Remark 3.3. To maintain consistency with the notations used throughout this thesis, in its present formulation, the FWT algorithm is initialized at $j = 0$ (corresponding to the initial approximation scale $2^j = 1$). This differs from Mallat's book (2009, p. 301), where j typically starts at $L < 0$ such that $2^L = N^{-1}$, with N being the number of discretization points along the vertical or horizontal axis (*i.e.*, the support size of X). It should be noted that this change is a matter of convention and does not affect the content of this chapter.

Why the Continuous Framework Matters. Digital signal processing as well as CNNs exclusively deal with discrete signals. Therefore, one can wonder what are the benefits of adopting a purely theoretical continuous point of view. One motivation is to provide a physical interpretation to FWT, since discrete signals are themselves an abstract representation of the real world. Besides, mathematical properties of input images and wavelets (see Section 3.2.3), as well as stability results such as Lipschitz continuity to deformations (see Section 3.4.1) are established in the continuous domain. The unravelled bond between the two worlds is therefore a powerful tool to extend these results to the field of digital signal processing.

In Chapter 4, we will adopt a slightly different approach to link discrete and continuous image processing. Instead of considering input images X as a uniform sampling of continuous functions F using (3.43), we replace the scaling function Φ by the *Shannon scaling function* Φ^{sh}, which is independent of the choice of QMFs (h, g). It is defined by $\Phi^{\mathrm{sh}} := \mathrm{sinc}_\pi \otimes \mathrm{sinc}_\pi$, where $\mathrm{sinc}_\pi : x \mapsto \sin(\pi x)/(\pi x)$ denotes the normalized sinc function. According to Mallat (2009, Theorem 3.5, p. 68), the family $(\Phi^{\mathrm{sh}}_{\boldsymbol{n}})_{\boldsymbol{n}\in\mathbb{Z}^2}$ is an orthonormal basis of $\mathcal{V}^{\mathrm{sh}}$, which is a subspace of functions with compact support in the Fourier domain:

$$\mathcal{V}^{\mathrm{sh}} := \left\{ F \in L^2_{\mathbb{R}}(\mathbb{R}^2) \;\middle|\; \operatorname{supp} \widehat{F} \subset [-\pi,\, \pi]^2 \right\}. \tag{3.46}$$

Note that we actually use the 2D formulation, mentioned in p. 82.

3.2.3 Sparsity of Wavelet Representations

This section examines the ability of wavelet bases to encode input signals using a small number of nonzero coefficients. As we shall see, the sparsity of signal representations can be increased by choosing wavelets with a large number of *vanishing moments* and small support (Mallat, 2009, p. 284). For each class of signals under consideration, we compare the asymptotic behavior of the linear and nonlinear reconstruction errors, $\mathcal{E}^{\mathrm{l}}_M(\mathrm{X})$ (3.9) and $\mathcal{E}^{\mathrm{nl}}_M(\mathrm{X})$ (3.10), as the number of terms M approaches infinity. In the linear framework, the set of selected indices $\mathcal{B}_M$ corresponds to the largest values of j (*i.e.*, the coarsest scales), as they produce the largest-amplitude wavelet coefficients, and therefore yield the smallest reconstruction error $\mathcal{E}^{\mathrm{l}}_M(\mathrm{X})$ in average.

One-Dimensional Framework

In the one-dimensional setting, we consider a pair of 1D scaling function and wavelet (ϕ, ψ), characterized by a pair of low- and high-pass QMFs (h, g), as explained in Remark 3.1. Let $f \in L^2_{\mathbb{R}}(\mathbb{R})$ denote an input signal, from which we can get a discrete sequence $\mathrm{x} \in l^2_{\mathbb{R}}(\mathbb{Z})$, such that $\mathrm{x}[n] := (f * \overline{\phi})(n)$, similar to (3.45).

Lipschitz-Continuous Functions. First, we consider the class of Lipschitz-continuous functions, characterized by a Lipschitz constant $\alpha > 0$. For any $u \in \mathbb{R}$, there exists a polynomial ϖ_u of degree $d \leq \lfloor \alpha \rfloor$ and a residual function ε_u such that, for any $t \in \mathbb{R}$,

$$f(t) = \varpi_u(t) + \varepsilon_u(t), \tag{3.47}$$

$$\text{with} \quad |\varepsilon_u(t)| \leq C|t-u|^\alpha \tag{3.48}$$

for a certain constant $C > 0$. Now, we assume that ψ has at least $q := \lfloor \alpha \rfloor + 1$ vanishing moments. By definition, ψ is orthogonal to any polynomial of degree $d < q$. This property can be characterized on the low-pass filter h:

$$\forall d \in \{0\,..\,q-1\}\,,\, \widehat{\mathrm{h}}^{(d)}(\pi) = 0, \tag{3.49}$$

where $\widehat{\mathrm{h}}^{(d)}$ denotes the d-th derivative of $\widehat{\mathrm{h}}$, with the one-dimensional discrete-time Fourier transform being defined in (3.19). Then, for any $j \in \{1\,..\,J\}$ and $n \in \mathbb{Z}$, the dilated and translated wavelet $\psi_n^{(j)}$ also has q vanishing moment. Therefore, the wavelet coefficient $\mathrm{d}^{(j)}[n]$ only depends on the residual part of the function:

$$\mathrm{d}^{(j)}[n] := \langle \varepsilon_{2^j n},\, \psi_n^{(j)} \rangle, \tag{3.50}$$

where (3.47) has been applied with $u \leftarrow 2^j n$. According to (3.48), $\varepsilon_{2^j n}$ has small amplitude in a neighborhood around $2^j n$, if the Lipschitz constant α is large. Therefore, $\left|\mathrm{d}^{(j)}[n]\right|$ is small if the wavelet (which is centered around $2^j n$) is sufficiently compact in time. This occurs at finer scales, *i.e.*, for smaller values of j. Thus, the wavelet coefficients that have significant amplitudes correspond to large values of j, capturing slow variations of the input signal. Furthermore, coarser scales contain exponentially fewer basis functions, which leads to fast-decaying wavelet coefficients, and a sparse signal representation. We therefore get a polynomial decay for the reconstruction error, in both linear and nonlinear schemes:

$$\mathcal{E}_M^{\mathrm{l}}(\mathrm{x}) = O(M^{-2\alpha}) \qquad \text{and} \qquad \mathcal{E}_M^{\mathrm{nl}}(\mathrm{x}) = O(M^{-2\alpha}). \tag{3.51}$$

Piecewise Lipschitz-Continuous Functions. Lipschitz-continuous functions are too restrictive for many real-life scenarios. For instance, audio signals usually contain localized events that are characterized by sharp transitions between smoother regions. To model such signals, one can consider piecewise Lipschitz-continuous functions. In this case, the reconstruction error for linear M-term approximations no longer decays as fast as in the case of regular functions, because the signal contains high-frequency features that cannot be captured by the large-scale wavelet coefficients. However, a finite number of sharp transitions does not affect the asymptotic behavior of nonlinear M-term approximations, because the wavelet coefficients detecting these transitions, which are large at first, decay exponentially with increasing rank. All in all, we get

$$\mathcal{E}_M^{\mathrm{l}}(\mathrm{x}) = O(M^{-1}) \qquad \text{and} \qquad \mathcal{E}_M^{\mathrm{nl}}(\mathrm{x}) = O(M^{-2\alpha}). \tag{3.52}$$

Remark 3.4. Another important property of the mother wavelet ψ is its support size (or decay rate in the case of infinite support). This property has a direct impact on the number of wavelet coefficients that can detect an irregularity in the signal. A smaller support size generally leads to sparser signal representation, especially when it contains many irregularities. However, Daubechies (1988) showed that the number of vanishing moments is proportional to the size of the wavelet support. Therefore, a tradeoff must be found to produce efficient M-term approximations.

Bounded Variation Functions. Another class of signals that can be considered for functions that are not necessarily uniformly regular is *bounded-variations functions*, for which the linear and nonlinear approximation errors satisfy

$$\mathcal{E}_M^{\mathrm{l}}(\mathrm{x}) = O(M^{-1}) \qquad \text{and} \qquad \mathcal{E}_M^{\mathrm{nl}}(\mathrm{x}) = O(M^{-2}). \tag{3.53}$$

Two-Dimensional Framework

Lipschitz-Continuous Functions. The asymptotic behavior of the linear and nonlinear approximation errors is similar to the one-dimensional framework (3.51), except for a factor 2 in the exponent:

$$\mathcal{E}_M^{\mathrm{l}}(\mathrm{X}) = O(M^{-\alpha}) \qquad \text{and} \qquad \mathcal{E}_M^{\mathrm{nl}}(\mathrm{X}) = O(M^{-\alpha}). \tag{3.54}$$

Piecewise Lipschitz-Continuous Functions. Unfortunately, unlike in 1D, the asymptotic decay in $O(M^{-\alpha})$ of $\mathcal{E}_M^{\mathrm{nl}}(\mathrm{X})$ established in (3.54) no longer holds. This is because images have discontinuities along one-dimensional curves. Therefore, FWT will produce an important number of high-amplitude coefficients along these curves, which only decay in $O(M^{-1})$ and therefore remain dominant whatever the Lipschitz constant. We thus have

$$\mathcal{E}_M^{\mathrm{nl}}(\mathrm{X}) = O(M^{-1}). \tag{3.55}$$

Bounded Variation Functions. These functions include large classes of images which do not have irregular textures (Mallat, 2009, p. 467). As for Lipschitz-continuous functions, the asymptotic behavior of the linear and nonlinear approximation errors resembles the 1D case (3.53):

$$\mathcal{E}_M^{\mathrm{l}}(\mathrm{X}) = O(M^{-1/2}) \qquad \text{and} \qquad \mathcal{E}_M^{\mathrm{nl}}(\mathrm{x}) = O(M^{-1}). \tag{3.56}$$

Despite being not as well-behaved as when applied to piecewise Lipschitz-continuous 1D signals, the 2D FWT has led to successful applications, among which image denoising (Mallat and Hwang, 1992; Y. Xu et al., 1994; Donoho and Johnstone, 1995; Malfait and Roose, 1997), image restoration (Starck and Bijaoui, 1994), or the JPEG-2000 standard for image compression (Marcellin et al., 2000), principally based on the CREW algorithm (Zandi et al., 1995). Subsequent research was conducted to improve this asymptotic decay, leading to the design of geometric wavelets (Candès and Donoho, 1999, 2000; Do and Vetterli, 2000, 2005; Le Pennec and Mallat, 2005; Peyré and Mallat, 2008).

Remark 3.5. Although the Lipschitz constant does not affect the asymptotic behavior of the approximation error, having a large number of vanishing moments is still important for generating low-amplitude wavelet coefficients in smooth regions and achieving sparsity in image representations. Additionally, the size of the wavelet support remains an important property in two dimensions, as it affects the multiplicative constant that determines the rate at which the approximation error decays.

3.2.4 Examples of Wavelet Bases

As covered in Section 3.2.3, a good wavelet basis must be spanned by a wavelet with a high number of vanishing moments (high frequency localization) and a fast decay rate (high temporal or spatial localization). This section presents several historical approaches, and explains how the time-frequency tradeoff has been handled.

Two Precursors: the Haar and Shannon Wavelets. Long before Mallat's multiresolution theory, Haar (1911) constructed an orthonormal basis of $L^2_{\mathbb{R}}(\mathbb{R})$ by translating and dilating a very simple wavelet (although the designation did not exist at the time) with value 1 on $[0, 1/2]$ and -1 on $[1/2, 1]$ (0 everywhere else). The corresponding scaling function is the rectangular function. This first attempt perfectly fits the multiresolution framework. It is associated with the pair of QMFs (h, g) satisfying

$$\mathrm{h}[n] := \begin{cases} \sqrt{2}/2 & \text{if } n \in \{0,\, 1\}; \\ 0 & \text{otherwise.} \end{cases} \tag{3.57}$$

The Haar wavelet has a very compact support size, but only one vanishing moment. Seen from another angle, the Fourier transform of the Haar scaling function is the sinc function, which slowly decays in $O(1/\xi)$. Therefore, the Haar scaling function and wavelet are highly localized in space but not in frequency. As such, the Haar basis is ill-suited for sparse coding of piecewise Lipschitz-regular signals (see Section 3.2.3).

If we switch roles between spatial and Fourier domains, we get the Shannon basis, associated with a sinc scaling function (Shannon, 1949), whose Fourier transform is compactly-supported. The corresponding QMFs (h, g) satisfy, for any $\omega \in [-\pi,\, \pi]$,

$$\widehat{\mathrm{h}}(\omega) := \begin{cases} \sqrt{2} & \text{if } \omega \in [-\frac{\pi}{2},\, \frac{\pi}{2}]; \\ 0 & \text{otherwise.} \end{cases} \tag{3.58}$$

The Shannon wavelet has an infinite number of vanishing moments. However, it decays slowly in $O(1/t)$. The situation is therefore reversed compared to the Haar wavelet, making the Shannon basis a poor candidate for sparse coding of piecewise Lipschitz-regular signals.

Besides, the QMFs h and g slowly decay in $O(1/n)$, which makes them difficult to approximate in practice. It requires very large vectors to avoid numerical instabilities. From a computational point of view, FWT with large convolution kernels can be implemented in the Fourier domain by using the FFT algorithm. However, in CNNs, only spatial filters with relatively small size are implemented. Therefore, the Shannon basis is ill-suited to approximate the behavior of freely-trained networks, such as done in Section 5.4.2.

Several successful attempts to design a fast-decaying wavelet with a large number of vanishing moments have been made since the 1980s. Two major representatives are presented below.

Meyer Wavelets. Meyer (1985) designed a class of wavelet bases spanned by smooth wavelets (in both spatial and Fourier domains). The corresponding QMFs (h, g) satisfy, for any $\omega \in [-\pi,\, \pi]$,

$$\widehat{\mathrm{h}}(\omega) := \begin{cases} \sqrt{2} & \text{if } |\omega| \leq \pi/3; \\ 0 & \text{if } |\omega| \geq 2\pi/3. \end{cases} \tag{3.59}$$

Between $\pi/3$ and $2\pi/3$, $\widehat{\mathrm{h}}$ follows a smooth curve satisfying (3.17), such that $\widehat{\mathrm{h}}$ remains C^d at the junctions, for a given $d \in \mathbb{N}$.

As for Shannon, the Meyer wavelets are C^∞ with an infinite number of vanishing moments. However, unlike the Shannon wavelet, they decay at an exponential rate which increases with the smoothness d of $\widehat{\mathrm{h}}$. Therefore, Meyer bases are far more suitable for sparse coding than Haar or Shannon bases. Nevertheless, from a numerical point of view, this decay can remain slow, which can be an issue when relatively small filters are required as in CNNs.

Daubechies Wavelets with Compact Support. Instead of using QMFs with infinite support as in the Meyer case, Daubechies (1988) designed a family of compactly-supported wavelets, characterized by finitely-supported QMFs, also referred to as *finite impulse response* (FIR) filters. For any $q \in \mathbb{N} \setminus \{0\}$, a Daubechies wavelet with q vanishing moments is characterized by a pair of QMFs (h, g) satisfying, for any $\omega \in [-\pi, \pi]$,

$$\widehat{\mathrm{h}}(\omega) := \sqrt{2} \left(\frac{1 + \mathrm{e}^{-i\omega}}{2} \right)^q \varpi(\mathrm{e}^{-i\omega}), \tag{3.60}$$

where ϖ is a polynomial designed such that $\widehat{\mathrm{h}}$ satisfies the orthogonality condition (3.17). Note that, by design, $\widehat{\mathrm{h}}(\omega)$ and its $q-1$ first derivatives are equal to 0 at $\omega = \pi$, as stated in (3.49). Daubechies (1988) showed that, for a given number of vanishing moments q, the QMFs have at least $2q$ nonzero coefficients, whereas the corresponding wavelets have a minimal support size of $2q-1$. In fact, Daubechies wavelets have, by design, the smallest possible support size for a given q. This result is in line with Heisenberg's uncertainty principle introduced in (3.13): increasing the number of vanishing moments (*i.e.*, increased frequency localization) is done at the cost of an increased support size (*i.e.*, decreased spatial localization). However, the filter's energy remains concentrated in a small region of its support when q is large.

It is worth noticing that the Daubechies wavelet with $q = 1$ vanishing moment is nothing else that the Haar wavelet.

3.2.5 Discrete Wavelet Packet Transform

In multiresolution approximations, each subspace $\mathcal{J}_k^{(j)}$ (3.22) (discrete framework) or $\mathcal{V}_k^{(j)}$ (3.40) (continuous framework) is characterized by four symmetric Fourier square windows of size $\pi/2^j$, containing most of its energy (or even all energy for the Shannon basis). Therefore, the basis functions $\mathrm{E}_{k,\boldsymbol{n}}^{(j)} \in \mathbf{E}^{(J)}$ (3.25) or $\Psi_{k,\boldsymbol{n}}^{(j)} \in \boldsymbol{\Psi}^{(J)}$ (3.41) are less localized in frequency (and more localized in space) at finer scales (smaller values of j).

For some classes of input signals, one may want to increase Fourier resolution (and decrease spatial resolution) at high frequencies. To this end, Coifman and Wickerhauser (1992) generalized the link between wavelets and filter banks, by splitting the detail subspaces $\mathcal{J}_{1-3}^{(j)}$ or $\mathcal{V}_{1-3}^{(j)}$ into 4 orthogonal subspaces, similarly to (3.22) or (3.40). Then, for any $j \in \mathbb{N}$ and any $l \in \{0 \,..\, 4^j - 1\}$, we get

$$\mathcal{J}_l^{(j)} = \bigoplus_{k=0}^{3} \mathcal{J}_{4l+k}^{(j+1)} \qquad \text{and} \qquad \mathcal{V}_l^{(j)} = \bigoplus_{k=0}^{3} \mathcal{V}_{4l+k}^{(j+1)}. \tag{3.61}$$

Generalizing (3.23) (discrete framework) on the one hand, and (3.35), (3.36) (continuous framework) on the other hand, the basis functions satisfy, for any $k \in \{0 \,..\, 3\}$ and $\boldsymbol{p} \in \mathbb{Z}^2$,

$$\mathrm{E}_{4l+k,\boldsymbol{p}}^{(j+1)} = \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathrm{G}_k[\boldsymbol{n} - 2\boldsymbol{p}] \, \mathrm{E}_{l,\boldsymbol{n}}^{(j)} \qquad \text{and} \qquad \Psi_{4l+k,\boldsymbol{p}}^{(j+1)} = \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathrm{G}_k[\boldsymbol{n} - 2\boldsymbol{p}] \, \Psi_{l,\boldsymbol{n}}^{(j)}, \tag{3.62}$$

where we have denoted $\Psi_{0,\boldsymbol{n}}^{(j)} := \Phi_{\boldsymbol{n}}^{(j)}$. As a reminder, at $j = 0$, $(\mathrm{E}_{0,\boldsymbol{n}}^{(0)})_{\boldsymbol{n} \in \mathbb{Z}^2}$ denotes the pixel orthonormal basis of $\mathcal{J}_0^{(0)} := l^2_{\mathbb{R}}(\mathbb{Z}^2)$, such as introduced in (3.21), and $(\Psi_{0,\boldsymbol{n}}^{(0)})_{\boldsymbol{n} \in \mathbb{Z}^2} := (\Phi_{\boldsymbol{n}}^{(0)})_{\boldsymbol{n} \in \mathbb{Z}^2}$ denotes the orthonormal basis of the initial approximation space $\mathcal{V}^{(0)}$ (3.32).

Recall that any discrete image $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ encode the linear projection of an input function $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ on $\mathcal{V}$, as described in (3.43). Then, the feature maps $\mathrm{D}_l^{(j)}$ of wavelet packet coefficients, encoding the linear projection of X on $\mathcal{J}_l^{(j)}$ (discrete framework), or F on $\mathcal{V}_l^{(j)}$ (continuous framework), are computed using the fast *wavelet packet transform* (WPT) algorithm (Coifman and Wickerhauser, 1992). As for the FWT, initialization is done with $\mathrm{D}_0^{(0)} := \mathrm{X}$. Then, similarly to (3.24), we get, for any $l \in \{0\,..\,4^J - 1\}$ and any $k \in \{0\,..\,3\}$,

$$\mathrm{D}_{4l+k}^{(j+1)} = (\mathrm{D}_l^{(j)} * \overline{\mathrm{G}_k}) \downarrow 2. \tag{3.63}$$

Examples of Wavelet Packet Bases. Depending on whether we choose to split a given detail subspace as in (3.61), there are many possible wavelet packet bases. Several approaches have been proposed to select the "best" basis for a given class of input signals (Laine and Fan, 1993; Learned and Willsky, 1995; Sengur et al., 2007). However, this topic is beyond the scope of this thesis. Instead, two typical examples of wavelet packet bases are given below.

A special case is the standard wavelet basis $\mathbf{E}^{(J)}$ (discrete framework) or $\boldsymbol{\Psi}^{(J)}$ (continuous framework), as introduced in (3.25) and (3.41). In this example, only the subspaces of coarser resolution $\mathcal{J}_0^{(j)}$ (discrete framework) or $\mathcal{V}_0^{(j)}$ (continuous framework) are decomposed, for $j \in \{0\,..\,J-1\}$. Another example is the *pseudo-local cosine* basis, corresponding to a fully-decomposed binary tree. In this context, (3.25) and (3.41) become

$$\mathbf{E}^{(J)} := \biguplus_{l=0}^{4^J-1} (\mathrm{E}_{l,\boldsymbol{n}}^{(J)})_{\boldsymbol{n}\in\mathbb{Z}^2} \qquad \text{and} \qquad \boldsymbol{\Psi}^{(J)} := \biguplus_{l=0}^{4^J-1} (\Psi_{l,\boldsymbol{n}}^{(J)})_{\boldsymbol{n}\in\mathbb{Z}^2}. \tag{3.64}$$

Unlike standard wavelet bases, the Fourier windows characterizing each subspace $\mathcal{J}_l^{(J)}$ (discrete framework) or $\mathcal{V}_l^{(J)}$ (continuous framework) are of identical size $\pi/2^J$. Therefore, the basis functions (3.64) are equally well localized, both in space and frequency. The WPT decomposition in a pseudo-local cosine basis is performed in $O(N_\mathrm{f} \times N^2 \times J)$ operations, where, as a reminder, N_f and N respectively denote the support size of (h, g) (QMFs) and X (input images).

3.2.6 Discrete Wavelet Transforms are Unstable to Translations

So far in this chapter, we explored the topic of designing sparse image representations. In particular, wavelet bases and their like—among which wavelet packets—are good candidates for this purpose, due to a tradeoff between spatial and frequency resolution. However, we set aside an important property for pattern recognition and image classification: translation (or shift) invariance. As explained in Section 2.4, a good feature extractor must retain discriminant image component while decreasing intra-class variability. Using the notations introduced in Section 3.1.1, translation invariance (up to a phase shift) can be formalized as follows:

$$\forall n \in \mathcal{B},\ \forall \boldsymbol{u} \in \mathbb{R}^2,\ |\Gamma_n(\mathcal{T}_{\boldsymbol{u}}\mathrm{X})| = |\Gamma_n(\mathrm{X})|\,, \tag{3.65}$$

where $\mathcal{T}_{\boldsymbol{u}} : l^2_{\mathbb{R}}(\mathbb{Z}^2) \to l^2_{\mathbb{R}}(\mathbb{Z}^2)$ translates input images by a vector $\boldsymbol{u} \in \mathbb{R}^2$. A formal definition involving antialiased interpolation will be provided in Chapter 4 (4.41). On the

discrete grid, the translation operator is simply defined as follows:

$$\forall \boldsymbol{p} \in \mathbb{Z}^2,\ \mathcal{T}_{\boldsymbol{p}} \mathrm{X}[\boldsymbol{n}] := \mathrm{X}[\boldsymbol{n} - \boldsymbol{p}]. \tag{3.66}$$

Recall that, for wavelet bases, the feature extractor $\boldsymbol{\Gamma}$ has been defined in (3.26). Then, (3.65) implies, for any $j \in \mathbb{N} \setminus \{0\}$, $k \in \{0\,..\,3\}$, and $\boldsymbol{n} \in \mathbb{Z}^2$,

$$\forall \boldsymbol{u} \in \mathbb{R}^2,\ |\Gamma^{(j)}_{k,\boldsymbol{n}}(\mathcal{T}_{\boldsymbol{u}} \mathrm{X})| = |\Gamma^{(j)}_{k,\boldsymbol{n}}(\mathrm{X})|. \tag{3.67}$$

One of the main drawbacks of the discrete wavelet transform is that (3.67) is not satisfied, due to the subsampling operation in (3.24). To understand this phenomenon, we consider, as in Section 3.2.3, the one-dimensional setting. Let $\mathrm{x} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ denote an input sequence corresponding to a continuous signal $f \in L^2_{\mathbb{R}}(\mathbb{R})$. Similarly to (3.45), we have, for any $n \in \mathbb{Z}$, $\mathrm{x}[n] := (f * \overline{\phi})(n)$, where $\phi \in L^2_{\mathbb{R}}(\mathbb{R})$ denotes the 1D scaling function corresponding to the QMFs (h, g). We respectively denote by $\mathrm{c}^{(j)}$ and $\mathrm{d}^{(j)}$ the sequence of scaling and wavelet coefficients at scale $j \in \{1\,..\,J\}$, computed, similarly to (3.24), by

$$\mathrm{c}^{(j+1)} = (\mathrm{c}^{(j)} * \overline{\mathrm{h}}) \downarrow 2 \qquad \text{and} \qquad \mathrm{d}^{(j+1)} = (\mathrm{c}^{(j)} * \overline{\mathrm{g}}) \downarrow 2, \tag{3.68}$$

initialized at $\mathrm{c}^{(0)} := \mathrm{x}$. Now, we translate x by one unit, and denote by $\mathrm{c}'^{(j)}$ and $\mathrm{d}'^{(j)}$ the corresponding scaling and wavelet coefficients at scale j. Then, the sequences of wavelet coefficients $\mathrm{d}^{(j)}$ and $\mathrm{d}'^{(j)}$ satisfy, for any $n \in \mathbb{Z}$,

$$\mathrm{d}^{(j)}[n] = (f * \overline{\psi}^{(j)}_0)(2^j n) \qquad \text{and} \qquad \mathrm{d}'^{(j)}[n] = (f * \overline{\psi}^{(j)}_0)(2^j n - 1), \tag{3.69}$$

where $\psi^{(j)}_0$ is a dilated version of the mother wavelet ψ by a factor 2^j. We notice that $\mathrm{d}^{(j)}$ and $\mathrm{d}'^{(j)}$ are obtained by uniformly sampling $f * \overline{\psi}^{(j)}_0$ on two non-overlapping grids. Since $\psi^{(j)}_0$ is high-frequency, the two sequences of wavelet coefficients may vary significantly. Actually, their energy $\|\mathrm{d}^{(j)}\|_2^2$ and $\|\mathrm{d}'^{(j)}\|_2^2$ may be completely different. To get a better intuition about this, consider an example where $f \leftarrow \delta$ is chosen to be the unit impulse signal,[1] Then, (3.69) becomes

$$\mathrm{d}^{(j)}[n] = \overline{\psi}^{(j)}_0(2^j n) = 2^{-\frac{j}{2}}\, \overline{\psi}(n); \tag{3.70}$$

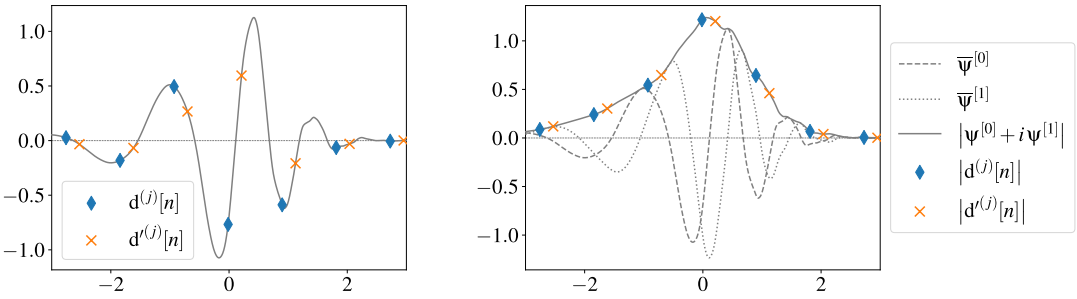
$$\mathrm{d}'^{(j)}[n] = \overline{\psi}^{(j)}_0(2^j n - 1) = 2^{-\frac{j}{2}}\, \overline{\psi}(n - 2^{-j}), \tag{3.71}$$

where ψ denotes the mother wavelet. Figure 3.3a illustrates this situation with $j = 2$. We can relate this phenomenon to Shannon's sampling theorem (Shannon, 1949). It implies that reconstructing a high-frequency input from a partial subsampled signal leads to aliasing artifacts if not handled carefully. More information about shift invariance can be found in a review by Kingsbury and Magarey (1998).

On the other hand, if x is shifted by $2^j p$ units ($p \in \mathbb{Z}$), then a similar reasoning shows that $\mathrm{d}^{(j)}$ is shifted by p units. Therefore, a partial form of shift equivariance exists, when inputs are shifted by a multiple of the subsampling factor.

In the next section, we explore an approach designed to solve the problem of shift instability, by considering complex-valued *redundant frames*, instead of real-valued orthonormal bases. As we shall see, the proposed solution naturally solves another weakness of wavelet bases: the lack of directional selectivity.

[1] Technically, $\delta \notin L^2_{\mathbb{R}}(\mathbb{R})$. However, the observed phenomenon holds for L^2 input signals.



(a) Real-valued framework

(b) Complex-valued framework

Figure 3.3. Illustration of aliasing effects: outputs of 1D FWT (a) and DT-ℂWT (b), for a discrete impulse signal and a shifted version by one pixel. In the real-valued case (a), the analytic expressions of $\mathrm{d}^{(j)}[n]$ and $\mathrm{d}'^{(j)}[n]$ are provided in (3.70) and (3.71), respectively. The plots have been drawn for $j = 2$, using a Daubechies wavelet with $q = 6$ vanishing moments.

3.3 Complex Redundant Discrete Wavelet Transforms

We saw in Section 3.2.6 that image representations in a wavelet basis are unstable to small shifts. Moreover, since the basis functions are tensor products of one-dimensional scaling function and wavelet, FWT performs well at detecting horizontal and vertical features, but fails at discriminating intermediate orientations. The third mother wavelet $\Psi_3 := \psi \otimes \psi$ presents a checkerboard pattern (see Figure 3.2), and as such is not properly oriented. To overcome the lack of directional selectivity and shift invariance, Kingsbury (1999) introduced a complex-valued, discrete wavelet decomposition in a redundant tight frame generated by 6 oriented mother wavelets. The decomposition is performed with a fast algorithm based on a dual-tree filter bank decomposition, called the *dual-tree complex wavelet transform* (DT-ℂWT). As discussed in Section 3.1.3, redundancy in frames can be exploited to improve the sparsity of image representations. The combined effect of overcompleteness with shift invariance is known to produce more suitable image representations, as stated by Pustelnik et al. (2016) in the context of inverse problems.

The main ideas for DT-ℂWT have been reviewed by I. W. Selesnick et al. (2005), and are summarized below. Section 3.3.1 provides some intuition behind the concept of complex analytic wavelets. Then, Section 3.3.2 introduces the complex wavelet tight frame on which input images are decomposed, and Section 3.3.3 describes the DT-ℂWT algorithm, taking advantage of standard, separable FWT decompositions. Finally, Section 3.3.4 extends the concept of wavelet packets to the dual-tree framework.

3.3.1 General Intuition

The Fourier basis defined in (3.12) satisfies both directional selectivity and shift invariance (3.65), with $\mathcal{B} \leftarrow \mathcal{B}_N$, where $N \in \mathbb{N} \setminus \{0\}$ denotes the support size of input images. According to I. W. Selesnick et al. (2005), these properties are linked to the basis functions being *analytic*. It means that their Fourier transform is supported in one half of the frequency axis. In the 2D setting, the Fourier basis functions have their Fourier support localized in one quadrant of the frequency plane. More precisely,

$$\forall \boldsymbol{n} \in \mathcal{B}_N,\ \widehat{\mathrm{E}_{\boldsymbol{n}}} = \delta_{\boldsymbol{\theta}_{\boldsymbol{n}}}, \tag{3.72}$$

where $\mathrm{E}_{\boldsymbol{n}} \in \mathcal{I}_N$ satisfies (3.12), and $\delta_{\boldsymbol{\theta_n}}$ denotes the 2D unit impulse function at frequency $\boldsymbol{\theta_n} := 2\pi(\boldsymbol{n}/N + \boldsymbol{p})$, where $\boldsymbol{p} \in \mathbb{Z}^2$ is chosen such that $\boldsymbol{\theta_n} \in [-\pi, \pi]^2$. In the above expression, we have considered the 2D *discrete-time Fourier transform*, defined, analogously to (3.19), by

$$\widehat{\mathrm{X}}(\boldsymbol{\omega}) := \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathrm{X}[\boldsymbol{n}] \mathrm{e}^{-i\langle \boldsymbol{\omega}, \boldsymbol{n} \rangle} \qquad \forall \boldsymbol{\omega} \in [-\pi, \pi]^2 . \tag{3.73}$$

Can this idea be transposed to wavelet bases? According to Figure 3.2, the basis functions $\mathrm{E}^{(j)}_{k,\boldsymbol{n}}$ (discrete framework) or $\Psi^{(j)}_{k,\boldsymbol{n}}$ (continuous framework) have their energy concentrated in four symmetric square regions of size $\pi/2^j$. Now, the idea is to break symmetry by "shutting down" three of these four square regions. This can be done by considering a couple of one-dimensional complex scaling function and wavelet $(\phi, \psi) \in L^2_{\mathbb{C}}(\mathbb{R}) \times L^2_{\mathbb{C}}(\mathbb{R})$ satisfying

$$\operatorname{supp} \widehat{\phi} \subset \mathbb{R}_+ \qquad \text{and} \qquad \operatorname{supp} \widehat{\psi} \subset \mathbb{R}_+. \tag{3.74}$$

Then, building tensor products of ϕ and ψ, as done in (3.42), yields 2D oriented wavelets Ψ_{1-3} whose energy is mainly located on the upper-right quadrant of the Fourier domain, as evidenced in Figure 3.4. We will then qualify Ψ_{1-3} as analytic.[2]

In Section 3.3.2, we will see that, under certain conditions, (ϕ, ψ) generates a tight redundant frame of $L^2_{\mathbb{C}}(\mathbb{R}^2)$. Then, Section 3.3.3 explains how to build a *dual-tree filter bank* and how to decompose an input function $F \in L^2_{\mathbb{C}}(\mathbb{R}^2)$ in the wavelet tight frame.

3.3.2 Complex Wavelet Tight Frames

Let $(\mathrm{h}^{[0]}, \mathrm{g}^{[0]})$ and $(\mathrm{h}^{[1]}, \mathrm{g}^{[1]})$ denote two pairs of QMFs. We respectively denote by $(\phi^{[0]}, \psi^{[0]})$ and $(\phi^{[1]}, \psi^{[1]})$ the pairs of 1D scaling function and wavelet characterized by $(\mathrm{h}^{[0]}, \mathrm{g}^{[0]})$ and $(\mathrm{h}^{[1]}, \mathrm{g}^{[1]})$, as explained in Remark 3.1. Moreover, we assume that $\psi^{[0]}$ and $\psi^{[1]}$ form a Hilbert transform pair:

$$\psi^{[1]} = \mathcal{H}(\psi^{[0]}). \tag{3.75}$$

where $\mathcal{H} : L^2_{\mathbb{R}}(\mathbb{R}) \to L^2_{\mathbb{R}}(\mathbb{R})$ denotes the *Hilbert transform*, defined, for any $\xi \in \mathbb{R}$, by

$$\widehat{\psi^{[1]}}(\xi) := -i \operatorname{sgn}(\xi)\, \widehat{\psi^{[0]}}(\xi). \tag{3.76}$$

We consider the following complex scaling function and wavelet:

$$\phi := \frac{1}{\sqrt{2}}(\phi^{[0]} + i\, \phi^{[1]}) \qquad \text{and} \qquad \psi := \frac{1}{\sqrt{2}}(\psi^{[0]} + i\, \psi^{[1]}). \tag{3.77}$$

Then, ϕ and ψ are analytic (3.74). We now build complex 2D wavelets, in a similar fashion as (3.42):

$$\Psi_1^{\nearrow} := \phi \otimes \psi; \qquad \Psi_2^{\nearrow} := \psi \otimes \phi; \qquad \Psi_3^{\nearrow} := \psi \otimes \psi; \tag{3.78}$$

$$\Psi_1^{\searrow} := \phi \otimes \psi^*; \qquad \Psi_2^{\searrow} := \psi \otimes \phi^*; \qquad \Psi_3^{\searrow} := \psi \otimes \psi^*, \tag{3.79}$$

[2] There are several possible extensions to the concept of analytic signals in higher dimensions. In this thesis, "analytic" means that the Fourier transform is located in the top-right quadrant (and, by abuse of language, in any of the four quadrants). It is worth noticing that Havlicek et al. (1997) uses a less restrictive definition, involving the half-plane of positive x-values.

where * denotes the complex conjugate. Each wavelet corresponds to a specific orientation, as shown in Figure 3.4. We also consider their complex conjugates, respectively denoted by $\Psi_k^{\swarrow}$ on the one hand, and $\Psi_k^{\nwarrow}$ on the other hand. In these notations, the arrow points to the Fourier quadrant in which energy is concentrated. Finally, we denote by $\Psi_{k,\boldsymbol{n}}^{\nearrow(j)}$ the translated and dilated version of $\Psi_k^{\nearrow}$, according to (3.37). We do the same for the three other Fourier quadrants.

Similarly to (3.29) and (3.41), let $\mathcal{V} \subset L^2_{\mathbb{C}}(\mathbb{R}^2)$ denote an initial approximation space for continuous input images:

$$\mathcal{V} := \operatorname{span}(\Phi_{\boldsymbol{n}})_{\boldsymbol{n}\in\mathbb{Z}^2} \subset L^2_{\mathbb{C}}(\mathbb{R}^2), \tag{3.80}$$

and $\boldsymbol{\Psi}^{(J)}$ denote the corresponding wavelet basis (3.41), where the mother real-valued scaling function Φ and wavelets Ψ_{1-3} satisfy (3.42) with $(\phi,\, \psi) \leftarrow (\phi^{[0]},\, \psi^{[0]})$ as introduced above. We then have the following fundamental property. For any $k \in \{1\,..\,3\}$ and $j \in \{1\,..\,J\}$, the detail subspace $\mathcal{V}_k^{(j)}$ (3.39) is also spanned a the four-time redundant complex family:

$$\mathcal{V}_k^{(j)} = \operatorname{span}(\Psi_{k,\boldsymbol{n}}^{\nearrow(j)},\, \Psi_{k,\boldsymbol{n}}^{\searrow(j)},\, \Psi_{k,\boldsymbol{n}}^{\swarrow(j)},\, \Psi_{k,\boldsymbol{n}}^{\nwarrow(j)})_{\boldsymbol{n}\in\mathbb{Z}^2}, \tag{3.81}$$

which has the *tight frame* property. Thus, for any $F \in L^2_{\mathbb{C}}(\mathbb{R}^2)$, the orthogonal projection of F on $\mathcal{V}_k^{(j)}$, denoted by $\Pi_k^{(j)} F$, can be recovered from the feature maps of complex wavelet coefficients:

$$\Pi_k^{(j)} F = \frac{1}{4} \sum_{\boldsymbol{n}\in\mathbb{Z}^2} \Big(\mathrm{D}_k^{\nearrow(j)}[\boldsymbol{n}]\, \Psi_{k,\boldsymbol{n}}^{\nearrow(j)} + \mathrm{D}_k^{\searrow(j)}[\boldsymbol{n}]\, \Psi_{k,\boldsymbol{n}}^{\searrow(j)} + \mathrm{D}_k^{\swarrow(j)}[\boldsymbol{n}]\, \Psi_{k,\boldsymbol{n}}^{\swarrow(j)} + \mathrm{D}_k^{\nwarrow(j)}[\boldsymbol{n}]\, \Psi_{k,\boldsymbol{n}}^{\nwarrow(j)} \Big), \tag{3.82}$$

where $\mathrm{D}_k^{\nearrow(j)},\, \mathrm{D}_k^{\searrow(j)},\, \mathrm{D}_k^{\swarrow(j)},\, \mathrm{D}_k^{\nwarrow(j)} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ satisfy, similarly to (3.44) and (3.45),

$$\forall \boldsymbol{n} \in \mathbb{Z}^2,\ \mathrm{D}_k^{\nearrow(j)}[\boldsymbol{n}] = \langle F,\, \Psi_{k,\boldsymbol{n}}^{\nearrow(j)} \rangle = \Big(F * \overline{\Psi}_{k,\boldsymbol{0}}^{\nearrow(j)^*} \Big)(2^j \boldsymbol{n}). \tag{3.83}$$

Then, the family

$$\boldsymbol{\Psi}_{\mathbb{C}}^{(J)} := \biguplus_{j=1}^{J} \biguplus_{k=1}^{3} (\Psi_{k,\boldsymbol{n}}^{\nearrow(j)},\, \Psi_{k,\boldsymbol{n}}^{\searrow(j)},\, \Psi_{k,\boldsymbol{n}}^{\swarrow(j)},\, \Psi_{k,\boldsymbol{n}}^{\nwarrow(j)})_{\boldsymbol{n}\in\mathbb{Z}^2} \uplus (4\,\Phi_{\boldsymbol{n}}^{(J)})_{\boldsymbol{n}\in\mathbb{Z}^2} \tag{3.84}$$

is a *tight frame* of $\mathcal{V}$. In this expression, $\Phi_{\boldsymbol{n}}^{(J)} \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ denotes the translated and dilated version of the (real-valued) mother scaling function Φ at the coarsest scale J. Unlike the orthonormal basis $\boldsymbol{\Psi}^{(J)}$ defined in (3.41), the elements of $\boldsymbol{\Psi}_{\mathbb{C}}^{(J)}$ are no longer linearly independent, but any continuous input image $F \in \mathcal{V}$ can be perfectly reconstructed from this multiresolution representation. Note that, when F is real-valued, $\mathrm{D}_k^{\swarrow(j)}$ and $\mathrm{D}_k^{\nwarrow(j)}$ are the respective complex conjugates of $\mathrm{D}_k^{\nearrow(j)}$ and $\mathrm{D}_k^{\searrow(j)}$. Therefore, in practice we only need to compute and store half of the wavelet coefficients to reconstruct the signal.

In the standard wavelet basis (3.41), each atom $\Psi_{k,\boldsymbol{n}}^{(j)}$ has its energy concentrated in four symmetric square windows in the Fourier domain, as shown in Figure 3.2. In the complex frame (3.84) however, each of the four windows is assigned to a specific atom $\Psi_{k,\boldsymbol{n}}^{\nearrow(j)}$, $\Psi_{k,\boldsymbol{n}}^{\searrow(j)}$, $\Psi_{k,\boldsymbol{n}}^{\swarrow(j)}$ or $\Psi_{k,\boldsymbol{n}}^{\nwarrow(j)}$. The wavelets are therefore analytic and oriented. Figure 3.4 illustrates this situation.

3.3.3 Dual-Tree Filter Bank Decomposition

Let $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ denote a continuous input image. To compute the feature maps of complex wavelet coefficients $\mathrm{D}_k^{\nearrow(j)}$ and $\mathrm{D}_k^{\searrow(j)} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ (3.83), Kingsbury (1999) developed the DT-$\mathbb{C}$WT algorithm, which is described in this section. Extending this result to $F \in L^2_{\mathbb{C}}(\mathbb{R}^2)$ is possible; however, for the purpose of conciseness, the focus of this section is on real-valued inputs.

Four Parallel FWTs. DT-$\mathbb{C}$WT is done by performing four parallel FWTs, using four distinct sets of 2D real-valued scaling function and wavelets, which are tensor products of $(\phi^{[0]}, \psi^{[0]})$ and $(\phi^{[1]}, \psi^{[1]})$. More precisely, for any $m := 2i + i' \in \{0\,..\,3\}$, we consider:

$$\Phi^{[m]} := \phi^{[i]} \otimes \phi^{[i']}; \quad \Psi_1^{[m]} := \phi^{[i]} \otimes \psi^{[i']}; \quad \Psi_2^{[m]} := \psi^{[i]} \otimes \phi^{[i']}; \quad \Psi_3^{[m]} := \phi^{[i]} \otimes \phi^{[i']}. \tag{3.85}$$

Then, the wavelets $\Psi_{k,\boldsymbol{n}}^{\nearrow(j)}$ and $\Psi_{k,\boldsymbol{n}}^{\searrow(j)}$ from the tight frame $\boldsymbol{\Psi}_{\mathbb{C}}^{(J)}$ introduced in (3.84) satisfy the following expression:

$$\begin{pmatrix} \Psi_{k,\boldsymbol{n}}^{\nearrow(j)} \\ \Psi_{k,\boldsymbol{n}}^{\searrow(j)} \end{pmatrix} = \frac{1}{2}\left[\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \Psi_{k,\boldsymbol{n}}^{[0](j)} \\ \Psi_{k,\boldsymbol{n}}^{[3](j)} \end{pmatrix} + i \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} \Psi_{k,\boldsymbol{n}}^{[2](j)} \\ \Psi_{k,\boldsymbol{n}}^{[1](j)} \end{pmatrix} \right], \tag{3.86}$$

where $\Psi_{k,\boldsymbol{n}}^{[m](j)} \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ satisfies (3.37) with $\Psi_k \leftarrow \Psi_k^{[m]}$.

In practice, each FWT is computed with a 2D FB decomposition as described in Section 3.2.1. More specifically, for any $m := 2i + i' \in \{0\,..\,3\}$, we use the 2D filter bank $\mathbf{G}^{[m]} := (\mathrm{G}_k^{[m]})_{k \in \{0..3\}}$ defined, similarly to (3.20), by

$$\mathrm{G}_0^{[m]} := \mathrm{h}^{[i]} \otimes \mathrm{h}^{[i']}; \quad \mathrm{G}_1^{[m]} := \mathrm{h}^{[i]} \otimes \mathrm{g}^{[i']}; \quad \mathrm{G}_2^{[m]} := \mathrm{g}^{[i]} \otimes \mathrm{h}^{[i']}; \quad \mathrm{G}_3^{[m]} := \mathrm{g}^{[i]} \otimes \mathrm{g}^{[i']}, \tag{3.87}$$

where the QMFs $(\mathrm{h}^{[0]}, \mathrm{g}^{[0]})$ and $(\mathrm{h}^{[1]}, \mathrm{g}^{[1]})$ have been introduced in Section 3.3.2. For each $m \in \{0\,..\,3\}$, the m-th FWT is initialized with

$$\mathrm{D}_0^{[m](0)} \leftarrow \mathrm{X}^{[m]}, \tag{3.88}$$

where $\mathrm{X}^{[m]} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ encodes the linear projection of F on

$$\mathcal{V}^{[m]} := \mathrm{span}\big(\Phi_{\boldsymbol{n}}^{[m]}\big)_{\boldsymbol{n} \in \mathbb{Z}^2}, \tag{3.89}$$

as in (3.43) and (3.45). Then, the discrete wavelet coefficients are computed using the FB decomposition described in Section 3.2.1 with $\mathrm{G} \leftarrow \mathrm{G}^{[m]}$. The corresponding feature maps are denoted by $\mathrm{D}_k^{[m](j)} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ and satisfy (3.44) with $\Psi_{k,\boldsymbol{n}}^{(j)} \leftarrow \Psi_{k,\boldsymbol{n}}^{[m](j)}$.

Finally, we use the relation (3.77) between real and complex scaling functions / wavelets, as well as the tensor product constructions (3.78), (3.79) and (3.85), which yields the following expression of the complex feature maps $\mathrm{D}_k^{\nearrow(j)}$ and $\mathrm{D}_k^{\searrow(j)}$ defined in (3.83):

$$\begin{pmatrix} \mathrm{D}_k^{\nearrow(j)} \\ \mathrm{D}_k^{\searrow(j)} \end{pmatrix} = \frac{1}{2}\left[\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \mathrm{D}_k^{[0](j)} \\ \mathrm{D}_k^{[3](j)} \end{pmatrix} - i \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} \mathrm{D}_k^{[2](j)} \\ \mathrm{D}_k^{[1](j)} \end{pmatrix} \right], \tag{3.90}$$

Expression (3.90) provides a fast computation of the complex wavelet coefficients, using separable convolutions and feature map combinations.

Shift Invariance. Due to the near-analytic property of DT-$\mathbb{C}$WT, the complex wavelet feature maps defined in (3.83), and computed following (3.90), are nearly shift invariant, in contrast to standard FWT. Back to the one-dimensional situation, we can show that the modulus of the complex wavelet ψ (3.77) provides a smooth envelope for the real wavelet $\psi^{[0]}$. Consider again the example from Section 3.2.6, taking as inputs two shifted unit impulse signals. The corresponding sequences of wavelet coefficients $\mathrm{d}^{(j)}$ and $\mathrm{d}'^{(j)}$ respectively satisfy (3.70) and (3.71), with ψ being the complex analytic mother wavelet. Then, if j is large enough, we get $\left|\mathrm{d}'^{(j)}[n]\right| \approx \left|\mathrm{d}^{(j)}[n]\right|$ for any $n \in \mathbb{Z}$. Figure 3.3 illustrates both FWT and DT-$\mathbb{C}$WT situations.

Half-Sample Delay Condition. A necessary and sufficient condition for ψ to be analytic (3.76) is that $(\mathrm{h}^{[0]}, \mathrm{h}^{[1]})$ (low-pass filters) must satisfy the *half-sample delay* condition (I. Selesnick, 2001; Yu and Ozkaramanli, 2005):

$$\forall \omega \in [-\pi, \pi], \, \widehat{\mathrm{h}}^{[1]}(\omega) = \mathrm{e}^{-i\frac{\omega}{2}} \, \widehat{\mathrm{h}}^{[0]}(\omega). \tag{3.91}$$

However, if one want to use finitely-supported filters, like those associated to the Daubechies wavelets, (3.91) cannot be exactly satisfied. In this case, approximations are therefore needed. Moreover, we want both filters to have nearly equal support size and number of vanishing moments. In addition, (3.16)–(3.18) must be satisfied for both sets of filters. Several approaches have been proposed for this purpose. In particular, the *quarter-shift* (Q-shift) solution (Kingsbury, 2000, 2003) yields two low-pass filters which are flipped with respect to each other: $\mathrm{h}^{[1]} = \overline{\mathrm{h}^{[0]}}$. Designing such filters must be done carefully to approximately meet the half-sample delay condition. This solution is well-suited for orthogonal filters with arbitrarily large support.

Remark 3.6 (first decomposition stage)**.** In practice, F is unknown; thus we cannot compute $\mathrm{X}^{[0-3]}$ as introduced above. Instead, we only have one discrete input image $\mathrm{X} := \mathrm{X}^{[0]}$, encoding the linear projection of F onto $\mathcal{V}$ in the orthonormal basis $(\boldsymbol{\Phi}_{\boldsymbol{n}})_{\boldsymbol{n}\in\mathbb{Z}^2}$. Therefore, for each $m \in \{0\,..\,3\}$, the m-th FWT is initialized with

$$\mathrm{D}_0^{[m](0)} \leftarrow \mathrm{X}, \tag{3.92}$$

which is different from the "perfect" initialization (3.88). Unfortunately in this context, the complex feature maps obtained with (3.90) only become near-analytic when j goes to ∞. In order to get near-analytic image representations from the first iterations, I. W. Selesnick et al. (2005) came along with the following solution. The first stage of FWT is computed with a specific set of filters, denoted by $(\widetilde{\mathrm{h}}^{[0]}, \widetilde{\mathrm{g}}^{[0]})$ and $(\widetilde{\mathrm{h}}^{[1]}, \widetilde{\mathrm{g}}^{[1]})$, satisfying the *one-sample delay* condition:

$$\forall n \in \mathbb{Z}, \, \widetilde{\mathrm{h}}^{[1]}[n] := \widetilde{\mathrm{h}}^{[0]}[n-1]. \tag{3.93}$$

Unlike the half-sample delay condition (3.91), implementing $(\widetilde{\mathrm{h}}^{[1]}, \widetilde{\mathrm{g}}^{[1]})$ from $(\widetilde{\mathrm{h}}^{[0]}, \widetilde{\mathrm{g}}^{[0]})$ is straightforward.

Discrete Frames. We consider the following family of complex, near-analytic basis functions:

$$\mathbf{E}_{\mathbb{C}}^{(J)} := \biguplus_{j=1}^{J} \biguplus_{k=1}^{3} \left(\mathrm{E}_{k,\boldsymbol{n}}^{\nearrow(j)}, \, \mathrm{E}_{k,\boldsymbol{n}}^{\searrow(j)}, \, \mathrm{E}_{k,\boldsymbol{n}}^{\swarrow(j)}, \, \mathrm{E}_{k,\boldsymbol{n}}^{\nwarrow(j)}\right)_{\boldsymbol{n}\in\mathbb{Z}^2} \uplus \left(4\,\mathrm{E}_{0,\boldsymbol{n}}^{(J)}\right)_{\boldsymbol{n}\in\mathbb{Z}^2}, \tag{3.94}$$

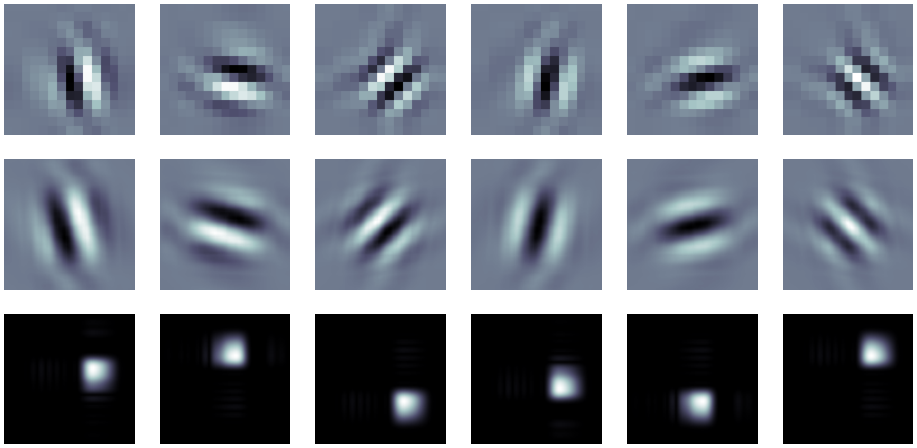


Figure 3.4. Top: real part of the discrete basis functions $\mathrm{E}^{\nearrow(j)}_{k,\mathbf{0}}$, $\mathrm{E}^{\searrow(j)}_{k,\mathbf{0}}$ (3.95) with depth $j = 2$ and $k = 1\,..\,3$, computed with Q-shift orthogonal QMFs of length 10. Middle: real part of the continuous basis functions (wavelets) $\Psi^{\nearrow(j)}_{k,\mathbf{0}}$, $\Psi^{\searrow(j)}_{k,\mathbf{0}}$ (3.86). Bottom: modulus of the discrete-time Fourier transform of $\mathrm{E}^{\nearrow(j)}_{k,\mathbf{0}}$ and $\mathrm{E}^{\searrow(j)}_{k,\mathbf{0}}$ (the origin is located at the center).

where we have denoted, analogously to (3.86),

$$\begin{pmatrix} \mathrm{E}^{\nearrow(j)}_{k,\boldsymbol{n}} \\ \mathrm{E}^{\searrow(j)}_{k,\boldsymbol{n}} \end{pmatrix} = \frac{1}{2}\left[\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \mathrm{E}^{[0](j)}_{k,\boldsymbol{n}} \\ \mathrm{E}^{[3](j)}_{k,\boldsymbol{n}} \end{pmatrix} + i \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} \mathrm{E}^{[2](j)}_{k,\boldsymbol{n}} \\ \mathrm{E}^{[1](j)}_{k,\boldsymbol{n}} \end{pmatrix} \right], \tag{3.95}$$

where, for each $m \in \{0\,..\,3\}$, $\mathrm{E}^{[m](j)}_{k,\boldsymbol{n}} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ denote the atoms of the m-th wavelet basis, satisfying (3.23) and (3.25) with $\mathbf{G} \leftarrow \mathbf{G}^{[m]}$ such as introduced in (3.87). Besides, in (3.94), we have denoted $\mathrm{E}^{(J)}_{0,\boldsymbol{n}} := \mathrm{E}^{[0](J)}_{0,\boldsymbol{n}}$ for any $\boldsymbol{n} \in \mathbb{Z}^2$ (coarsest scale). We remind that real-valued wavelet bases have been introduced in Section 3.2.1. Expression (3.90) implies that

$$\langle \mathrm{X},\, \mathrm{E}^{\nearrow(j)}_{k,\boldsymbol{n}} \rangle := \mathrm{D}^{\nearrow(j)}_{k}[\boldsymbol{n}] \qquad \forall j,\, k,\, \boldsymbol{n}, \tag{3.96}$$

as well as the three other quadrants. Then, we can get a close reconstruction of any input image $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ in the family $\mathbf{E}^{(J)}_{\mathbb{C}}$, using the above complex wavelet coefficients. Besides, an M-term approximation of X in $\mathbf{E}^{(J)}_{\mathbb{C}}$ can be obtained by computing the M largest coefficients as in (3.26), with a similar decay rate in $O(1/M)$. As for Figure 3.2 in the case of orthogonal wavelet basis, Figure 3.4 illustrates the redundant discrete and continuous basis functions in the DT-$\mathbb{C}$WT framework.

3.3.4 The Dual-Tree Complex Wavelet Packet Transform

The *dual-tree complex wavelet packet transform* (DT-$\mathbb{C}$WPT), introduced by Bayram and I. W. Selesnick (2008), is an extension of DT-$\mathbb{C}$WT to the wavelet packet framework. Similarly to Section 3.2.5, the complex detail subspaces

$$\mathcal{J}^{\nearrow(j)}_{k} := \mathrm{span}\big(\mathrm{E}^{\nearrow(j)}_{k,\boldsymbol{n}}\big)_{\boldsymbol{n}\in\mathbb{Z}^2} \qquad \text{and} \qquad \mathcal{W}^{\nearrow(j)}_{k} := \mathrm{span}\big(\Psi^{\nearrow(j)}_{k,\boldsymbol{n}}\big)_{\boldsymbol{n}\in\mathbb{Z}^2} \tag{3.97}$$

can be split into four subspaces with decreased spatial localization and increased frequency localization. The exact same reasoning applies to the three other quadrants of the Fourier

domain. To achieve this goal, it would be tempting, for each $m \in \{0\,..\,3\}$, to decompose the dual-tree feature map $\mathrm{D}_l^{[m](j)}$ using the FB $\mathbf{G}^{[m]}$ introduced in (3.87), as done in the 1D case by Z.-M. Xie et al. (2004). Like (3.63), we would get

$$\mathrm{D}_{4l+k}^{[m](j+1)} = \left(\mathrm{D}_l^{[m](j)} * \overline{\mathrm{G}}_k^{[m]}\right) \downarrow 2. \tag{3.98}$$

However, this leads to non-analytic solutions, as explained by Bayram and I. W. Selesnick (2008). To keep the near-analytic properties of standard DT-$\mathbb{C}$WT, these authors proposed to apply a unique filter bank $\mathbf{G} \in \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^4$ to each high-frequency feature map $\mathrm{D}_l^{[m](j)}$, for any $j \geq 2$ and any $l \geq 1$. Then, (3.98) becomes

$$\mathrm{D}_{4l+k}^{[m](j+1)} = \left(\mathrm{D}_l^{[m](j)} * \overline{\mathrm{G}_k}\right) \downarrow 2. \tag{3.99}$$

Besides, after the first decomposition stage using the high-pass filters $\widetilde{\mathrm{g}}^{[0]}$ and $\widetilde{\mathrm{g}}^{[1]}$, the roles must be switched between $(\mathrm{h}^{[0]}, \mathrm{g}^{[0]})$ and $(\mathrm{h}^{[1]}, \mathrm{g}^{[1]})$. Finally, for any decomposition depth $j \in \{1\,..\,J\}$ and output channel $l \in \{1\,..\,4^j - 1\}$, the feature maps of complex wavelet packet coefficients at scale 2^j, denoted by $\mathrm{D}_l^{\nearrow(j)}, \mathrm{D}_l^{\searrow(j)} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$, are computed with linear combinations between the real-valued feature maps $\mathrm{D}_l^{[m](j)}$ $(m \in \{0\,..\,3\})$, similarly to (3.90). In the pseudo-local cosine framework (fully-decomposed binary tree), we therefore get

$$\begin{pmatrix} \mathrm{D}_l^{\nearrow(J)} \\ \mathrm{D}_l^{\searrow(J)} \end{pmatrix} = \frac{1}{2}\left[\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} \mathrm{D}_l^{[0](J)} \\ \mathrm{D}_l^{[3](J)} \end{pmatrix} - i\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} \mathrm{D}_l^{[2](J)} \\ \mathrm{D}_l^{[1](J)} \end{pmatrix}\right]. \tag{3.100}$$

Moreover, the families of basis functions

$$\mathbf{E}_{\mathbb{C}}^{(J)} := \biguplus_{l=1}^{4^J-1} \left(\mathrm{E}_{l,\boldsymbol{n}}^{\nearrow(J)}, \mathrm{E}_{l,\boldsymbol{n}}^{\searrow(J)}, \mathrm{E}_{l,\boldsymbol{n}}^{\swarrow(J)}, \mathrm{E}_{l,\boldsymbol{n}}^{\nwarrow(J)}\right)_{\boldsymbol{n}\in\mathbb{Z}^2} \uplus \left(\mathrm{E}_{0,\boldsymbol{n}}^{(J)}\right)_{\boldsymbol{n}\in\mathbb{Z}^2}; \tag{3.101}$$

$$\boldsymbol{\Psi}_{\mathbb{C}}^{(J)} := \biguplus_{l=1}^{4^J-1} \left(\Psi_{l,\boldsymbol{n}}^{\nearrow(J)}, \Psi_{l,\boldsymbol{n}}^{\searrow(J)}, \Psi_{l,\boldsymbol{n}}^{\swarrow(J)}, \Psi_{l,\boldsymbol{n}}^{\nwarrow(J)}\right)_{\boldsymbol{n}\in\mathbb{Z}^2} \uplus \left(\Psi_{0,\boldsymbol{n}}^{(J)}\right)_{\boldsymbol{n}\in\mathbb{Z}^2}, \tag{3.102}$$

allow near-exact reconstruction of any input $\mathrm{X} \in \mathcal{J} := l^2_{\mathbb{C}}(\mathbb{Z}^2)$ (discrete framework) and $F \in \mathcal{V} \subset L^2_{\mathbb{C}}(\mathbb{R}^2)$ (3.80) (continuous framework), respectively. In the above expressions, the complex atoms are computed similarly to (3.95) and (3.86):

$$\begin{pmatrix} \mathrm{E}_{l,\boldsymbol{n}}^{\nearrow(J)} \\ \mathrm{E}_{l,\boldsymbol{n}}^{\searrow(J)} \end{pmatrix} = \frac{1}{2}\left[\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} \mathrm{E}_{l,\boldsymbol{n}}^{[0](J)} \\ \mathrm{E}_{l,\boldsymbol{n}}^{[3](J)} \end{pmatrix} + i\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} \mathrm{E}_{l,\boldsymbol{n}}^{[2](J)} \\ \mathrm{E}_{l,\boldsymbol{n}}^{[1](J)} \end{pmatrix}\right]; \tag{3.103}$$

$$\begin{pmatrix} \Psi_{l,\boldsymbol{n}}^{\nearrow(J)} \\ \Psi_{l,\boldsymbol{n}}^{\searrow(J)} \end{pmatrix} = \frac{1}{2}\left[\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} \Psi_{l,\boldsymbol{n}}^{[0](J)} \\ \Psi_{l,\boldsymbol{n}}^{[3](J)} \end{pmatrix} + i\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} \Psi_{l,\boldsymbol{n}}^{[2](J)} \\ \Psi_{l,\boldsymbol{n}}^{[1](J)} \end{pmatrix}\right], \tag{3.104}$$

where $\mathrm{E}_{l,\boldsymbol{n}}^{[m](j)} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ and $\Psi_{k,\boldsymbol{n}}^{[m](j)} \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ denote the atoms of the m-th pseudo-cosine orthonormal bases, satisfying (3.62) with $\mathbf{G} \leftarrow \mathbf{G}^{[m]}$ such as introduced in (3.87). Moreover, for the sake of consistency with standard pseudo-cosine bases (3.64), we have denoted $\Psi_{0,\boldsymbol{n}}^{(J)} := \Phi_{\boldsymbol{n}}^{(J)}$ in (3.102). In this context, DT-$\mathbb{C}$WPT tiles the frequency plane into 4×4^J overlapping square windows of size $\pi/2^{J-1}$. A more detailed analysis of this property will be conducted in Section 4.6.2.

3.4 The Wavelet Scattering Transform

In the previous sections, we covered various types of discrete wavelet transforms, implemented as convolutions with separable filters. Introducing redundant complex frames partly addressed the double issue of directional selectivity and stability to small input shifts. In this section, we take a step further and consider the more general problem of stability to deformations. According to Mallat (2012), convolutions with complex and analytic wavelets are stable to local deformations because their bandwidth is larger at higher frequencies. However, as seen in Section 3.3.3 in the context of DT-$\mathbb{C}$WT, near shift invariance only appears at coarser scales. A possible way to extend shift invariance to higher frequencies is to consider wavelet packet transforms, as covered in Section 3.3.4. Unfortunately though, increasing the Fourier localization at these frequencies results in decreasing the stability to deformations.

To tackle this issue, Mallat (2012) came out with an elegant solution, which computes cascading wavelet convolutions followed by nonlinear operations. The corresponding feature extractor is called a *wavelet scattering transform*. It produces nearly translation-invariant image representations which are stable to deformations and preserve high frequency information. The wavelet scattering transform shares the structure of a multilayer convolutional network (Bruna and Mallat, 2013). We therefore leave the framework introduced in Section 3.1.1 and enter one in which wavelets meet deep learning.

3.4.1 Lipschitz-Continuity to Diffeomorphisms

Another important property of feature extractors for image classification is stability with respect to small deformations such as translations, rotations, scaling, distortions, *etc.* Such deformations can be modelled by an operator $\mathcal{D}_{\boldsymbol{\tau}}$ characterized by a diffeomorphism $\boldsymbol{\tau} : \mathbb{R}^2 \to \mathbb{R}^2$, transforming any input $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ into

$$\mathcal{D}_{\boldsymbol{\tau}} F : \boldsymbol{x} \mapsto F(\boldsymbol{x} - \boldsymbol{\tau}(\boldsymbol{x})). \tag{3.105}$$

Note that the translation operator $\mathcal{T}_{\boldsymbol{h}}$ defined by $\mathcal{T}_{\boldsymbol{h}} F : \boldsymbol{x} \mapsto F(\boldsymbol{x} - \boldsymbol{h})$ is a particular case, where $\boldsymbol{\tau} : \boldsymbol{x} \mapsto \boldsymbol{h}$ is a constant mapping.

In the following, $\boldsymbol{\tau}$ is referred to as a *displacement field*. Considering an operator $\Gamma : L^2_{\mathbb{R}}(\mathbb{R}^2) \to E$, where E denotes a given Hilbert space, Bruna and Mallat (2013) characterize stability to deformations in terms of Lipschitz continuity. In its most general formulation, the Lipschitz continuity condition supposes the existence of a constant $C > 0$ such that, for any inputs $F, G \in L^2_{\mathbb{R}}(\mathbb{R}^2)$,

$$\|\Gamma(G) - \Gamma(F)\|_E \leq C \, \|G - F\|_{L^2} \, . \tag{3.106}$$

We now consider the more restrictive case of Lipschitz continuity to deformations. For any input $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ and any displacement field $\boldsymbol{\tau}$, applying condition (3.106) with $G \leftarrow \mathcal{D}_{\boldsymbol{\tau}} F$ yields (Mallat, 2012)

$$\|\Gamma(\mathcal{D}_{\boldsymbol{\tau}} F) - \Gamma(F)\|_E \leq C \, \|F\|_{L^2} \, \|\nabla \boldsymbol{\tau}\|_\infty \, , \tag{3.107}$$

where $\|\nabla \boldsymbol{\tau}\|_\infty := \sup_{\boldsymbol{y} \in \mathbb{R}^2} \|\nabla \boldsymbol{\tau}(\boldsymbol{y})\|$ denotes the maximum deformation amplitude of $\boldsymbol{\tau}$. The matrix norm $\|\nabla \boldsymbol{\tau}(\boldsymbol{y})\|$ measures the deformation amplitude of $\boldsymbol{\tau}$ at any location $\boldsymbol{y} \in \mathbb{R}^2$.

Expression (3.107) implies perfect translation invariance, because the Jacobian matrix of a constant displacement field is null. However, in many applications (including the windowed scattering transform as explained below), we only consider nearly shift-invariant operators; thus Lipschitz continuity as defined above is never met. A more useful approach consists in restricting (3.107) to deformations with a minimal normalized deformation amplitude, defined by

$$\rho(\boldsymbol{\tau}) := \|\nabla \boldsymbol{\tau}\|_\infty \,/\, \|\boldsymbol{\tau}\|_\infty \,, \tag{3.108}$$

with $\|\boldsymbol{\tau}\|_\infty := \sup_{\boldsymbol{y}} \|\boldsymbol{\tau}(\boldsymbol{y})\|_2$. Imposing $\rho(\boldsymbol{\tau}) \geq \rho_{\text{inf}}$ for a certain $\rho_{\text{inf}} > 0$, as implicitly done by Mallat (2012, Corol. 2.15), eliminates "flat" deformations such as translations. On the other hand, Lipschitz continuity can also be restricted to deformations up to some maximal amplitude: $\|\nabla \boldsymbol{\tau}\|_\infty \leq \upsilon_{\text{sup}}$.

The modulus of the Fourier transform $\Gamma : F \mapsto |\widehat{F}|$, where $\widehat{F}$ has been defined in (3.28), is invariant to global translation but is not Lipschitz continuous to deformations: the constant C in (3.107) becomes arbitrarily large at high frequencies. This has an impact on the stability of discrete representations in the Fourier basis such as introduced in Section 3.1.2. According to Mallat (2012), Lipschitz continuity is preserved by increasing the size of the Fourier support at higher frequencies. This is what happens with multiresolution transforms such as DT-$\mathbb{C}$WT. However, wavelet coefficients only become shift invariant (up to a phase shift) when j goes to infinity (*i.e.*, at low frequencies), as discussed in Section 3.3.3. In fact, DT-$\mathbb{C}$WT generally exhibits near-shift equivariance but not invariance at finer scales. A possible workaround is to further decompose high-frequency feature maps, leading to the wavelet packet transform—see Section 3.3.4. Unfortunately, we face the same issue as with the Fourier transform: decreasing the Fourier support of the basis functions increases the value of C in (3.107). This tradeoff between shift invariance and stability to deformations at finer scales can be circumvented by considering a multilayer wavelet transform, as outlined in the next section.

3.4.2 General Principles of the Wavelet Scattering Transform

For the sake of consistency with the rest of the chapter, the wavelet scattering transform is presented from the DT-$\mathbb{C}$WT perspective, as implemented by Singh and Kingsbury (2017). However, the concept is much more general and holds for any tight frame wavelet transform, as explained by Andén and Mallat (2014). In what follows, we consider the complex wavelet tight frame $\boldsymbol{\Psi}_{\mathbb{C}}^{(J)}$ as denoted in (3.84).

First-order Scattering Transform. Let $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ denote a 2D input signal and $J \in \mathbb{N} \setminus \{0\}$ denote the number of DT-$\mathbb{C}$WT decomposition stages. For a given scale $j \in \{1\,..\,J\}$ and orientation $k \in \{1\,..\,3\}$, we consider

$$F_k^{\nearrow(j)} := F * \overline{\Psi}_{k,\mathbf{0}}^{\nearrow(j)} \in L^2_{\mathbb{C}}(\mathbb{R}^2), \tag{3.109}$$

where $\Psi_{k,\boldsymbol{n}}^{\nearrow(j)} \in L^2_{\mathbb{C}}(\mathbb{R}^2)$ denotes a scaled and translated version of the mother wavelet $\Psi_k^{\nearrow}$ such as defined in (3.78). According to (3.83), a uniform sampling of $F_k^{\nearrow(j)}$ at interval 2^j yields the feature map of DT-$\mathbb{C}$WT coefficients $\mathrm{D}_k^{\nearrow(j)} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$. To improve translation invariance while keeping stability to deformations, it is possible to convolve $F_k^{\nearrow(j)}$ with

the dilated low-pass filter $\Phi^{(J)} := 2\,\Phi_{\mathbf{0}}^{(J)}$.[3] We get the following operator:

$$S_k^{\nearrow(j)} : F \mapsto \left(F * \overline{\Psi}_{k,\mathbf{0}}^{\nearrow(j)}\right) * \overline{\Phi}^{(J)}. \tag{3.110}$$

However, since $F_k^{\nearrow(j)}$ is high-frequency, filtering with $\Phi^{(J)}$ discards almost any information it contains. The solution proposed by Bruna and Mallat (2013) to avoid this is to apply a nonlinear pointwise operator on $F_k^{\nearrow(j)}$ before low-pass filtering. The most commonly used is the modulus operator, pushing the signal energy toward lower frequencies. Therefore, a significant proportion of the energy is captured by the low-pass filter $\Phi^{(J)}$. In this context, (3.110) is replaced by

$$S_k^{\nearrow(j)} : F \mapsto \left(U_k^{\nearrow(j)} F\right) * \overline{\Phi}^{(J)}, \qquad \text{with} \qquad U_k^{\nearrow(j)} : F \mapsto \left|F * \overline{\Psi}_{k,\mathbf{0}}^{\nearrow(j)}\right|. \tag{3.111}$$

Then, a uniform sampling of $S_k^{\nearrow(j)} F$ at interval 2^J yields *first-order scattering coefficients*. Note that the same reasoning holds for the three other Fourier quadrants $\searrow$, $\swarrow$ and $\nwarrow$. However, since input images are real-valued, we only need to compute the scattering coefficients for the top-right and bottom-right quadrants (six wavelet orientations).

Multilayer Scattering Transform. From now on, the operators introduced in (3.111) are respectively denoted, for any Fourier quadrant $q_1 \in \{\nearrow, \searrow\}$, scale $j_1 \in \{1\,..\,J\}$ and orientation $k_1 \in \{1\,..\,3\}$, by $S[\lambda_1]$ and $U[\lambda_1]$, where we have introduced the parameter $\lambda_1 := (q_1, j_1, k_1)$. Even when applying the modulus operator, filtering by $\Phi^{(J)}$ results in a loss of information, which can be recovered by performing DT-$\mathbb{C}$WT on $U[\lambda_1] F$. We then get second-order operators, defined by

$$S[\boldsymbol{\lambda}] : F \mapsto \left(U[\boldsymbol{\lambda}] F\right) * \overline{\Phi}^{(J)}, \qquad \text{with} \qquad U[\boldsymbol{\lambda}] := U[\lambda_2] \circ U[\lambda_1]\,. \tag{3.112}$$

However, since the modulus operator pushes energy toward lower frequencies, we can restrict the second-order scattering transform to coarser scales $j_2 \in \{j_1 + 1\,..\,J\}$, with marginal loss of information. Again, a uniform sampling of $S[\boldsymbol{\lambda}] F$ at interval 2^J yields *second-order scattering coefficients*. The principle can be further extended to deeper decompositions. However, the energy of $S[\boldsymbol{\lambda}] F$ has an exponential decay rate with increasing depth (Waldspurger, 2016). Therefore, in practice the wavelet scattering transform is generally performed up to the second order. Moreover, the low-frequency features discarded by the wavelet transform are directly captured by the low-pass filter $\Phi^{(J)}$. A uniform sampling of $S[\varnothing] F := F * \overline{\Phi}^{(J)}$ at interval 2^J yields *zeroth-order scattering coefficients*.

Let $P \in \mathbb{N} \setminus \{0\}$ denote the depth of the scattering transform, *i.e.*, the maximum order of scattering coefficients. For any scattering layer $p \in \{1\,..\,P\}$, we consider a family of parameters satisfying

$$\Lambda_p \subset \left(\{\nearrow, \searrow\} \times \{1\,..\,J\} \times \{1\,..\,3\}\right)^p. \tag{3.113}$$

Any parameter $\boldsymbol{\lambda} := (\lambda_1, \cdots, \lambda_p) \in \Lambda_p$ defines an "admissible path" of length p along which to compute the scattering transform $S[\boldsymbol{\lambda}] F$, following (3.112). More specifically, for any $p' \in \{1\,..\,p\}$, $\lambda_{p'} := (q_{p'}, j_{p'}, k_{p'})$ satisfies

$$j_{p'} \in \left\{(j_{p'-1} + 1)\,..\,J\right\}, \qquad \text{with} \qquad j_0 := 0. \tag{3.114}$$

[3] The factor 2 in front of $\Phi_{\mathbf{0}}^{(J)}$ is necessary for energy conservation (see Section 3.4.3), due to the redundancy of the DT-$\mathbb{C}$WT tight frame.

Note that, if $p > J$, then Λ_p is empty. Therefore, we can restrict the scattering transform to $P \leq J$. Besides, by extension we denote $\Lambda_0 := \{\varnothing\}$ (zeroth-order coefficients). Finally, we consider the union of all parameters:

$$\Lambda^{(P)} := \biguplus_{p=0}^{P} \Lambda_p. \tag{3.115}$$

In practice, the wavelet scattering coefficients, defined as a uniform sampling of $S[\boldsymbol{\lambda}]\, F$ at interval 2^J, can be approximated with multiple executions of the DT-$\mathbb{C}$WT algorithm described in Section 3.3.3, starting from a discrete input $\mathrm{X} := \mathrm{X}^{[0]} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$, encoding the linear projection of F onto $\mathcal{V}$ in the orthonormal basis $(\Phi_{\boldsymbol{n}})_{\boldsymbol{n}\in\mathbb{Z}^2}$. The corresponding feature maps are denoted by $\mathrm{C}_{\boldsymbol{\lambda}} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ for any $\boldsymbol{\lambda} \in \Lambda^{(P)}$, and satisfy

$$(S[\boldsymbol{\lambda}]\, F)(2^J \boldsymbol{n}) \approx \mathrm{C}_{\boldsymbol{\lambda}}[\boldsymbol{n}] \qquad \forall \boldsymbol{n} \in \mathbb{Z}^2. \tag{3.116}$$

3.4.3 Properties of the Wavelet Scattering Transform

The scattering properties, which have been proven by Mallat (2012), are summarized in this section. First, the wavelet scattering transform nearly preserves energy. For any $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$,

$$\|\Pi_{\mathcal{V}} F\|^2_{L^2} \approx \frac{1}{2} \sum_{p=0}^{J} \sum_{\boldsymbol{\lambda}\in\Lambda_p} \|S[\boldsymbol{\lambda}]\, F\|^2_{L^2}\,, \tag{3.117}$$

where $\Pi_{\mathcal{V}} : L^2_{\mathbb{R}}(\mathbb{R}^2) \to \mathcal{V}$ denotes the orthogonal projection onto the subspace $\mathcal{V}$ introduced in (3.80). The factor $1/2$ is due to the redundancy of the DT-$\mathbb{C}$WT tight frame. The above expression is only an approximation because of the constraint (3.114), which implies minor energy leaks. Moreover, the scattering energy has an exponential decay as a function of the path length p. This allows limiting the number of scattering layers in practice, generally restricted to $P = 2$.

In practice, a discrete scattering transform is performed on a discrete input $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$, as explained above. We then get

$$\|\Pi_{\mathcal{V}} F\|^2_{L^2} = \|\mathrm{X}\|^2_2 \approx \frac{1}{2} \sum_{p=0}^{J} \sum_{\boldsymbol{\lambda}\in\Lambda_p} \|\mathrm{C}_{\boldsymbol{\lambda}}\|^2_2\,. \tag{3.118}$$

Due to the low-pass filtering by $\Phi^{(J)}$, the wavelet scattering transform is nearly translation invariant:

$$\|\boldsymbol{h}\|_2 \ll 2^J \implies \forall \boldsymbol{\lambda} \in \Lambda^{(P)},\, S[\boldsymbol{\lambda}](\mathcal{T}_{\boldsymbol{h}} F) \approx S[\boldsymbol{\lambda}]\, F. \tag{3.119}$$

Besides, it is Lipschitz continuous to diffeomorphisms: (3.107) is satisfied with $\Gamma \leftarrow S^{(P)}$, where we have defined the scattering transform operator:

$$S^{(P)} : F \mapsto (S[\boldsymbol{\lambda}]\, F)_{\boldsymbol{\lambda}\in\Lambda^{(P)}}\,, \qquad \text{and} \qquad \left\|S^{(P)} F\right\|^2 := \sum_{p=0}^{J} \sum_{\boldsymbol{\lambda}\in\Lambda_p} \|S[\boldsymbol{\lambda}]\, F\|^2_{L^2}\,. \tag{3.120}$$

Note that Lipschitz-continuity to deformations is restricted to displacement fields $\boldsymbol{\tau}$ satisfying $\rho(\boldsymbol{\tau}) \geq 2^{-J}$ (3.108) and $\|\nabla\boldsymbol{\tau}\|_\infty \leq 1/2$. Moreover, input functions $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ must be compactly supported.

3.4.4 Deep Learning with Wavelet Scattering Networks

In Sections 3.4.2 and 3.4.3, we saw that the wavelet scattering transform satisfies both near-translation invariance and Lipschitz-continuity to deformations, while preserving high-frequency information. Moreover, it can be implemented as a multilayer convolution network. The current section is about deep learning applications for the wavelet scattering transform, seen as a CNN-like feature extractor.

Bruna and Mallat (2013) employed wavelet scattering coefficients as input for a linear classifier, in particular a support vector machine (SVM), as described in Section 2.1.3. The overall architecture, which is fully-deterministic except for the final classifier, is referred to as a *wavelet scattering network* (ScatterNet). The authors achieved state-of-the-art accuracy on handwritten digits and texture datasets. A variation has been proposed by Sifre and Mallat (2013) to include rotational invariance. In the field of audio signals, Andén and Mallat (2014) applied the scattering transform along log-frequencies to obtain transposition-invariant representations. Subsequent adaptations of the wavelet scattering transform include DT-$\mathbb{C}$WT-based ScatterNets (Singh and Kingsbury, 2017), which is the variant presented in Section 3.4.2, geometric ScatterNets on Riemanian manifolds (Perlmutter et al., 2020), and graph ScatterNets (Gama et al., 2019; D. Zou and Lerman, 2020).

ScatterNets achieve excellent results on small image datasets but do not scale well to more complex ones. According to Oyallon et al. (2017, 2018), this is partly due to non-geometric sources of variability within classes. Instead, these authors proposed to use scattering coefficients as input for a CNN, showing that the network complexity can be reduced while keeping competitive performance. More recent work (Zarka et al., 2020) proposed to sparsify wavelet scattering coefficients by learning a dictionary matrix, and managed to outperform AlexNet (Krizhevsky et al., 2017). This approach was later extended by the same team (Zarka et al., 2021), where the authors proposed to learn 1×1 convolutions between feature maps of scattering coefficients and to apply soft-thresholding to reduce within-class variability. This model reached the classification accuracy of ResNet-18 on ImageNet-1K. Other works proposed architectures in which the scattering transform is no longer deterministic. Cotter and Kingsbury (2019) built a learnable ScatterNet. In this model, feature maps of scattering coefficients are mixed together using trainable weights, to account for cross-channel filtering as implemented in CNNs—see Chapter 2 (2.48). Their architecture outperformed VGG networks on small image datasets. Recently, Gauthier et al. (2022) introduced parametric ScatterNets, in which the scale, orientation and aspect ratio of each wavelet filter are adjusted during training. Their approach has proven successful when trained on limited dataset.

All these papers are driven by the purpose of building ad-hoc CNN-like feature extractors, implementing well defined mathematical operators specifically designed to meet a certain number of desired properties. By contrast with freely-trained convolutional networks, ScatterNets can be of great interest for tasks where theoretical guarantees are required, for instance in the medical field. Moreover, standard CNNs contain a large number of parameters, demanding large datasets for effective training. In cases where the amount of input data is limited, utilizing more deterministic models like ScatterNets can enhance generalizability by reducing overfitting. Finally, the wavelet scattering transform can be seen as a tool to get a better understanding of CNNs from a mathematical point of view. We will come back to this in Section 3.5.3.

3.5 Wavelets Meet CNNs, Beyond Scattering Networks

ScatterNets are not the only attempt to combine wavelet analysis and deep neural networks. Section 3.5.1 is a brief overview of wavelet-enhanced CNNs. Then, in Section 3.5.2, we review the main theoretical studies on CNNs from the perspective of signal processing, including—but not limited to—ScatterNets. Finally, Section 3.5.3 is a wrap-up of the two background chapters, introducing the main contributions of this thesis.

3.5.1 Wavelet-Based Feature Extraction and CNNs

In a general machine learning context, discrete wavelet, wavelet packet and Gabor-like transforms have been widely used as feature extractors for signal, image and texture classification (Laine and Fan, 1993; Learned and Willsky, 1995; Pittner and Kamarthi, 1999; Yen, 2000; Subasi, 2007; C.-C. Liu and Dai, 2009; Khushaba et al., 2011; Dua et al., 2012; T. Li and Zhou, 2016).

In the deep learning framework, wavelet-based CNNs generally pursue one of the two following goals: improving classification performance, and / or reducing the model complexity. The central concept is to introduce prior assumptions about the network's behavior, considering the tendency of CNNs to spontaneously learn wavelet-like patterns, as explained in Section 2.4.1. Three main approaches are identified: (1) replace or complement freely-trained convolution kernels by wavelet or Gabor filters; (2) preprocess input data with wavelet transforms; (3) use wavelet transforms as a downsampling strategy in pooling layers.

Within the first scenario, Chang and Morgan (2014) proposed to initialize freely-trained convolution kernels with wavelet or Gabor functions, and let the network adjust the weights during training. Alternatively, additional channels performing discrete wavelet transform can be stacked to an existing CNN architecture (Fujieda et al., 2017; M. Liu et al., 2021). In a related spirit, some models include deterministic Gabor filters in parallel to freely-trained layers (Sarwar et al., 2017), freely-trained convolution kernels modulated by oriented Gabor filters (Luan et al., 2018), linear combinations of discrete cosine transforms (Ulicny et al., 2019), wavelet packet transforms (P. Liu et al., 2019), or parametric Gabor filters (Alekseev and Bobe, 2019; Pérez et al., 2020).

The second approach, data preprocessing, takes advantage of the sparsity and stability of wavelet representations to achieve equivalent or superior accuracy on image classification, with possibly reduced model complexity. The method proposed by Williams and R. Li (2016) consists in transforming input images through FWT (see Section 3.2.1), and using a specific CNN for each feature map of wavelet coefficients. Besides, as explained in Section 3.4.4, Oyallon et al. (2018) used wavelet scattering coefficients as input for a CNN. Finally, L. Liu et al. (2020) exploited the properties of wavelet representations to build a dual-branch network, using dense convolutions for local features and dilated convolutions for large receptive fields.

Regarding wavelet pooling, Williams and R. Li (2018) proposed the use of fast wavelet transform as an alternative downsampling strategy to standard average pooling. A similar approach has been undertaken by Lu et al. (2018), using DT-$\mathbb{C}$WT instead of FWT.

3.5.2 Theoretical Studies in CNNs

This section provides an overview of theoretical studies on CNNs. Notably, several papers have leveraged wavelet theory to deduce insights into deep learning models. The section begins by focusing on invariance, equivariance, and adversarial robustness, before broadening the discussion to encompass other important theoretical aspects of CNNs.

Invariance, Equivariance and Adversarial Robustness. Several papers analyze invariance properties of CNN-related feature extractors. Extensive studies related to the original wavelet scattering transform (see Section 3.4.3) have been proposed by Mallat (2012, 2016). Besides, Czaja and W. Li (2019, 2020) worked on a model variant based on uniform covering frames—*i.e.*, frames splitting the frequency domain into windows of roughly equal size. This is related to the pseudo-cosine frames such as described in Sections 3.2.5 and 3.3.4. Beyond the scattering transform, Wiatowski and Bölcskei (2018) considered a wide variety of feature extractors involving convolutions, Lipschitz-continuous non-linearities and pooling operators. These authors showed that outputs become more translation invariant with increasing network depth. Moreover, Weiler and Cesa (2021) examined equivariant properties of steerable CNNs with respect to the two-dimensional Euclidean group. They highlighted the importance of preventing aliasing when applying the theoretical results in practice. Finally, a series of papers addressed the question by modeling CNNs as *convolutional kernel networks* (CKNs) (Mairal et al., 2014). In particular, Bietti and Mairal (2019a) proved that certain classes of CNNs are contained into the reproducing kernel Hilbert space (RKHS) of a multilayer convolutional kernel representation. As such, stability metrics are estimated, based on the RKHS norm which is difficult to control in practice. Subsequent studies were conducted in this framework, including Bietti and Mairal (2019b), Scetbon and Harchaoui (2020), and Bietti (2022).

Other works explored the stability of CNNs to adversarial perturbations in a more generic sense, a subject which is also addressed by Szegedy et al. (2014). Stability is often measured in terms of Lipschitz continuity, satisfying (3.106) in its most general formulation. Virmaux and Scaman (2018) showed that an exact computation of the Lipschitz bound C_0 in CNNs—*i.e.*, smallest value of C satisfying (3.106)—is an NP hard problem, and proposed an algorithm to estimate this quantity in practice. Moreover, Balan et al. (2018) and D. Zou et al. (2020) proposed an estimation of the Lipschitz bound for generic models involving channel aggregation, as found in standard CNNs. A recent study by Pérez et al. (2020) proposed to control the Lipschitz constant of various CNN architectures by introducing discretized Gabor filters, for which a closed form expression of the Lipschitz constant is established.

Other Theoretical Aspects. For the sake of completeness, we review other theoretical aspects of CNNs. Mallat and Waldspurger (2015) proposed a more general formulation of ScatterNets by considering trainable unitary operators instead of wavelet transforms. Using a probabilistic framework, they showed that maximizing class separation, as explained in Section 2.4, is equivalent to minimizing the expected decay of scattering coefficients across the network—*i.e.*, in (3.117), minimizing the decay of $\sum_{\boldsymbol{\lambda}\in\Lambda_p} \|S[\boldsymbol{\lambda}]\,F\|_{L^2}^2$ when p increases. For the specific case of wavelet operators, Waldspurger (2016) proved that, under additional conditions, the scattering coefficients decay exponentially. Regarding complex convolutions, Tygert et al. (2016) showed that, under specific conditions on their

filters, complex-valued convolutional networks (using the modulus as a nonlinear pooling operator) can compute multiscale windowed power spectra, wavelet or wavelet packet moduli of locally-stationary input sequences.

On unrelated topics, Kuo (2016) proposed a study on pointwise nonlinear activation functions in CNNs. In a paper by J. C. Ye et al. (2018), a framework based on deep convolutional framelets is introduced. They established several theoretical results such as energy compaction, perfect reconstruction conditions under ReLU, number of channels and network depth depending on the signal intrinsic sparsity, role of skip connections such as found in ResNet.

Another desirable property that has been identified for CNNs is orthogonality of convolution kernels. According to Rodríguez et al. (2017) and Bansal et al. (2018), orthogonality improves training efficiency by enhancing robustness and decorrelating features. To promote near-orthogonality, J. Wang et al. (2020) proposed to regularize training with orthogonality constraints on convolution layers. Recently, Achour et al. (2022) provided theoretical insights into orthogonal CNNs, including stability guarantees for the aforementioned regularization strategy.

Furthermore, a class of convolutional models based on the *phase harmonics*, a nonlinear operator adjusting the phase of complex coefficients, has been developed by Mallat et al. (2020), S. Zhang and Mallat (2021), and Brochard et al. (2022). In particular, Mallat et al. (2020) showed that this operator admits a dual representation similar to rectifiers (*e.g.*, ReLU), when the convolution kernels are well-localized in the Fourier domain. Recently, Sander et al. (2022) investigated whether neural ordinary differential equations in the continuous framework and ResNets in the discrete framework exhibit similar behavior.

3.5.3 What is Missing?

As mentioned previously, there is an abundant literature on the properties of CNNs, many of which being linked to wavelet analysis. However, some questions remain unanswered to date. In this section, we outline the main motivations for the contribution chapters that follow.

ScatterNets are specifically designed to meet some desired properties. As deep learning architectures, they are sometimes used as explanatory models for standard, freely-trained networks. However, unlike the latter models, they implement convolutions with complex filters and take advantage of the modulus as a nonlinear operator. Therefore, as implied by Tygert et al. (2016), there is no exact correspondence between ScatterNets and real-valued CNNs. Put differently, a link is missing: can complex convolutions and moduli be adapted to more standard models, in which convolutions with real-valued kernels are followed by pointwise activation functions and nonlinear downsampling operators such as max pooling?

In most studies, invariance properties are obtained for continuous signals. Whereas CNNs can be mathematically described in the continuous framework, the feature maps computed at their hidden and output layers are actually discrete sequences, which can be considered as a sampling of the continuous outputs at fixed intervals. An example is provided in (3.116) for scattering coefficients. At each convolution and pooling layers, the sampling interval is increased, resulting in a loss of information. Unfortunately, this may greatly affect shift invariance due to aliasing effects. Section 3.2.6 provides an explanation for such a phenomenon, in the context of standard FWT. Convolutions with complex

and oriented filters may avoid this behavior, as explained in Section 3.3.3. Yet again, conventional architectures do not implement such filters. Actually, shift instability in CNNs is a well known phenomenon, as discussed in Section 2.4.2. This is what motivated R. Zhang (2019) to design antialiased convolutional networks. Another limitations of the aforementioned studies is the lack of consideration for max pooling.

Chapter 4 specifically addresses these questions. Our work seeks evidence that shift invariance properties—which is established for DT-$\mathbb{C}$WPT and ScatterNets—are, to some extent, embedded in standard, freely-trained architectures. In particular, we show that max pooling serves as a "demodulator" for real-valued, high-frequency feature maps, analogous to the role played by the modulus operator for complex feature maps in ScatterNets. However, as will be highlighted, the discrete nature of CNNs is a critical limitation to reach proper shift invariance. Depending on the filter's frequency and orientation, instabilities may persist. For this reason, in Chapter 5 we explore the possibility of adding an imaginary part to the real-valued feature maps, then replacing the max pooling layer by a modulus. By doing so, we establish a one-to-one connection between complex moduli such as found in ScatterNets and real max pooling such as implemented in freely-trained models.

Chapter 4

Shift Invariance of Max Pooling Feature Maps in CNNs

UNDERSTANDING THE MATHEMATICAL PROPERTIES of deep convolutional neural networks (CNNs) remains a challenging issue today. On the other hand, wavelet and multi-resolution analysis are built upon a well-established mathematical framework. We refer the reader to Chapters 2 and 3 for a general overview of both fields.

There is a broad literature revealing strong connections between these two paradigms, as discussed in Sections 3.4 and 3.5. Inspired by this line of research, the present chapter extends existing knowledge about CNN properties. Specifically, we assess the shift invariance of max pooling feature maps through both theoretical and empirical approaches in the context of image classification. When trained on image datasets such as MNIST, CIFAR-10 or ImageNet, CNNs tend to learn parameters in the first layer that closely resemble oriented Gabor filters, as outlined in Section 2.4.1. By leveraging the properties of discrete Gabor-like convolutions, we establish conditions under which the feature maps computed by the subsequent max pooling operator approximate the modulus of complex Gabor-like coefficients, in which case they are stable with respect to small input shifts. We then compute a probabilistic measure of shift invariance for these layers. More specifically, we show that some filters, depending on their frequency and orientation, are more likely than others to produce stable image representations. We experimentally validate our theory by considering a deterministic feature extractor based on the dual-tree wavelet packet transform, a particular case of discrete Gabor-like decomposition. We demonstrate a strong correlation between shift invariance on the one hand and similarity with complex modulus on the other hand.

This chapter is mainly adapted from the following preprint: H. Leterme, K. Polisano, V. Perrier, and K. Alahari (2022). "On the Shift Invariance of Max Pooling Feature Maps in Convolutional Neural Networks". arXiv: 2209.11740.

4.1 Motivations and Main Contributions

In many computer vision applications, including classification, input images are transformed through a non-linear operator $\boldsymbol{\Gamma}$, generally referred to as a feature extractor—see Chapter 2 (2.24). The output feature maps, which contain high-level information, can in turn be fed into deeper feature extractors. Specifically, deep neural networks contain a

nested sequence $\left(\boldsymbol{\Gamma}^{(p)}\right)_{p\in\{0..P-1\}}$ of nonlinear operators with a large number of trainable parameters, as detailed in (2.29) for a simple multilayer perceptron (MLP), and (2.57) for a CNN. The final classifier generally preforms multinomial logistic regression.

As discussed in Section 2.4, CNN feature extractors are expected to retain discriminant image components while decreasing intra-class variability. In particular, two shifted versions of a single image should in general receive the same label. As a result, their feature representations should exhibit a high degree of similarity. In this chapter, we set aside the broader concept of stability with respect to deformations such as discussed in Section 3.4.1, and focus on global shift invariance. It has been noted that many CNNs trained on natural image datasets perform some kind of discrete real-valued Gabor transform in their first layer, as covered in Section 2.4.1. In other words, images are decomposed through subsampled convolutions using filters with well-defined frequency and orientation. This observation, which is exploited in several papers (see Section 3.5.1), reveals the discriminative nature of CNNs' first layer. Whether such a layer can extract stable features is partly addressed by Azulay and Weiss (2019) and R. Zhang (2019). These papers point out that convolution and pooling layers may greatly diverge from shift invariance, due to aliasing when subsampling. Section 3.2.6 provides an intuition about the aliasing phenomenon. In response, recent work (R. Zhang, 2019; Vasconcelos et al., 2020; X. Zou et al., 2023) introduced antialiased convolution and pooling operators. They managed to increase both stability and predictive power of CNNs, despite the resulting loss of information.

In the current chapter, we show that, in certain situations, the first max pooling layer can actually *reduce* aliasing and therefore recover stability. Inspired by Waldspurger (2015, pp. 190–191), we unveil a connection between the output of this pooling operator and the modulus of complex Gabor-like coefficients, which is known to be nearly shift invariant. This work led us to design an alternative antialiasing solution based on complex-valued convolutions. Unlike the previously-mentioned papers, this approach preserves high-frequency information, as evidenced in Chapter 5.

4.1.1 Proposed Approach

Let $\mathrm{W} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ denote a band-pass, oriented and analytic *Gabor-like* filter, for which a formal definition will be provided in (4.12). We first consider an operator, referred to as *real-max-pooling* ($\mathbb{R}$Max), which computes the subsampled convolution of an input image $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ with the real part of W; then calculates the maximum value over a sliding discrete grid:

$$U^{\max}_{m,q}[\mathrm{W}] : \mathrm{X} \mapsto \mathrm{MaxPool}_q\left(\left(\mathrm{X} * \overline{\mathrm{Re}\,\mathrm{W}}\right) \downarrow m\right), \tag{4.1}$$

where $m \in \mathbb{N}\setminus\{0\}$ denotes a subsampling factor and $*, \downarrow$ respectively refer to the convolution and subsampling operations, introduced in (2.44) and (2.45). In the above expression, $\mathrm{MaxPool}_q$ selects the maximum value over a sliding grid of size $(2q+1)\times(2q+1)$, with a subsampling factor of 2. More formally, for any $\mathrm{Y} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ and any $\boldsymbol{n} \in \mathbb{Z}^2$,

$$\mathrm{MaxPool}_q(\mathrm{Y})[\boldsymbol{n}] := \max_{\|\boldsymbol{p}\|_\infty \leq q} \mathrm{Y}[2\boldsymbol{n}+\boldsymbol{p}]. \tag{4.2}$$

Therefore, $\mathrm{MaxPool}_q$ satisfies (2.54) with $m' \leftarrow 2$ and $\Omega_0 \leftarrow \Omega_0^{\max}$. On the other hand, we consider an operator, referred to as *complex-modulus* ($\mathbb{C}$Mod), computing the modulus

of subsampled convolution of X with W:

$$U_m^{\mathrm{mod}}[\mathrm{W}] : \mathrm{X} \mapsto \left| (\mathrm{X} * \overline{\mathrm{W}}) \downarrow (2m) \right|. \tag{4.3}$$

Remark 4.1. In the above definitions, convolutions are performed with a subsampling factor which is twice larger for $\mathbb{C}$Mod, compared to $\mathbb{R}$Max. However, since max pooling is also computed with subsampling, both operators have the same overall subsampling factor of $2m$.

First, we show that, under the Gabor hypothesis, $\mathbb{C}$Mod is stable with respect to small input shifts. We then establish conditions on the filter's frequency and orientation under which $\mathbb{C}$Mod and $\mathbb{R}$Max produce comparable outputs:

$$U_m^{\mathrm{mod}}[\mathrm{W}]\,(\mathrm{X}) \approx U_{m,\,q}^{\mathrm{max}}[\mathrm{W}]\,(\mathrm{X}). \tag{4.4}$$

We deduce a measure of shift invariance for $\mathbb{R}$Max operators, which benefits from the stability of $\mathbb{C}$Mod. Next, we extend our results to multichannel operators (*i.e.*, applied on RGB input images), such as implemented in conventional CNN architectures. Our framework therefore provides a theoretical grounding to study these networks.

We apply our theoretical results on the dual-tree complex wavelet packet transform (DT-$\mathbb{C}$WPT), a particular case of discrete Gabor-like decomposition with perfect reconstruction properties (see Section 3.3.4), possessing characteristics comparable to those of trained convolution layers in CNNs. Finally, we verify our predictions on a deterministic setting based on DT-$\mathbb{C}$WPT. Given an input image, we compute the mean discrepancy between the outputs of $\mathbb{C}$Mod and $\mathbb{R}$Max, for each wavelet packet filter.[1] We then observe that shift invariance, when measured on $\mathbb{R}$Max feature maps, is nearly achieved when they remain close to $\mathbb{C}$Mod outputs. We therefore establish a domain of validity for shift invariance of $\mathbb{R}$Max operators.

Prior to this work, we presented a preliminary study (Leterme et al., 2021), where we experimentally showed that an operator based on the real part of DT-$\mathbb{C}$WPT can mimic the behavior of the first convolution layer with fewer parameters, while keeping the network's predictive power. Our model was solely based on real-valued filters, which are known to be generally unstable (see Section 3.2.6). Yet, we observed a limited but genuine form of shift invariance, compared with other models based on the standard, non-analytic wavelet packet transform. At the same time, we became aware of a preliminary work by Waldspurger (2015, pp. 190–191), suggesting a potential connection between the combinations "real wavelet transform → max pooling" on the one hand and "complex wavelet transform → modulus" on the other hand. Following this idea, we decided to study whether invariance properties of complex moduli could somehow be captured by the max pooling operator. As shown in the present chapter, Waldspurger's work does not fully extend to discrete and subsampled convolutions. We address this issue by adopting a probabilistic point of view.

4.1.2 Related Work

In Chapter 3, we reviewed several papers in which wavelets meet deep learning. In Section 3.4, we discussed wavelet scattering networks (ScatterNets) (Bruna and Mallat, 2013),

[1] As seen in Section 3.3.4, DT-$\mathbb{C}$WPT paves the Fourier domain into square regions of identical size, each of them associated with a specific filter.

which compute several layers of complex wavelet convolutions followed by non-linear operations. They generate translation-invariant image representations that are stable to deformations and preserve high-frequency information. However, these mathematical properties cannot be directly applied to conventional CNN architectures for several reasons. Firstly, ScatterNets use convolutions with complex-valued wavelet filters, followed by a modulus operator, while standard CNNs implement real-valued convolutions, followed by a nonlinear pointwise activation, like ReLU, and downsampling operators, such as max pooling. Moreover, the scattering transform applies a nonlinear operator on feature maps with varying sizes. This is typical of multiresolution transforms, where the subsampling factor is typically larger at coarser scales. In contrast, in freely-trained models, each convolution layer is characterized by a unique subsampling factor shared across all output channels. In the present study, we take these specificities into account. In particular, our results have been tested using the pseudo-cosine version of DT-ℂWPT (see Section 3.3.4), which yields output feature maps of equal sizes.

ScatterNets are driven by the purpose of building ad-hoc CNN-like feature extractors, implementing well defined mathematical operators specifically designed to meet a certain number of desired properties. By contrast, our work seeks evidence that such properties are—to some extent—embedded in *existing* CNN architectures, with no need to alter their behavior or introduce new features.

Besides, several mathematical studies about CNNs have been reviewed in Section 3.5.2. As explained in Section 3.5.3 however, the commonly-used case of discrete, subsampled convolutions followed by ReLU and max pooling is not directly addressed. In some papers, invariance properties are obtained for continuous signals, thus ignoring aliasing effects due to subsampled convolutions (see Section 2.4.2). As for the wavelet scattering transform, it is designed for shift invariance in both continuous and discrete frameworks. However, as mentioned above, complex convolutions are not used in freely-trained networks; therefore, a link is missing to understand invariance properties in the latter models. Finally, kernel representations (Bietti and Mairal, 2019a) do not seem to suffer from aliasing effects. Yet, they consider linear, Gaussian pooling layers instead of max pooling. By discarding high-frequency information, shift invariance is preserved.

To summarize, this chapter tackles the question of shift invariance for real-valued CNNs, using max pooling as an nonlinear aggregator preserving high-frequency information. We remind that, as covered in Section 2.4.3, max pooling is known to produce superior performances over linear pooling operators.

4.2 Shift Invariance of ℂMod Outputs

The primary goal of this chapter is to theoretically establish conditions for near-shift invariance at the output of the first max pooling layer. In this section, we start by proving shift invariance of ℂMod operators. Then, in Section 4.3, we establish conditions under which ℝMax and ℂMod produce closely related outputs. Finally, in Section 4.4, we derive a probabilistic measure of shift invariance for ℝMax.

4.2.1 Notations

We introduce the notations used throughout this chapter. Some of them have already been used in the previous chapters, and are presented as a reminder, while others are specific

to the current chapter.

The complex conjugate of any number $z \in \mathbb{C}$ is denoted by z^*. For any $p \in \mathbb{R}_{>0} \cup \{\infty\}$, $\boldsymbol{x} \in \mathbb{R}^2$ and $r \in \mathbb{R}_+$, we denote by $B_p(\boldsymbol{x}, r) \subset \mathbb{R}^2$ the closed l^p-ball with center $\boldsymbol{x}$ and radius r. When $\boldsymbol{x} = \boldsymbol{0}$, we write $B_p(r)$.

Continuous Framework. Given $p > 0$ and a measurable subset of $\mathbb{R}$ or $\mathbb{R}^2$ denoted by E, we consider $L^p_{\mathbb{C}}(E)$ as the space of measurable complex-valued functions $F : E \to \mathbb{C}$ such that

$$\|F\|_{L^p} := \int_E |F(x)|^p \, \mathrm{d}x < +\infty. \tag{4.5}$$

Whenever we talk about equality in $L^p_{\mathbb{C}}(E)$ or inclusion in E, it shall be understood as "almost everywhere with respect to the Lebesgue measure". Besides, we denote by $L^2_{\mathbb{R}}(\mathbb{R}^2) \subset L^2_{\mathbb{C}}(\mathbb{R}^2)$ the subset of real-valued functions. For any $F \in L^2_{\mathbb{C}}(\mathbb{R}^2)$, $\overline{F}$ denotes its flipped version: $\overline{F}(\boldsymbol{x}) := F(-\boldsymbol{x})$.

The 2D Fourier transform of any $F \in L^2_{\mathbb{C}}(\mathbb{R}^2)$ is denoted by $\widehat{F} \in L^2_{\mathbb{C}}(\mathbb{R}^2)$, such that

$$\forall \boldsymbol{\nu} \in \mathbb{R}^2, \, \widehat{F}(\boldsymbol{\nu}) := \iint_{\mathbb{R}^2} F(\boldsymbol{x}) \mathrm{e}^{-i\langle \boldsymbol{\nu}, \boldsymbol{x} \rangle} \, \mathrm{d}^2\boldsymbol{x}. \tag{4.6}$$

For any $\varepsilon > 0$ and $\boldsymbol{\nu} \in \mathbb{R}^2$, we denote by $\mathcal{V}(\boldsymbol{\nu}, \varepsilon) \subset L^2_{\mathbb{C}}(\mathbb{R}^2)$ the set of functions whose Fourier transform is supported in a square region of size $\varepsilon \times \varepsilon$ centered in $\boldsymbol{\nu}$:

$$\mathcal{V}(\boldsymbol{\nu}, \varepsilon) := \left\{ \Psi \in L^2_{\mathbb{C}}(\mathbb{R}^2) \;\middle|\; \operatorname{supp} \widehat{\Psi} \subset B_\infty(\boldsymbol{\nu}, \varepsilon/2) \right\}. \tag{4.7}$$

For any $\boldsymbol{h} \in \mathbb{R}^2$, we also consider the translation operator, denoted by $\mathcal{T}_{\boldsymbol{h}}$, defined by

$$\mathcal{T}_{\boldsymbol{h}} F : \boldsymbol{x} \mapsto F(\boldsymbol{x} - \boldsymbol{h}). \tag{4.8}$$

Discrete Framework. We consider $l^2_{\mathbb{C}}(\mathbb{Z}^d)$ as the space of d-dimensional sequences $\mathrm{X} \in \mathbb{C}^{\mathbb{Z}^d}$ such that

$$\|\mathrm{X}\|_2^2 := \sum_{\boldsymbol{n} \in \mathbb{Z}^d} \left|\mathrm{X}[\boldsymbol{n}]\right|^2 < +\infty. \tag{4.9}$$

Indexing is made between square brackets: $\forall \mathrm{X} \in l^2_{\mathbb{C}}(\mathbb{Z}^d)$, $\forall \boldsymbol{n} \in \mathbb{Z}^d$, $\mathrm{X}[\boldsymbol{n}] \in \mathbb{C}$, and we denote by $l^2_{\mathbb{R}}(\mathbb{Z}^d) \subset l^2_{\mathbb{C}}(\mathbb{Z}^d)$ the subset of real-valued sequences. For any $\mathrm{X} \in l^2_{\mathbb{C}}(\mathbb{Z}^d)$, $\overline{\mathrm{X}}$ denotes its flipped version: $\overline{\mathrm{X}}[\boldsymbol{n}] := \mathrm{X}[-\boldsymbol{n}]$. The subsampling operator is denoted by $\downarrow$: for any $\mathrm{X} \in l^2_{\mathbb{C}}(\mathbb{Z}^d)$ and any $m \in \mathbb{N} \setminus \{0\}$, $(\mathrm{X} \downarrow m)[\boldsymbol{n}] := \mathrm{X}[m\boldsymbol{n}]$.

2D images, feature maps and convolution kernels are considered as elements of $l^2_{\mathbb{C}}(\mathbb{Z}^2)$, and are denoted by straight capital letters. Besides, arrays of 2D sequences are denoted by bold straight capital letters, for instance: $\mathbf{X} = (\mathrm{X}_k)_{k \in \{0..K-1\}}$. Note that indexing starts at 0 to comply with practical implementations.

The 2D discrete-time Fourier transform of any $\mathrm{X} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ is denoted by

$$\widehat{\mathrm{X}} \in L^2_{\mathbb{C}}([-\pi, \pi]^2), \tag{4.10}$$

such that

$$\forall \boldsymbol{\theta} \in [-\pi, \pi]^2, \, \widehat{\mathrm{X}}(\boldsymbol{\theta}) := \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathrm{X}[\boldsymbol{n}] \mathrm{e}^{-i\langle \boldsymbol{\theta}, \boldsymbol{n} \rangle}. \tag{4.11}$$

For any $\kappa \in]0,\, 2\pi]$ and $\boldsymbol{\theta} \in B_\infty(\pi)$, we denote by $\mathcal{J}(\boldsymbol{\theta},\, \kappa) \subset l^2_{\mathbb{C}}(\mathbb{Z}^2)$ the set of 2D sequences whose Fourier transform is supported in a square region of size $\kappa \times \kappa$ centered in $\boldsymbol{\theta}$:

$$\mathcal{J}(\boldsymbol{\theta},\, \kappa) := \left\{ \mathrm{W} \in l^2_{\mathbb{C}}(\mathbb{Z}^2) \;\middle|\; \operatorname{supp} \widehat{\mathrm{W}} \subset B_\infty(\boldsymbol{\theta},\, \kappa/2) \right\}. \tag{4.12}$$

Remark 4.2. The support $B_\infty(\boldsymbol{\theta},\, \kappa/2)$ actually lives in the quotient space $[-\pi,\, \pi]^2 \,/\, (2\pi\mathbb{Z}^2)$. Consequently, when $\boldsymbol{\theta}$ is close to an edge, a fraction of this region is located at the far end of the frequency domain. From now on, the choice of $\boldsymbol{\theta}$ and κ is implicitly assumed to avoid such a situation.

4.2.2 Intuition

In many CNNs for computer vision, input images are first transformed through subsampled (or strided) convolutions. For instance, in AlexNet, convolution kernels are of size 11×11 and the subsampling factor is equal to 4. Figure 1.1 displays the corresponding kernels after training with ImageNet. This linear transform is generally followed by rectified linear unit (ReLU) and max pooling.

We can observe that many kernels display oscillating patterns with well-defined orientations (Gabor-like filters). We denote by $\mathrm{V} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ one of these "well-behaved" filters. Its Fourier spectrum roughly consists in two bright spots which are symmetric with respect to the origin.[2] Now, we consider a complex-valued companion $\mathrm{W} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ such that

$$\widehat{\mathrm{W}}(\boldsymbol{\omega}) := (1 + \operatorname{sgn}\langle \boldsymbol{\omega},\, \boldsymbol{u} \rangle) \cdot \widehat{\mathrm{V}}(\boldsymbol{\omega}) \qquad \forall \boldsymbol{\omega} \in [-\pi,\, \pi]^2\,, \tag{4.13}$$

where $\boldsymbol{u}$ denotes a unit vector orthogonal to the filter's orientation—see Section 5.2 for a more detailed discussion on this topic.

We can show that V is the real part of W, and that $\mathrm{W} = \mathrm{V} + i\mathcal{H}(\mathrm{V})$, where $\mathcal{H}$ denotes the two-dimensional Hilbert transform as introduced by Havlicek et al. (1997). Analogously to (3.76) for wavelets in the continuous framework, $\mathcal{H}(\mathrm{V})$ satisfies

$$\widehat{\mathcal{H}(\mathrm{V})}(\boldsymbol{\omega}) := -i \operatorname{sgn}\langle \boldsymbol{\omega},\, \boldsymbol{u} \rangle\, \widehat{\mathrm{V}}(\boldsymbol{\omega}). \tag{4.14}$$

As a consequence, $\widehat{\mathrm{W}}$ is equal to $2\widehat{\mathrm{V}}$ on one half of the Fourier domain, and 0 on the other half. Therefore, only one bright spot remains in the spectrum. We refer the reader to Figure 5.1 for visual example of complex-valued Gabor-like filter. It turns out that such complex filters with high frequency resolution produce stable signal representations, as we will see in Section 4.2. In the subsequent sections, we then wonder whether this property is kept when considering the max pooling of real-valued convolutions.

In what follows, W will be referred to as a discrete Gabor-like filter, and the coefficients resulting from the convolution with W will be referred to as discrete Gabor-like coefficients. The aim of this section is to show that, under the Gabor hypothesis on the convolution kernels $\mathrm{W} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$, ℂMod is nearly shift-invariant. To clarify, we establish that

$$U_m^{\mathrm{mod}}[\mathrm{W}]\,(\mathrm{X}) \approx U_m^{\mathrm{mod}}[\mathrm{W}]\,(\mathcal{T}_{\boldsymbol{u}}\mathrm{X}), \tag{4.15}$$

for "small" translation vectors $\boldsymbol{u} \in \mathbb{R}^2$, where a formal definition of the translation operator will be defined in (4.41). This result is hinted by Kingsbury and Magarey (1998) but not formally proven.

[2] Actually, the Fourier transform of any real-valued sequence is centrally symmetric: $\widehat{\mathrm{V}}(-\boldsymbol{\omega}) = \widehat{\mathrm{V}}(\boldsymbol{\omega})^*$. The specificity of well-oriented filters lies in the concentration of their power spectrum around two precise locations.

4.2.3 Continuous Framework

We introduce several results regarding functions defined on the continuous space $\mathbb{R}^2$. Near-shift invariance on discrete 2D sequences will then be derived from these results by taking advantage of sampling theorems. Lemma 4.1 below is adapted from Waldspurger (2015, pp. 190–191).

Lemma 4.1. *Given $\varepsilon > 0$ and $\boldsymbol{\nu} \in \mathbb{R}^2$, let $\Psi \in \mathcal{V}(\boldsymbol{\nu}, \varepsilon)$ denote a complex-valued filter such as defined in (4.7). Now, for any real-valued function $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$, we consider the complex-valued function $F_0 \in L^2_{\mathbb{C}}(\mathbb{R}^2)$ defined by*

$$F_0 : \boldsymbol{x} \mapsto (F * \overline{\Psi})(\boldsymbol{x})\, \mathrm{e}^{i\langle \boldsymbol{\nu}, \boldsymbol{x} \rangle}. \tag{4.16}$$

Then F_0 is low-frequency. Specifically,

$$\operatorname{supp} \widehat{F_0} \subset B_\infty(\varepsilon/2). \tag{4.17}$$

Proof. Applying the Fourier transform on (4.16) yields, for any $\boldsymbol{\xi} \in \mathbb{R}^2$,

$$\widehat{F_0}(\boldsymbol{\xi}) = \widehat{(F * \overline{\Psi})}(\boldsymbol{\xi} - \boldsymbol{\nu}) = \mathcal{T}_{\boldsymbol{\nu}}(\widehat{F}\, \widehat{\overline{\Psi}})(\boldsymbol{\xi}). \tag{4.18}$$

By hypothesis on Ψ, we have

$$\operatorname{supp}(\widehat{F}\, \widehat{\overline{\Psi}}) \subset \operatorname{supp} \widehat{\overline{\Psi}} \subset B_\infty(-\boldsymbol{\nu}, \varepsilon/2). \tag{4.19}$$

The translation operator $\mathcal{T}_{\boldsymbol{\nu}}$ shifts the support with respect to $\boldsymbol{\nu}$, which yields (4.17). □

On the other hand, the following proposition provides a shift invariance bound for low-frequency functions such as introduced above.

Proposition 4.1. *For any $F_0 \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ such that $\operatorname{supp} \widehat{F_0} \subset B_\infty(\varepsilon/2)$, and any $\boldsymbol{h} \in \mathbb{R}^2$,*

$$\|\mathcal{T}_{\boldsymbol{h}} F_0 - F_0\|_{L^2} \leq \alpha(\varepsilon \boldsymbol{h})\, \|F_0\|_{L^2}\,, \tag{4.20}$$

where we have defined

$$\alpha : \boldsymbol{\tau} \mapsto \frac{\|\boldsymbol{\tau}\|_1}{2}. \tag{4.21}$$

Proof. Using the 2D Plancherel formula, we compute

$$\begin{aligned}
\|\mathcal{T}_{\boldsymbol{h}} F_0 - F_0\|_{L^2}^2 &= \frac{1}{4\pi^2} \left\| \widehat{\mathcal{T}_{\boldsymbol{h}} F_0} - \widehat{F_0} \right\|_{L^2}^2 \\
&= \frac{1}{4\pi^2} \iint_{B_\infty(\varepsilon/2)} \left| \widehat{F_0}(\boldsymbol{\xi}) \right|^2 \left| \mathrm{e}^{-i\langle \boldsymbol{h}, \boldsymbol{\xi} \rangle} - 1 \right|^2 \mathrm{d}^2\boldsymbol{\xi} \\
&= \frac{1}{4\pi^2} \iint_{B_\infty(\varepsilon/2)} \left| \widehat{F_0}(\boldsymbol{\xi}) \right|^2 (2 - 2\cos\langle \boldsymbol{h}, \boldsymbol{\xi} \rangle)\, \mathrm{d}^2\boldsymbol{\xi} \\
&\leq \frac{1}{4\pi^2} \iint_{B_\infty(\varepsilon/2)} \left| \widehat{F_0}(\boldsymbol{\xi}) \right|^2 |\langle \boldsymbol{h}, \boldsymbol{\xi} \rangle|^2\, \mathrm{d}^2\boldsymbol{\xi},
\end{aligned}$$

because $\cos t \geq 1 - \frac{t^2}{2}$. Note that the integral is computed on a compact domain because, according to Lemma 4.1, $\operatorname{supp} \widehat{F_0} \subset B_\infty(\varepsilon/2)$. Now, we use the Cauchy-Schwarz inequality to compute:

$$\begin{aligned} \forall \boldsymbol{\xi} \in B_\infty(\varepsilon/2),\ |\langle \boldsymbol{h}, \boldsymbol{\xi} \rangle| &\leq \|\boldsymbol{h}\|_1 \cdot \|\boldsymbol{\xi}\|_\infty \\ &\leq \frac{\varepsilon}{2} \|\boldsymbol{h}\|_1 . \end{aligned}$$

Therefore,

$$\|\mathcal{T}_{\boldsymbol{h}} F_0 - F_0\|_{L^2}^2 \leq \frac{\varepsilon}{4} \|\boldsymbol{h}\|_1^2 \, \|F_0\|_{L^2}^2 , \tag{4.22}$$

which yields the result. □

4.2.4 Adaptation to Discrete 2D Sequences

Given $\kappa \in]0,\, 2\pi]$ and $\boldsymbol{\theta} \in B_\infty(\pi)$, let $\mathrm{W} \in \mathcal{J}(\boldsymbol{\theta},\, \kappa)$ denote a discrete Gabor-like filter such as defined in (4.12). For any image $\mathrm{X} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ with finite support and any subsampling factor $m \in \mathbb{N} \setminus \{0\}$, we express $(\mathrm{X} * \overline{\mathrm{W}}) \downarrow m$ using the continuous framework introduced above, and derive an invariance formula.

For any sampling interval $s \in \mathbb{R}_{>0}$, let $\Phi^{(s)} \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ denote the Shannon scaling function parameterized by s, such that

$$\widehat{\Phi^{(s)}} := s \mathbb{1}_{B_\infty(\pi/s)}. \tag{4.23}$$

This 2D function is a tensor product of scaled and normalized sinc functions. For any $\boldsymbol{n} \in \mathbb{Z}^2$, we denote by $\Phi^{(s)}_{\boldsymbol{n}}$ a shifted version of $\Phi^{(s)}$, satisfying

$$\Phi^{(s)}_{\boldsymbol{n}}(\boldsymbol{x}) := \Phi^{(s)}(\boldsymbol{x} - s\boldsymbol{n}). \tag{4.24}$$

Then, similarly to (3.46), $\{\Phi^{(s)}_{\boldsymbol{n}}\}_{\boldsymbol{n} \in \mathbb{Z}^2}$ is an orthonormal basis of

$$\mathcal{V}^{(s)} := \{G \in L^2_{\mathbb{C}}(\mathbb{R}^2) \mid \operatorname{supp} \widehat{G} \subset B_\infty(\pi/s)\}. \tag{4.25}$$

Then, using the notation introduced in (4.7), we have $\mathcal{V}^{(s)} = \mathcal{V}(\mathbf{0},\, 2\pi/s)$.

Remark 4.3. In Chapter 3, we introduced a more restrictive framework, in which we have denoted $\mathcal{V}^{\mathrm{sh}} := \mathcal{V}^{(s)}$ and $\Phi^{\mathrm{sh}}_{\boldsymbol{n}} := \Phi^{(s)}_{\boldsymbol{n}}$, with $s := 1$ (distance-preserving mapping between discrete and continuous frameworks). In this chapter, we consider a more generic formulation using an arbitrary sampling interval $s > 0$. In addition to be more physically interpretable (the sampling interval can receive physical distance units), this notation will be of practical interest in subsequent sections, when considering multiples of this initial value, such as in Lemma 4.4. Moreover, we consider $\mathcal{V}^{(s)}$ as a $\mathbb{C}$-vector space.

We now consider the following lemma.

Lemma 4.2. *Let $s > 0$. For any $G \in \mathcal{V}^{(s)}$ and any $\boldsymbol{\xi} \in B_\infty(\pi/s)$, we have*

$$\widehat{G}(\boldsymbol{\xi}) = s\, \widehat{\mathrm{Y}}(s\boldsymbol{\xi}), \tag{4.26}$$

where $\mathrm{Y} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ is a uniform sampling of G, defined such that $\mathrm{Y}[\boldsymbol{n}] := s\, G(s\boldsymbol{n})$, for any $\boldsymbol{n} \in \mathbb{Z}^2$. Besides, we have the following norm equality:

$$\|G\|_{L^2} = \|\mathrm{Y}\|_2 . \tag{4.27}$$

Proof. Since $G \in \mathcal{V}^{(s)}$, the two-dimensional version of Shannon's sampling theorem (Mallat, 2009, Theorem 3.11, p. 81) yields

$$G = \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathrm{Y}[\boldsymbol{n}]\, \Phi_{\boldsymbol{n}}^{(s)}, \qquad \text{and} \qquad \widehat{G} = \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathrm{Y}[\boldsymbol{n}]\, \widehat{\Phi_{\boldsymbol{n}}^{(s)}}. \tag{4.28}$$

Besides, using (4.23), we can show that, for any $\boldsymbol{\xi} \in B_\infty(\pi/s)$,

$$\widehat{\Phi_{\boldsymbol{n}}^{(s)}}(\boldsymbol{\xi}) = \widehat{\Phi^{(s)}}(\boldsymbol{\xi})\, \mathrm{e}^{-i\langle s\boldsymbol{\xi}, \boldsymbol{n}\rangle} = s\, \mathrm{e}^{-i\langle s\boldsymbol{\xi}, \boldsymbol{n}\rangle}. \tag{4.29}$$

Therefore, plugging (4.29) into (4.28) proves (4.26).

Then, by combining (4.26) with the Plancherel formula, we get

$$\begin{aligned} \|G\|_{L^2}^2 &= \frac{1}{4\pi^2} \|\widehat{G}\|_{L^2}^2 \\ &= \frac{1}{4\pi^2} \iint_{B_\infty(\pi/s)} \left|\widehat{G}(\boldsymbol{\xi})\right|^2 \mathrm{d}^2\boldsymbol{\xi} \\ &= \frac{1}{4\pi^2} \iint_{B_\infty(\pi/s)} \left|s\, \widehat{\mathrm{Y}}(s\boldsymbol{\xi})\right|^2 \mathrm{d}^2\boldsymbol{\xi}. \end{aligned}$$

The integral is performed on $B_\infty(\pi/s)$ because $G \in \mathcal{V}^{(s)}$. Then, by applying the change of variable $\boldsymbol{\xi}' \leftarrow s\boldsymbol{\xi}$, we get

$$\begin{aligned} \|G\|_{L^2}^2 &= \frac{1}{4\pi^2} \iint_{B_\infty(\pi)} \left|\widehat{\mathrm{Y}}(\boldsymbol{\xi}')\right|^2 \mathrm{d}^2\boldsymbol{\xi}' \\ &= \frac{1}{4\pi^2} \|\widehat{\mathrm{Y}}\|_{L^2}^2 = \|\mathrm{Y}\|_2^2, \end{aligned}$$

hence (4.27), which concludes the proof. □

We then get the following proposition. In a similar spirit as Section 3.2.2, it draws a bond between the discrete and continuous frameworks.

Proposition 4.2. *Let* $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ *denote an input image with finite support, and* $\mathrm{W} \in \mathcal{J}(\boldsymbol{\theta}, \kappa)$*. Considering a sampling interval* $s \in \mathbb{R}_{>0}$*, we define* F_{X} *and* $\Psi_{\mathrm{W}} \in \mathcal{V}^{(s)}$ *such that*

$$F_{\mathrm{X}} := \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathrm{X}[\boldsymbol{n}]\, \Phi_{\boldsymbol{n}}^{(s)} \qquad \textit{and} \qquad \Psi_{\mathrm{W}} := \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathrm{W}[\boldsymbol{n}]\, \Phi_{\boldsymbol{n}}^{(s)}. \tag{4.30}$$

Then,

$$\Psi_{\mathrm{W}} \in \mathcal{V}(\boldsymbol{\theta}/s, \kappa/s). \tag{4.31}$$

Moreover, for all $\boldsymbol{n} \in \mathbb{Z}$*,*

$$\mathrm{X}[\boldsymbol{n}] = s\, F_{\mathrm{X}}(s\boldsymbol{n}); \qquad \mathrm{W}[\boldsymbol{n}] = s\, \Psi_{\mathrm{W}}(s\boldsymbol{n}), \tag{4.32}$$

and, for a given subsampling factor $m \in \mathbb{N} \setminus \{0\}$*,*

$$\left((\mathrm{X} * \overline{\mathrm{W}}) \downarrow m\right)[\boldsymbol{n}] = \left(F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}}\right)(ms\boldsymbol{n}). \tag{4.33}$$

Proof. First, F_{X} and Ψ_{W} are well defined because $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ and $\mathrm{W} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$. By construction, F_{X} and $\Psi_{\mathrm{W}} \in \mathcal{V}^{(s)}$. Therefore, according to Shannon's sampling theorem (Mallat, 2009, Theorem 3.11, p. 81),

$$F_{\mathrm{X}} := s \sum_{\boldsymbol{n} \in \mathbb{Z}^2} F_{\mathrm{X}}(s\boldsymbol{n})\, \Phi^{(s)}_{\boldsymbol{n}} \qquad \text{and} \qquad \Psi_{\mathrm{W}} := s \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \Psi_{\mathrm{W}}(s\boldsymbol{n})\, \Phi^{(s)}_{\boldsymbol{n}}. \tag{4.34}$$

By uniqueness of decompositions in an orthonormal basis, we get (4.32). Moreover, using (4.26) in Lemma 4.2, we get, for any $\boldsymbol{\xi} \in B_\infty(\pi/s)$,

$$\widehat{\Psi}_{\mathrm{W}}(\boldsymbol{\xi}) = s\, \widehat{\mathrm{W}}(s\boldsymbol{\xi}). \tag{4.35}$$

Since $\widehat{\Psi}_{\mathrm{W}}(\boldsymbol{\xi}) = 0$ outside $B_\infty(\pi/s)$, (4.35) is true for any $\boldsymbol{\xi} \in \mathbb{R}^2$. Therefore, by hypothesis on W,

$$\operatorname{supp} \widehat{\Psi}_{\mathrm{W}} \subset B_\infty(\boldsymbol{\theta}/s,\, \kappa/(2s)), \tag{4.36}$$

which yields (4.31).

We now prove (4.33). For $\boldsymbol{n} \in \mathbb{Z}^2$, we compute:

$$\begin{aligned}
(F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}})\,(ms\boldsymbol{n}) &= \iint_{\mathbb{R}^2} F_{\mathrm{X}}(ms\boldsymbol{n} - \boldsymbol{x})\, \overline{\Psi}_{\mathrm{W}}(\boldsymbol{x})\, \mathrm{d}^2\boldsymbol{x} \\
&= \iint_{\mathbb{R}^2} \sum_{\boldsymbol{p} \in \mathbb{Z}^2} \mathrm{X}[\boldsymbol{p}] \Phi^{(s)}_{\boldsymbol{p}}(ms\boldsymbol{n} - \boldsymbol{x})\, \overline{\Psi}_{\mathrm{W}}(\boldsymbol{x})\, \mathrm{d}^2\boldsymbol{x} \\
&= \sum_{\boldsymbol{p} \in \mathbb{Z}^2} \mathrm{X}[\boldsymbol{p}] \iint_{\mathbb{R}^2} \Phi^{(s)}_{\boldsymbol{p}}(ms\boldsymbol{n} - \boldsymbol{x})\, \overline{\Psi}_{\mathrm{W}}(\boldsymbol{x})\, \mathrm{d}^2\boldsymbol{x}.
\end{aligned}$$

The sum-integral interchange is possible because X has a finite support. Then:

$$(F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}})\,(ms\boldsymbol{n}) = \sum_{\boldsymbol{p} \in \mathbb{Z}^2} \mathrm{X}[\boldsymbol{p}] \iint_{\mathbb{R}^2} \overline{\Psi}_{\mathrm{W}}(\boldsymbol{x})\, \Phi^{(s)}(s(m\boldsymbol{n} - \boldsymbol{p}) - \boldsymbol{x})\, \mathrm{d}^2\boldsymbol{x} \tag{4.37}$$

$$= \sum_{\boldsymbol{p} \in \mathbb{Z}^2} \mathrm{X}[\boldsymbol{p}]\, (\overline{\Psi}_{\mathrm{W}} * \Phi^{(s)})(s(m\boldsymbol{n} - \boldsymbol{p})) \tag{4.38}$$

Since $\{\Phi^{(s)}_{\boldsymbol{n}}\}_{\boldsymbol{n} \in \mathbb{Z}^2}$ is an orthonormal basis of $\mathcal{V}^{(s)}$, the definition of Ψ_{W} in (4.30) implies, for any $\boldsymbol{p}' \in \mathbb{Z}^2$,

$$\overline{\mathrm{W}}[\boldsymbol{p}'] = \left\langle \Psi_{\mathrm{W}},\, \Phi^{(s)}_{-\boldsymbol{p}'} \right\rangle = \left(\overline{\Psi}_{\mathrm{W}} * \Phi^{(s)}\right)(s\boldsymbol{p}'), \tag{4.39}$$

because $\Phi^{(s)}$ is even. Therefore, plugging (4.39) with $\boldsymbol{p}' \leftarrow (m\boldsymbol{n} - \boldsymbol{p})$ into (4.38) yields

$$(F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}})\,(ms\boldsymbol{n}) = \sum_{\boldsymbol{p} \in \mathbb{Z}^2} \mathrm{X}[\boldsymbol{p}]\, \overline{\mathrm{W}}[m\boldsymbol{n} - \boldsymbol{p}] = (\mathrm{X} * \overline{\mathrm{W}})[m\boldsymbol{n}], \tag{4.40}$$

hence the result. □

Proposition 4.2 introduces a latent subspace of $L^2_{\mathbb{R}}(\mathbb{R}^2)$ from which input images are uniformly sampled. This allows us to define, for any $\boldsymbol{u} \in \mathbb{R}^2$, a translation operator $\mathcal{T}_{\boldsymbol{u}}$ on discrete sequences, even if $\boldsymbol{u}$ has non-integer values:

$$\mathcal{T}_{\boldsymbol{u}} \mathrm{X}[\boldsymbol{n}] := s\, \mathcal{T}_{s\boldsymbol{u}} F_{\mathrm{X}}(s\boldsymbol{n}), \tag{4.41}$$

where F_{X} is defined in (4.30). We can indeed show that this definition is independent from the choice of sampling interval $s > 0$. Besides, given $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$, we have

$$\forall \boldsymbol{p} \in \mathbb{Z}^2,\ \mathcal{T}_{\boldsymbol{p}}\mathrm{X}[\boldsymbol{n}] = \mathrm{X}[\boldsymbol{n} - \boldsymbol{p}]; \tag{4.42}$$

$$\forall \boldsymbol{u},\ \boldsymbol{v} \in \mathbb{R}^2,\ \mathcal{T}_{\boldsymbol{u}}(\mathcal{T}_{\boldsymbol{v}}\mathrm{X}) = \mathcal{T}_{\boldsymbol{u}+\boldsymbol{v}}\mathrm{X}, \tag{4.43}$$

which shows that $\mathcal{T}_{\boldsymbol{u}}$ corresponds to the intuitive idea of a translation operator. Expressions (4.42) and (4.43) are direct consequence of the following lemma, which bonds the shift operator in the discrete and continuous frameworks.

Lemma 4.3. *For any* $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ *and any* $\boldsymbol{u} \in \mathbb{R}^2$,

$$F_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}} = \mathcal{T}_{s\boldsymbol{u}}F_{\mathrm{X}}. \tag{4.44}$$

Proof. Let $\boldsymbol{u} \in \mathbb{R}^2$. By definition of $F_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}}$ and $\mathcal{T}_{\boldsymbol{u}}\mathrm{X}$,

$$F_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}} = s \sum_{\boldsymbol{n}\in\mathbb{Z}^2} \mathcal{T}_{s\boldsymbol{u}}F_{\mathrm{X}}(s\boldsymbol{n})\, \Phi^{(s)}_{\boldsymbol{n}}. \tag{4.45}$$

On the other hand, $F_{\mathrm{X}} \in \mathcal{V}^{(s)}$ by construction. Therefore, $\mathcal{T}_{s\boldsymbol{u}}F_{\mathrm{X}} \in \mathcal{V}^{(s)}$. Then, according to Shannon's sampling theorem (Mallat, 2009, Theorem 3.11, p. 81), we get

$$\mathcal{T}_{s\boldsymbol{u}}F_{\mathrm{X}} = s \sum_{\boldsymbol{n}\in\mathbb{Z}^2} \mathcal{T}_{s\boldsymbol{u}}F_{\mathrm{X}}(s\boldsymbol{n})\, \Phi^{(s)}_{\boldsymbol{n}}, \tag{4.46}$$

which concludes the proof. □

We now consider the following corollary to Proposition 4.2.

Corollary 4.1. *For any shift vector* $\boldsymbol{u} \in \mathbb{R}^2$, *we have*

$$\left((\mathcal{T}_{\boldsymbol{u}}\mathrm{X} * \overline{\mathrm{W}}) \downarrow m\right)[\boldsymbol{n}] = \left(\mathcal{T}_{s\boldsymbol{u}}F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}}\right)(ms\boldsymbol{n})\,. \tag{4.47}$$

Proof. Applying (4.33) in Proposition 4.2 with $\mathrm{X} \leftarrow \mathcal{T}_{\boldsymbol{u}}\mathrm{X}$, we get

$$\left((\mathcal{T}_{\boldsymbol{u}}\mathrm{X} * \overline{\mathrm{W}}) \downarrow m\right)[\boldsymbol{n}] = \left(F_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}} * \overline{\Psi}_{\mathrm{W}}\right)(ms\boldsymbol{n})\,, \tag{4.48}$$

and Lemma 4.3 concludes the proof. □

4.2.5 Shift Invariance in the Discrete Framework

We consider the $\mathbb{C}$Mod operator defined in (4.3). For the sake of conciseness, in what follows we will write U^{mod}_m instead of $U^{\mathrm{mod}}_m[\mathrm{W}]$, when no ambiguity is possible. First, we state the following lemma.

Lemma 4.4. *For any input image* $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ *with finite support, and any Gabor-like filter* $\mathrm{W} \in \mathcal{J}(\boldsymbol{\theta}, \kappa)$, *we consider the low-frequency function*

$$F_0 : \boldsymbol{x} \mapsto (F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}})(\boldsymbol{x})\, \mathrm{e}^{i\langle \boldsymbol{\theta}/s,\, \boldsymbol{x}\rangle}, \tag{4.49}$$

with F_{X} *and* Ψ_{W} *satisfying* (4.30). *If* $\kappa \leq \pi/m$, *then*

$$F_0 \in \mathcal{V}^{(s')}. \tag{4.50}$$

Moreover, for any $\boldsymbol{h} \in \mathbb{R}^2$,

$$\sum_{\boldsymbol{n}\in\mathbb{Z}^2} \left|\mathcal{T}_{\boldsymbol{h}} F_0(s'\boldsymbol{n}) - F_0(s'\boldsymbol{n})\right|^2 = \frac{1}{s'^2} \left\|\mathcal{T}_{\boldsymbol{h}} F_0 - F_0\right\|_{L^2}^2 , \tag{4.51}$$

where we have denoted $s' := 2ms$. *Finally,*

$$\left\|U_m^{\mathrm{mod}} \mathrm{X}\right\|_2 = \frac{1}{s'} \left\|F_0\right\|_{L^2} . \tag{4.52}$$

Proof. Let us write:

$$\sum_{\boldsymbol{n}\in\mathbb{Z}^2} \left|\mathcal{T}_{\boldsymbol{h}} F_0(s'\boldsymbol{n}) - F_0(s'\boldsymbol{n})\right|^2 = \sum_{\boldsymbol{n}\in\mathbb{Z}^2} |G(s'\boldsymbol{n})|^2 = \frac{1}{s'^2} \left\|\mathrm{Y}\right\|_2^2 , \tag{4.53}$$

where we have denoted, for any $\boldsymbol{n} \in \mathbb{Z}^2$,

$$G := \mathcal{T}_{\boldsymbol{h}} F_0 - F_0 \qquad \text{and} \qquad \mathrm{Y}[\boldsymbol{n}] := s' G(s'\boldsymbol{n}). \tag{4.54}$$

According to Proposition 4.2 (4.31), $\Psi_{\mathrm{W}} \in \mathcal{V}(\boldsymbol{\theta}/s,\, \kappa/s)$. Therefore, according to Lemma 4.1,

$$\operatorname{supp} \widehat{F_0} \subset B_\infty \left(\frac{\kappa}{2s}\right) . \tag{4.55}$$

Moreover, by hypothesis, $\kappa \leq \pi/m$; thus,

$$B_\infty \left(\frac{\kappa}{2s}\right) \subset B_\infty \left(\frac{\pi}{s'}\right) , \tag{4.56}$$

which yields (4.50), and $G \in \mathcal{V}^{(s')}$. Then, according to Lemma 4.2 (4.27) with $s \leftarrow s'$,

$$\left\|\mathrm{Y}\right\|_2 = \left\|G\right\|_{L^2} = \left\|\mathcal{T}_{\boldsymbol{h}} F_0 - F_0\right\|_{L^2} . \tag{4.57}$$

Therefore, plugging (4.57) into (4.53) yields (4.51).

Besides, according again to Lemma 4.2,

$$\left\|F_0\right\|_{L^2}^2 = \left\|\mathrm{X}_0\right\|_2^2 , \tag{4.58}$$

where we have defined, for any $\boldsymbol{n} \in \mathbb{Z}^2$,

$$\mathrm{X}_0[\boldsymbol{n}] := s' F_0(s'\boldsymbol{n}). \tag{4.59}$$

Then,

$$\begin{aligned}
\left\|\mathrm{X}_0\right\|_2^2 &= s'^2 \sum_{\boldsymbol{n}\in\mathbb{Z}^2} \left|(F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}})(s'\boldsymbol{n})\right|^2 && \text{(acc. to (4.49))} \\
&= s'^2 \sum_{\boldsymbol{n}\in\mathbb{Z}^2} \left|(\mathrm{X} * \overline{\mathrm{W}}) \downarrow (2m)[\boldsymbol{n}]\right|^2 && \text{(acc. to Proposition 4.2 with } m \leftarrow 2m) \\
&= s'^2 \left\|U_m^{\mathrm{mod}} \mathrm{X}\right\|_2^2 . && \text{(acc. to (4.3))}
\end{aligned}$$

Finally, plugging this result into (4.58) yields (4.52) and concludes the proof. □

We are now ready to state the main result about shift invariance of $\mathbb{C}$Mod outputs.

Theorem 4.1 (Shift invariance of $\mathbb{C}$Mod)**.** *Let* $\mathrm{W} \in \mathcal{J}(\boldsymbol{\theta}, \kappa)$ *denote a discrete Gabor-like filter and* $m \in \mathbb{N} \setminus \{0\}$ *denote a subsampling factor. Then, under the following condition:*

$$\kappa \leq \pi/m, \tag{4.60}$$

we have, for any input image $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ *with finite support and any translation vector* $\boldsymbol{u} \in \mathbb{R}^2$*,*

$$\left\|U_m^{\mathrm{mod}}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X}) - U_m^{\mathrm{mod}}\mathrm{X}\right\|_2 \leq \alpha(\kappa\boldsymbol{u}) \left\|U_m^{\mathrm{mod}}\mathrm{X}\right\|_2, \tag{4.61}$$

where α *has been defined in* (4.21)*.*

Proof. As in Lemma 4.4, we consider the low-frequency function F_0 satisfying (4.49), and denote $s' := 2ms$. We can write

$$|F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}}| = |F_0| \qquad \text{and} \qquad |\mathcal{T}_{s\boldsymbol{u}}F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}}| = |\mathcal{T}_{s\boldsymbol{u}}F_0|. \tag{4.62}$$

Recall that $U_m^{\mathrm{mod}}\mathrm{X} = |(\mathrm{X} * \overline{\mathrm{W}}) \downarrow (2m)|$, such as defined in (4.3). According to Proposition 4.2 (4.33) and Corollary 4.1 (4.47) with $m \leftarrow 2m$, we therefore get

$$U_m^{\mathrm{mod}}\mathrm{X}[\boldsymbol{n}] = \left|F_0(s'\boldsymbol{n})\right|; \tag{4.63}$$

$$U_m^{\mathrm{mod}}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X})[\boldsymbol{n}] = \left|(\mathcal{T}_{s\boldsymbol{u}}F_0)(s'\boldsymbol{n})\right|. \tag{4.64}$$

Then, using (4.63), (4.64) and the reverse triangle inequality,

$$\begin{aligned}\left\|U_m^{\mathrm{mod}}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X}) - U_m^{\mathrm{mod}}\mathrm{X}\right\|_2^2 &= \sum_{\boldsymbol{n}\in\mathbb{Z}^2} \Big|\left|(\mathcal{T}_{s\boldsymbol{u}}F_0)(s'\boldsymbol{n})\right| - \left|F_0(s'\boldsymbol{n})\right|\Big|^2 \\ &\leq \sum_{\boldsymbol{n}\in\mathbb{Z}^2} \left|(\mathcal{T}_{s\boldsymbol{u}}F_0)(s'\boldsymbol{n}) - F_0(s'\boldsymbol{n})\right|^2.\end{aligned}$$

Since condition (4.60) is satisfied, we can use Lemma 4.4 (4.51) with $\boldsymbol{h} \leftarrow s\boldsymbol{u}$:

$$\left\|U_m^{\mathrm{mod}}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X}) - U_m^{\mathrm{mod}}\mathrm{X}\right\|_2^2 \leq \frac{1}{s'^2}\left\|\mathcal{T}_{s\boldsymbol{u}}F_0 - F_0\right\|_{L^2}^2 \tag{4.65}$$

Now, according to Proposition 4.1 with $\varepsilon \leftarrow \kappa/s$ and $\boldsymbol{h} \leftarrow s\boldsymbol{u}$, we then get the following bound:

$$\left\|U_m^{\mathrm{mod}}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X}) - U_m^{\mathrm{mod}}\mathrm{X}\right\|_2^2 \leq \frac{\alpha(\kappa\boldsymbol{u})^2}{s'^2}\left\|F_0\right\|_{L^2}^2. \tag{4.66}$$

Finally, using Lemma 4.4 (4.52) yields (4.61), which completes the proof. □

Interestingly, the reference value used in Theorem 4.1, *i.e.*, $\left\|U_m^{\mathrm{mod}}\mathrm{X}\right\|_2$, is fully shift-invariant, as stated in the following proposition.

Proposition 4.3. *Let* $\mathrm{W} \in \mathcal{J}(\boldsymbol{\theta}, \kappa)$ *and* $m \in \mathbb{N} \setminus \{0\}$*. Under condition* (4.60)*, we have, for any* $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ *and any* $\boldsymbol{u} \in \mathbb{R}^2$*,*

$$\left\|U_m^{\mathrm{mod}}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X})\right\|_2 = \left\|U_m^{\mathrm{mod}}\mathrm{X}\right\|_2. \tag{4.67}$$

Proof. Let $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ and $s > 0$. We consider $F_0 \in L^2_{\mathbb{C}}(\mathbb{R}^2)$ as the "low-frequency" function satisfying (4.49). Again, we introduce $s' := 2ms$ and $\mathrm{X}_0 \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ satisfying (4.59). Moreover, for any $\mathrm{Y} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$, we denote by $F_{\mathrm{Y}}^{(s')}$ the Shannon interpolation of Y parameterized by s', analogously to (4.30):

$$F_{\mathrm{Y}}^{(s')} := \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathrm{Y}[\boldsymbol{n}]\, \Phi_{\boldsymbol{n}}^{(s')}. \tag{4.68}$$

On the one hand, Lemma 4.4 provides (4.52). On the other hand, we seek a similar result with $\mathrm{X} \leftarrow \mathcal{T}_{\boldsymbol{u}}\mathrm{X}$. For this purpose, (4.64) can be rewritten

$$U_m^{\mathrm{mod}}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X})[\boldsymbol{n}] = |\mathcal{T}_{s'\boldsymbol{u}'} F_0(s'\boldsymbol{n})|, \tag{4.69}$$

with $\boldsymbol{u}' := \boldsymbol{u}/(2m)$. Besides, according to Lemma 4.4 (4.50), $F_0 \in \mathcal{V}^{(s')}$. Therefore, Shannon's sampling theorem (Mallat, 2009, Theorem 3.11, p. 81) with $s \leftarrow s'$ yields

$$\begin{aligned} F_0 &= s' \sum_{\boldsymbol{n} \in \mathbb{Z}^2} F_0(s'\boldsymbol{n})\, \Phi_{\boldsymbol{n}}^{(s')} \\ &= \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathrm{X}_0[\boldsymbol{n}]\, \Phi_{\boldsymbol{n}}^{(s')} = F_{\mathrm{X}_0}^{(s')}, \end{aligned}$$

where we have used the notations introduced in (4.59) and (4.68). Then, using Lemma 4.3 with $\mathrm{X} \leftarrow \mathrm{X}_0$, $\boldsymbol{u} \leftarrow \boldsymbol{u}'$ and $s \leftarrow s'$, we get

$$F_{\mathcal{T}_{\boldsymbol{u}'}\mathrm{X}_0}^{(s')} = \mathcal{T}_{s'\boldsymbol{u}'} F_{\mathrm{X}_0}^{(s')} = \mathcal{T}_{s'\boldsymbol{u}'} F_0. \tag{4.70}$$

Besides, (4.32) (from Proposition 4.2) with $\mathrm{X} \leftarrow \mathcal{T}_{\boldsymbol{u}'}\mathrm{X}_0$ and $s \leftarrow s'$ becomes

$$\mathcal{T}_{\boldsymbol{u}'}\mathrm{X}_0[\boldsymbol{n}] = s'\, F_{\mathcal{T}_{\boldsymbol{u}'}\mathrm{X}_0}^{(s')}(s'\boldsymbol{n}), \tag{4.71}$$

and inserting (4.70) into (4.71) yields

$$\mathcal{T}_{\boldsymbol{u}'}\mathrm{X}_0[\boldsymbol{n}] = s'\, \mathcal{T}_{s'\boldsymbol{u}'} F_0(s'\boldsymbol{n}). \tag{4.72}$$

Therefore, (4.69) and (4.72) imply

$$\left\| U_m^{\mathrm{mod}}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X}) \right\|_2 = \frac{1}{s'} \left\| \mathcal{T}_{\boldsymbol{u}'}\mathrm{X}_0 \right\|_2. \tag{4.73}$$

Moreover, since $F_0 \in \mathcal{V}^{(s')}$, and according to (4.72), we can use Lemma 4.2 with $s \leftarrow s'$, $G \leftarrow \mathcal{T}_{s'\boldsymbol{u}'} F_0$ and $\mathrm{Y} \leftarrow \mathcal{T}_{\boldsymbol{u}'}\mathrm{X}_0$. We get

$$\left\| \mathcal{T}_{\boldsymbol{u}'}\mathrm{X}_0 \right\|_2 = \left\| \mathcal{T}_{s'\boldsymbol{u}'} F_0 \right\|_{L^2} = \left\| F_0 \right\|_{L^2}, \tag{4.74}$$

and plugging (4.74) into (4.73) yields

$$\left\| U_m^{\mathrm{mod}}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X}) \right\|_2 = \frac{1}{s'} \left\| F_0 \right\|_{L^2}. \tag{4.75}$$

Finally, considering Lemma 4.4 (4.52) concludes the proof. □

4.3 From ℂMod to ℝMax

ℂMod operators are found in ScatterNets (see Section 3.4) and complex-valued convolutional networks (Tygert et al., 2016). However, they are absent from conventional, freely-trained CNN architectures. Therefore, Theorem 4.1 cannot be applied as is. Instead, the first convolution layer contains real-valued kernels, and is generally followed by ReLU and max pooling. As shown in Section 4.5, this process can be described with ℝMax operators, such as defined in (4.1).

As explained in Section 2.4.1, an important number of trained convolution kernels exhibit oscillating patterns with well-defined frequencies and orientations. To elaborate, let $\mathrm{V} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ denote such a trained kernel, and consider $\mathrm{W} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ as the complex-valued companion of V satisfying (4.13). Then, W has its energy concentrated in a small region of the Fourier domain. We thus emit the hypotheses that $\mathrm{W} \in \mathcal{J}(\boldsymbol{\theta}, \kappa)$ (4.12) for a certain value of $\boldsymbol{\theta} \in [-\pi, \pi]^2$ and $\kappa \in \,]0, 2\pi]$. For the sake of conciseness, from now on we write $U^{\max}_{m,q}$ instead of $U^{\max}_{m,q}[\mathrm{W}]$, when no ambiguity is possible. In what follows, we establish conditions on W under which ℂMod (4.3) and ℝMax (4.1) operators produce comparable outputs. The final goal, achieved in Section 4.4, is to provide a shift invariance bound for ℝMax.

To give an intuition about why ℝMax may act as a proxy for ℂMod, we place ourselves in the continuous framework. Consider the real-valued wavelet transform output $\operatorname{Re} G := F * \operatorname{Re} \overline{\Psi}$, employed in ℝMax, as the real part of the complex-valued wavelet transform output $G := F * \overline{\Psi}$, used in ℂMod. At a given location $\boldsymbol{x} \in \mathbb{R}^2$, the corresponding imaginary part may carry a large amount of information, which somehow needs to be retrieved. The key idea is that, if Ψ is sufficiently localized in the Fourier domain, then only the phase of G significantly varies in the vicinity of $\boldsymbol{x}$, whereas its magnitude remains nearly constant. Therefore, finding the maximum value of $\operatorname{Re} G$ within a local neighborhood around $\boldsymbol{x}$ is nearly equivalent to shifting the phase of $G(\boldsymbol{x})$ towards 0. The resulting value then approximates $|G(\boldsymbol{x})|$. To put it differently, max pooling pushes energy towards lower frequencies, in a similar way as the modulus does for complex-valued transforms (Bruna and Mallat, 2013). This result is hinted in Section 4.3.1.

Regretfully, things do not work so smoothly in the discrete case. At first glance, this is surprising because Shannon's sampling theorem allows to cast discrete problems into the continuous framework, as done in Section 4.2.4. However, as explained in Section 4.3.2, max pooling operates over a discrete grid instead of a continuous window. Consequently, in some situations, the maximum value may fall far away from any zero-phase coefficient. Taking into account this behavior, we adopt a probabilistic point of view, as detailed in Section 4.3.4. Then, we provide in Section 4.3.5 an upper bound for the expected gap between ℂMod and ℝMax outputs.

4.3.1 Continuous Framework

This section, inspired from Waldspurger (2015, pp. 190–191), provides an intuition about resemblance between ℝMax and ℂMod in the continuous framework. As will be highlighted in Section 4.3.2, adaptation to discrete 2D sequences is not straightforward and will require a probabilistic approach.

We consider an input function $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ and a band-pass filter $\Psi \in \mathcal{V}(\boldsymbol{\nu}, \varepsilon)$. Let us

also consider

$$G : (\boldsymbol{x},\, \boldsymbol{h}) \mapsto \cos\big(\langle \boldsymbol{\nu},\, \boldsymbol{h}\rangle - H(\boldsymbol{x})\big), \tag{4.76}$$

where $H : \mathbb{R}^2 \to [0,\, 2\pi[$ denotes the phase of $F * \overline{\Psi}$. Lemma 4.1 introduced low-frequency functions F_0, with slow variations. In a nutshell, since $\operatorname{supp} F_0 \subset B_\infty(\varepsilon/2)$, we can write

$$\|\boldsymbol{h}\|_2 \ll \lambda_{F_0} \implies F_0(\boldsymbol{x}+\boldsymbol{h}) \approx F_0(\boldsymbol{x}), \tag{4.77}$$

where we have defined $\lambda_{F_0} := 2\pi/\varepsilon$. Therefore, according to Proposition 4.4 below, we get the following approximation of $F * \operatorname{Re}\overline{\Psi}$ in a neighborhood around any point $\boldsymbol{x} \in \mathbb{R}^2$:

$$\|\boldsymbol{h}\|_2 \ll \lambda_{F_0} \implies (F * \operatorname{Re}\overline{\Psi})(\boldsymbol{x}+\boldsymbol{h}) \approx \big|(F * \overline{\Psi})(\boldsymbol{x})\big|\, G(\boldsymbol{x},\boldsymbol{h}). \tag{4.78}$$

Proposition 4.4. *For any* $\boldsymbol{h} \in \mathbb{R}^2$,

$$\Big|(F * \operatorname{Re}\overline{\Psi})(\boldsymbol{x}+\boldsymbol{h}) - \big|(F * \overline{\Psi})(\boldsymbol{x})\big|\, G(\boldsymbol{x},\boldsymbol{h})\Big| \leq \big|F_0(\boldsymbol{x}+\boldsymbol{h}) - F_0(\boldsymbol{x})\big|. \tag{4.79}$$

Proof. Let us write:

$$\begin{aligned}
&(F * \operatorname{Re}\overline{\Psi})(\boldsymbol{x}+\boldsymbol{h}) - \big|(F * \overline{\Psi})(\boldsymbol{x})\big|\, G(\boldsymbol{x},\boldsymbol{h}) \\
&\qquad = \operatorname{Re}\Big((F * \overline{\Psi})(\boldsymbol{x}+\boldsymbol{h})\Big) - \big|(F * \overline{\Psi})(\boldsymbol{x})\big|\, \operatorname{Re}\Big(\mathrm{e}^{-i\langle \boldsymbol{\nu},\, \boldsymbol{h}\rangle}\, \mathrm{e}^{H(\boldsymbol{x})}\Big) \\
&\qquad = \operatorname{Re}\Big((F * \overline{\Psi})(\boldsymbol{x}+\boldsymbol{h})\Big) - \operatorname{Re}\Big(\big|(F * \overline{\Psi})(\boldsymbol{x})\big|\, \mathrm{e}^{H(\boldsymbol{x})}\, \mathrm{e}^{-i\langle \boldsymbol{\nu},\, \boldsymbol{h}\rangle}\Big) \\
&\qquad = \operatorname{Re}\Big((F * \overline{\Psi})(\boldsymbol{x}+\boldsymbol{h})\Big) - \operatorname{Re}\Big((F * \overline{\Psi})(\boldsymbol{x})\, \mathrm{e}^{-i\langle \boldsymbol{\nu},\, \boldsymbol{h}\rangle}\Big) \\
&\qquad = \operatorname{Re}\Big((F * \overline{\Psi})(\boldsymbol{x}+\boldsymbol{h}) - (F * \overline{\Psi})(\boldsymbol{x})\, \mathrm{e}^{-i\langle \boldsymbol{\nu},\, \boldsymbol{h}\rangle}\Big).
\end{aligned}$$

Therefore,

$$\begin{aligned}
\Big|(F * \operatorname{Re}\overline{\Psi})(\boldsymbol{x}+\boldsymbol{h}) - \big|(F * \overline{\Psi})(\boldsymbol{x})\big|\, G(\boldsymbol{x},\boldsymbol{h})\Big| &\leq \Big|(F * \overline{\Psi})(\boldsymbol{x}+\boldsymbol{h}) - (F * \overline{\Psi})(\boldsymbol{x})\, \mathrm{e}^{-i\langle \boldsymbol{\nu},\, \boldsymbol{h}\rangle}\Big| \\
&= \Big|F_0(\boldsymbol{x}+\boldsymbol{h})\, \mathrm{e}^{-i\langle \boldsymbol{\nu},\, \boldsymbol{x}+\boldsymbol{h}\rangle} - F_0(\boldsymbol{x})\, \mathrm{e}^{-i\langle \boldsymbol{\nu},\, \boldsymbol{x}+\boldsymbol{h}\rangle}\Big|,
\end{aligned}$$

which yields (4.79) and concludes the proof. □

On the one hand, we consider a continuous equivalent of the ℂMod operator $U_m^{\mathrm{mod}}[\mathrm{W}]$ as introduced in (4.3). Such an operator, denoted by $U^{\mathrm{mod}}[\Psi]$, is defined, for any $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$, by

$$U^{\mathrm{mod}}[\Psi]\,(F) : \boldsymbol{x} \mapsto \Big|(F * \overline{\Psi})(\boldsymbol{x})\Big|. \tag{4.80}$$

On the other hand, we consider the continuous counterpart of ℝMax as introduced in (4.1). It is defined as the maximum value of $F * \operatorname{Re}\overline{\Psi}$ over a sliding spatial window of size $r > 0$. This is possible because F and $\operatorname{Re}\overline{\Psi}$ both belong to $L^2_{\mathbb{R}}(\mathbb{R}^2)$, and therefore $F * \operatorname{Re}\overline{\Psi}$ is continuous. Such an operator, denoted by $U_r^{\max}[\Psi]$, is defined, for any $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$, by

$$U_r^{\max}[\Psi]\,(F) : \boldsymbol{x} \mapsto \max_{\|\boldsymbol{h}\|_\infty \leq r} (F * \operatorname{Re}\overline{\Psi})(\boldsymbol{x}+\boldsymbol{h}). \tag{4.81}$$

For the sake of conciseness, the parameter between square brackets is ignored from now on. If $r \ll \lambda_{F_0}$, then (4.78) is valid for any $\boldsymbol{h} \in B_\infty(r)$. Then, using (4.80) and (4.81), we get

$$r \ll \lambda_{F_0} \implies U_r^{\max} F(\boldsymbol{x}) \approx U^{\mathrm{mod}} F(\boldsymbol{x}) \max_{\|\boldsymbol{h}\|_\infty \leq r} G(\boldsymbol{x}, \boldsymbol{h}). \tag{4.82}$$

Using the periodicity of G, we can show that, if $r \geq \frac{\pi}{\|\boldsymbol{\nu}\|_2}$, then $\boldsymbol{h} \mapsto G(\boldsymbol{x}, \boldsymbol{h})$ necessarily reaches its maximum value (*i.e.*, 1) on $B_\infty(r)$. We therefore get

$$\frac{\pi}{\|\boldsymbol{\nu}\|_2} \leq r \ll \frac{2\pi}{\varepsilon} \implies U_r^{\max} F(\boldsymbol{x}) \approx U^{\mathrm{mod}} F(\boldsymbol{x}). \tag{4.83}$$

4.3.2 Adaptation to Discrete 2D Sequences

As in Section 4.2.4, we consider an input image $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$, a complex, analytic convolution kernel $\mathrm{W} \in \mathcal{J}(\boldsymbol{\theta}, \kappa)$, a subsampling factor $m \in \mathbb{N} \setminus \{0\}$ and an integer $q \in \mathbb{N} \setminus \{0\}$, referred to as a *half-size*, such that max pooling operates on a grid of size $(2q+1) \times (2q+1)$. We seek a relationship between

$$\mathrm{Y}^{\max} := U_{m,q}^{\max}[\mathrm{W}]\,(\mathrm{X}) \qquad \text{and} \qquad \mathrm{Y}^{\mathrm{mod}} := U_m^{\mathrm{mod}}[\mathrm{W}]\,(\mathrm{X}), \tag{4.84}$$

where $U_{m,q}^{\max}[\mathrm{W}]$ ($\mathbb{R}$Max) and $U_m^{\mathrm{mod}}[\mathrm{W}]$ ($\mathbb{C}$Mod) have been respectively defined in (4.1) and (4.3). As before, in what follows we omit the parameter between square brackets.

We now use the sampling results from Proposition 4.2. Let F_{X} and $\Psi_{\mathrm{W}} \in \mathcal{V}^{(s)}$ denote the functions satisfying (4.30). Recall that the continuous versions of $\mathbb{C}$Mod and $\mathbb{R}$Max operators, denoted by have been defined in (4.80) and (4.81), respectively. On the one hand, we apply (4.33) with $m \leftarrow 2m$ to $\mathrm{Y}^{\mathrm{mod}}$. For any $\boldsymbol{n} \in \mathbb{Z}^2$,

$$\begin{aligned} U_m^{\mathrm{mod}}\mathrm{X}[\boldsymbol{n}] &= (F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}})(\boldsymbol{x}_{\boldsymbol{n}}) && (4.85)\\ &= U^{\mathrm{mod}} F_{\mathrm{X}}(\boldsymbol{x}_{\boldsymbol{n}}), && (4.86) \end{aligned}$$

with $\boldsymbol{x}_{\boldsymbol{n}} := 2ms\boldsymbol{n}$. On the other hand, we postulate that

$$U_{m,q}^{\max}\mathrm{X}[\boldsymbol{n}] = U_r^{\max} F_{\mathrm{X}}(\boldsymbol{x}_{\boldsymbol{n}}) \tag{4.87}$$

for a certain value of $r \in \mathbb{R}_{>0}$. Then, (4.83) implies $\mathrm{Y}^{\mathrm{mod}} \approx \mathrm{Y}^{\max}$. However, as explained hereafter, (4.87) is not satisfied, due to the discrete nature of the max pooling grid. According to (4.1) and (4.2), we have

$$U_{m,q}^{\max}\mathrm{X}[\boldsymbol{n}] = \max_{\|\boldsymbol{p}\|_\infty \leq q} \mathrm{Re}\left((\mathrm{X} * \overline{\mathrm{W}}) \downarrow m\right)[2\boldsymbol{n} + \boldsymbol{p}]. \tag{4.88}$$

Therefore, according to (4.33) in Proposition 4.2, we get

$$U_{m,q}^{\max}\mathrm{X}[\boldsymbol{n}] = \max_{\|\boldsymbol{p}\|_\infty \leq q} \left(F_{\mathrm{X}} * \mathrm{Re}\,\overline{\Psi}_{\mathrm{W}}\right)(\boldsymbol{x}_{\boldsymbol{n}} + \boldsymbol{h}_{\boldsymbol{p}}), \tag{4.89}$$

with

$$\boldsymbol{x}_{\boldsymbol{n}} := 2ms\boldsymbol{n} \qquad \text{and} \qquad \boldsymbol{h}_{\boldsymbol{p}} := ms\boldsymbol{p}. \tag{4.90}$$

By considering $r_q := ms\left(q + \frac{1}{2}\right)$, we get a variant of (4.87) in which the maximum is evaluated on a discrete grid of $(2q+1)^2$ elements, instead of the continuous region $B_\infty(r_q)$,

as defined in (4.81) with $r \leftarrow r_q$. As a consequence, (4.82) is replaced in the discrete framework by

$$q \ll 2\pi/(m\kappa) \quad \Longrightarrow \quad U^{\max}_{m,\,q}\mathrm{X}[\boldsymbol{n}] \approx U^{\mathrm{mod}}_{m}\mathrm{X}[\boldsymbol{n}] \max_{\|\boldsymbol{p}\|_\infty \leq q} G_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p}), \tag{4.91}$$

where we have introduced, similarly to (4.76),

$$G_{\mathrm{X}} : (\boldsymbol{x}, \boldsymbol{h}) \mapsto \cos\big(\langle \boldsymbol{\nu}, \boldsymbol{h}\rangle - H_{\mathrm{X}}(\boldsymbol{x})\big), \tag{4.92}$$

with

$$\boldsymbol{\nu} := \boldsymbol{\theta}/s \qquad \text{and} \qquad H_{\mathrm{X}} := \angle\left(F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}}\right), \tag{4.93}$$

where $\angle : \mathbb{C} \to [0,\, 2\pi[$ denotes the phase operator. Unlike the continuous case, even if the window size r_q is large enough, the existence of $\boldsymbol{p} \in \{-q\,..\,q\}^2$ such that $G_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p}) = 1$ is not guaranteed, as illustrated in Figure 4.1 with $q = 1$. Instead, we can only seek a probabilistic estimation of the normalized mean squared error between $\mathrm{Y}^{\max}$ and $\mathrm{Y}^{\mathrm{mod}}$.

Approximation (4.91) implies

$$q \ll 2\pi/(m\kappa) \implies \left\|U^{\mathrm{mod}}_{m}\mathrm{X} - U^{\max}_{m,\,q}\mathrm{X}\right\|_2 \approx \left\|\delta_{m,\,q}\mathrm{X}\right\|_2, \tag{4.94}$$

where $\delta_{m,\,q}\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ is defined such that, for any $\boldsymbol{n} \in \mathbb{Z}^2$,

$$\delta_{m,\,q}\mathrm{X}[\boldsymbol{n}] := U^{\mathrm{mod}}_{m}\mathrm{X}[\boldsymbol{n}] \left(1 - \max_{\|\boldsymbol{p}\|_\infty \leq q} G_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p})\right). \tag{4.95}$$

Expression (4.94) suggests that the difference between the left and right terms can be bounded by a quantity which only depends on the product $m\kappa$ (subsampling factor $\times$ frequency localization) and the grid half-size q. In what follows, we establish a bound characterizing this approximation, which will be provided in Proposition 4.5.

For the sake of conciseness, we introduce the following notations:

$$A_{\mathrm{X}} : (\boldsymbol{x}, \boldsymbol{h}) \mapsto (F_{\mathrm{X}} * \mathrm{Re}\,\overline{\Psi}_{\mathrm{W}})(\boldsymbol{x} + \boldsymbol{h}); \tag{4.96}$$

$$\widetilde{A}_{\mathrm{X}} : (\boldsymbol{x}, \boldsymbol{h}) \mapsto \left|(F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}})(\boldsymbol{x})\right| G_{\mathrm{X}}(\boldsymbol{x}, \boldsymbol{h}). \tag{4.97}$$

We now consider, for any $\boldsymbol{n} \in \mathbb{Z}^2$, the vectors $\boldsymbol{h}^{\max}_{\boldsymbol{n}}$ and $\boldsymbol{h'}^{\max}_{\boldsymbol{n}} \in ms\,\{-q\,..\,q\}^2$ achieving the maximum value of $A_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p})$ and $\widetilde{A}_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p})$ over the max pooling grid, respectively. They satisfy

$$A^{\max}_{\mathrm{X}}(\boldsymbol{x_n}) := A_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h}^{\max}_{\boldsymbol{n}}) = \max_{\|\boldsymbol{p}\|_\infty \leq q} A_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p}); \tag{4.98}$$

$$\widetilde{A}^{\max}_{\mathrm{X}}(\boldsymbol{x_n}) := \widetilde{A}_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h'}^{\max}_{\boldsymbol{n}}) = \max_{\|\boldsymbol{p}\|_\infty \leq q} \widetilde{A}_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p}). \tag{4.99}$$

Then, according to (4.85) and (4.89), we get, for any $\boldsymbol{n} \in \mathbb{Z}^2$,

$$A^{\max}_{\mathrm{X}}(\boldsymbol{x_n}) = U^{\max}_{m,\,q}\mathrm{X}[\boldsymbol{n}]; \tag{4.100}$$

$$\widetilde{A}^{\max}_{\mathrm{X}}(\boldsymbol{x_n}) = U^{\mathrm{mod}}_{m}\mathrm{X}[\boldsymbol{n}] \max_{\|\boldsymbol{p}\|_\infty \leq q} G_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p}), \tag{4.101}$$

and (4.91) becomes

$$q \ll 2\pi/(m\kappa) \quad \Longrightarrow \quad A^{\max}_{\mathrm{X}}(\boldsymbol{x_n}) \approx \widetilde{A}^{\max}_{\mathrm{X}}(\boldsymbol{x_n}). \tag{4.102}$$

Remark 4.4. Expression (4.78) implies that $A_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p}) \approx \widetilde{A}_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p})$ for all $\boldsymbol{p} \in \{-q \,..\, q\}^2$, if $q \ll 2\pi/(m\kappa)$. However, this property does not guarantee that A_{X} and $\widetilde{A}_{\mathrm{X}}$ reach their maximum in the same exact location; *i.e.*, that $\boldsymbol{h}_{\boldsymbol{n}}^{\max} = \boldsymbol{h}'^{\max}_{\boldsymbol{n}}$.

The following lemma provides a bound for approximation (4.102).

Lemma 4.5. *For any* $\boldsymbol{x} \in \mathbb{R}^2$,

$$\left|A_{\mathrm{X}}^{\max}(\boldsymbol{x_n}) - \widetilde{A}_{\mathrm{X}}^{\max}(\boldsymbol{x_n})\right| \leq \max_{\boldsymbol{h} \in \left\{\boldsymbol{h}_{\boldsymbol{n}}^{\max}, \boldsymbol{h}'^{\max}_{\boldsymbol{n}}\right\}} \left|F_0(\boldsymbol{x_n} + \boldsymbol{h}) - F_0(\boldsymbol{x_n})\right|. \tag{4.103}$$

Proof. We apply Proposition 4.4 with $\boldsymbol{h} \leftarrow \boldsymbol{h}_{\boldsymbol{n}}^{\max}$ and $\boldsymbol{h} \leftarrow \boldsymbol{h}'^{\max}_{\boldsymbol{n}}$, respectively:

$$A_{\mathrm{X}}^{\max}(\boldsymbol{x_n}) \leq \widetilde{A}_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h}_{\boldsymbol{n}}^{\max}) + \left|F_0(\boldsymbol{x_n} + \boldsymbol{h}_{\boldsymbol{n}}^{\max}) - F_0(\boldsymbol{x_n})\right|; \tag{4.104}$$

$$\widetilde{A}_{\mathrm{X}}^{\max}(\boldsymbol{x_n}) \leq A_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h}'^{\max}_{\boldsymbol{n}}) + \left|F_0(\boldsymbol{x_n} + \boldsymbol{h}'^{\max}_{\boldsymbol{n}}) - F_0(\boldsymbol{x_n})\right|. \tag{4.105}$$

By construction, we have, for any $\boldsymbol{p} \in \{-q \,..\, q\}^2$,

$$\widetilde{A}_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p}) \leq \widetilde{A}_{\mathrm{X}}^{\max}(\boldsymbol{x_n}) \qquad \text{and} \qquad A_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p}) \leq A_{\mathrm{X}}^{\max}(\boldsymbol{x_n}). \tag{4.106}$$

Moreover, by definition, there exists $\boldsymbol{p}$ and $\boldsymbol{p}' \in \{-q \,..\, q\}^2$ such that $\boldsymbol{h}_{\boldsymbol{n}}^{\max} = \boldsymbol{h_p}$ and $\boldsymbol{h}'^{\max}_{\boldsymbol{n}} = \boldsymbol{h_{p'}}$. Therefore, (4.104) and (4.105) yield, respectively,

$$A_{\mathrm{X}}^{\max}(\boldsymbol{x_n}) \leq \widetilde{A}_{\mathrm{X}}^{\max}(\boldsymbol{x_n}) + \left|F_0(\boldsymbol{x_n} + \boldsymbol{h}_{\boldsymbol{n}}^{\max}) - F_0(\boldsymbol{x_n})\right|; \tag{4.107}$$

$$\widetilde{A}_{\mathrm{X}}^{\max}(\boldsymbol{x_n}) \leq A_{\mathrm{X}}^{\max}(\boldsymbol{x_n}) + \left|F_0(\boldsymbol{x_n} + \boldsymbol{h}'^{\max}_{\boldsymbol{n}}) - F_0(\boldsymbol{x_n})\right|, \tag{4.108}$$

which yields (4.103) and concludes the proof. □

Before stating Proposition 4.5, we consider the following hypothesis:

Hypothesis 4.1. *There exists* $\boldsymbol{h}_0 \in \mathbb{R}^2$ *with* $\|\boldsymbol{h}_0\|_2 = \sqrt{2}qms$, *such that*

$$\sum_{\boldsymbol{n} \in \mathbb{Z}^2} \max_{\boldsymbol{h} \in \left\{\boldsymbol{h}_{\boldsymbol{n}}^{\max}, \boldsymbol{h}'^{\max}_{\boldsymbol{n}}\right\}} \left|F_0(\boldsymbol{x_n} + \boldsymbol{h}) - F_0(\boldsymbol{x_n})\right|^2 \leq \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \left|F_0(\boldsymbol{x_n} + \boldsymbol{h}_0) - F_0(\boldsymbol{x_n})\right|^2. \tag{4.109}$$

The underlying idea is explained as follows. The absolute difference between $F_0(\boldsymbol{x_n}+\boldsymbol{h})$ and $F_0(\boldsymbol{x_n})$ is more likely to increase with the norm of $\boldsymbol{h}$. For any given $\boldsymbol{n} \in \mathbb{Z}^2$, we have, by construction, $\|\boldsymbol{h}_{\boldsymbol{n}}^{\max}\|_2 \leq \sqrt{2}qms$ and $\|\boldsymbol{h}'^{\max}_{\boldsymbol{n}}\|_2 \leq \sqrt{2}qms$. Therefore, we can expect to observe

$$\max_{\boldsymbol{h} \in \left\{\boldsymbol{h}_{\boldsymbol{n}}^{\max}, \boldsymbol{h}'^{\max}_{\boldsymbol{n}}\right\}} \left|F_0(\boldsymbol{x_n} + \boldsymbol{h}) - F_0(\boldsymbol{x_n})\right|^2 \leq \left|F_0(\boldsymbol{x_n} + \boldsymbol{h}_0) - F_0(\boldsymbol{x_n})\right|^2. \tag{4.110}$$

While this might occasionally not be true, Hypothesis 4.1 postulates that, when summing over all the datapoints, the inequality holds.

We now formally state the result characterizing approximation (4.94).

Proposition 4.5. *We assume that condition (4.60) is satisfied: $\kappa \leq \pi/m$. Then, under Hypothesis 4.1,*

$$\left\|U_m^{\mathrm{mod}}\mathrm{X} - U_{m,q}^{\max}\mathrm{X}\right\|_2 \leq \left\|\delta_{m,q}\mathrm{X}\right\|_2 + \beta_q(m\kappa)\left\|U_m^{\mathrm{mod}}\mathrm{X}\right\|_2, \tag{4.111}$$

where $\beta_q : \mathbb{R}_+ \to \mathbb{R}_+$ is defined by

$$\beta_q : \kappa' \mapsto q\kappa'. \tag{4.112}$$

Proof. Let us write:

$$\begin{aligned}
\left\|U_m^{\mathrm{mod}}\mathrm{X} - U_{m,q}^{\max}\mathrm{X}\right\|_2^2 &= \sum_{\boldsymbol{n}\in\mathbb{Z}^2}\left(U_m^{\mathrm{mod}}\mathrm{X}[\boldsymbol{n}] - U_{m,q}^{\max}\mathrm{X}[\boldsymbol{n}]\right)^2 \\
&= \sum_{\boldsymbol{n}\in\mathbb{Z}^2}\Bigg(U_m^{\mathrm{mod}}\mathrm{X}[\boldsymbol{n}] - U_m^{\mathrm{mod}}\mathrm{X}[\boldsymbol{n}]\max_{\|\boldsymbol{p}\|_\infty\leq q} G_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p}) \\
&\qquad\qquad + U_m^{\mathrm{mod}}\mathrm{X}[\boldsymbol{n}]\max_{\|\boldsymbol{p}\|_\infty\leq q} G_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p}) - U_{m,q}^{\max}\mathrm{X}[\boldsymbol{n}]\Bigg)^2 \\
&= \sum_{\boldsymbol{n}\in\mathbb{Z}^2}\left(\delta_{m,q}\mathrm{X}[\boldsymbol{n}] + \widetilde{A}_{\mathrm{X}}^{\max}(\boldsymbol{x_n}) - A_{\mathrm{X}}^{\max}(\boldsymbol{x_n})\right)^2,
\end{aligned}$$

according to (4.95), (4.100) and (4.101). Then, using the triangle inequality, we get

$$\left\|U_m^{\mathrm{mod}}\mathrm{X} - U_{m,q}^{\max}\mathrm{X}\right\|_2 \leq \left\|\delta_{m,q}\mathrm{X}\right\|_2 + \left(\sum_{\boldsymbol{n}\in\mathbb{Z}^2}\left(\widetilde{A}_{\mathrm{X}}^{\max}(\boldsymbol{x_n}) - A_{\mathrm{X}}^{\max}(\boldsymbol{x_n})\right)^2\right)^{1/2}. \tag{4.113}$$

Furthermore, Lemma 4.5 yields

$$\begin{aligned}
\sum_{\boldsymbol{n}\in\mathbb{Z}^2}\left(\widetilde{A}_{\mathrm{X}}^{\max}(\boldsymbol{x_n}) - A_{\mathrm{X}}^{\max}(\boldsymbol{x_n})\right)^2 &\leq \sum_{\boldsymbol{n}\in\mathbb{Z}^2}\max_{\boldsymbol{h}\in\left\{\boldsymbol{h}_{\boldsymbol{n}}^{\max}, \boldsymbol{h'}_{\boldsymbol{n}}^{\max}\right\}}\left|F_0(\boldsymbol{x_n}+\boldsymbol{h}) - F_0(\boldsymbol{x_n})\right|^2 &(4.114)\\
&\leq \sum_{\boldsymbol{n}\in\mathbb{Z}^2}\left|F_0(\boldsymbol{x_n}+\boldsymbol{h}_0) - F_0(\boldsymbol{x_n})\right|^2, &(4.115)
\end{aligned}$$

according to Hypothesis 4.1. Now, since (4.60) is satisfied, we can use Lemma 4.4 (4.51) with $\boldsymbol{h} \leftarrow \boldsymbol{h}_0$. We get

$$\begin{aligned}
\sum_{\boldsymbol{n}\in\mathbb{Z}^2}\left(\widetilde{A}_{\mathrm{X}}^{\max}(\boldsymbol{x_n}) - A_{\mathrm{X}}^{\max}(\boldsymbol{x_n})\right)^2 &\leq \frac{1}{4m^2s^2}\left\|\mathcal{T}_{\boldsymbol{h}_0}F_0 - F_0\right\|_{L^2}^2 \\
&\leq \alpha(\kappa\boldsymbol{h}_0/s)^2\,\frac{1}{4m^2s^2}\left\|F_0\right\|_{L^2}^2 \qquad \text{(acc. to Proposition 4.1)} \\
&= \alpha(\kappa\boldsymbol{h}_0/s)^2\left\|U_m^{\mathrm{mod}}\mathrm{X}\right\|_2^2. \qquad \text{(acc. to Lemma 4.4 (4.52))}
\end{aligned}$$

Since, according to Hypothesis 4.1, $\|\boldsymbol{h}_0\|_2 = \sqrt{2}qms$, it comes that $\|\boldsymbol{h}_0\|_1 = 2qms$. Therefore,

$$\alpha(\kappa\boldsymbol{h}_0/s)^2 = \frac{\kappa^2\left\|\boldsymbol{h}_0\right\|_1^2}{4s^2} = (qm\kappa)^2, \tag{4.116}$$

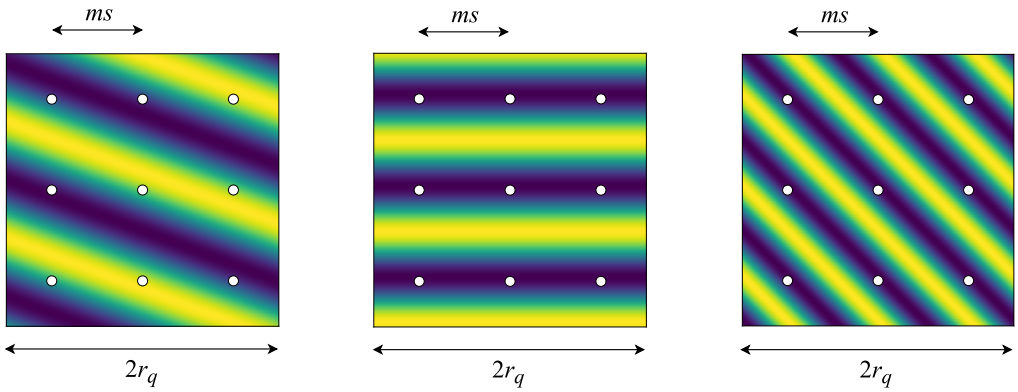


Figure 4.1. Search for the maximum value of $\boldsymbol{h} \mapsto G_{\mathrm{X}}(\boldsymbol{x}, \boldsymbol{h})$ over a discrete grid of size 3×3, *i.e.*, $q = 1$. This figure displays 3 examples with different frequencies $\boldsymbol{\nu} := \boldsymbol{\theta}/s$ and phases $H_{\mathrm{X}}(\boldsymbol{x})$. Hopefully the result will be close to the true maximum (left), but there are some pathological cases in which all points in the grid fall into pits (middle and right).

which yields

$$\sum_{\boldsymbol{n} \in \mathbb{Z}^2} \left(\widetilde{A}^{\max}_{\mathrm{X}}(\boldsymbol{x_n}) - A^{\max}_{\mathrm{X}}(\boldsymbol{x_n}) \right)^2 \leq \beta_q(m\kappa)^2 \left\| U^{\mathrm{mod}}_m \mathrm{X} \right\|_2^2. \tag{4.117}$$

Finally, plugging (4.117) into (4.113) concludes the proof. □

Remark 4.5. $\beta_q(m\kappa)$ is independent from the characteristic frequency $\boldsymbol{\theta} \in [-\pi, \pi]^2$.

We now seek a probabilistic estimation of $\left\| \delta_{m,q} \mathrm{X} \right\|_2$. For this purpose, we first reformulate the problem using the unit circle $\mathbb{S}^1 \subset \mathbb{C}$, before introducing a probabilistic framework in Section 4.3.4.

4.3.3 Notations on the Unit Circle

In what follows, for any $z \in \mathbb{C} \setminus \{0\}$, we denote by $\angle z \in [0, 2\pi[$ the argument of z. For any $z, z' \in \mathbb{S}^1$, the angle between z and z' is given by $\angle(z^* z')$. We then denote by $[z, z']_{\mathbb{S}^1} \subset \mathbb{S}^1$ the arc on the unit circle going from z to z' counterclockwise:

$$[z, z']_{\mathbb{S}^1} := \left\{ z'' \in \mathbb{S}^1 \;\middle|\; \angle(z^* z'') \leq \angle(z^* z') \right\}. \tag{4.118}$$

We remind readers that $\boldsymbol{x_n}$ and $\boldsymbol{h_p} \in \mathbb{R}^2$ have been defined in (4.90). By using the relation $\cos\alpha = \mathrm{Re}(\mathrm{e}^{i\alpha})$, (4.92) becomes, for any $\boldsymbol{n} \in \mathbb{Z}^2$ and any $\boldsymbol{p} \in \{-q \,..\, q\}^2$,

$$G_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p}) = \mathrm{Re}(Z^*_{\mathrm{X}}(\boldsymbol{x_n})\, Z_{\boldsymbol{p}}(m\boldsymbol{\theta})), \tag{4.119}$$

where we have defined the following functions with outputs on the unit circle:

$$Z_{\mathrm{X}} : \boldsymbol{x} \mapsto \mathrm{e}^{i\, H_{\mathrm{X}}(\boldsymbol{x})} \qquad \text{and} \qquad Z_{\boldsymbol{p}} : \boldsymbol{\omega} \mapsto= \mathrm{e}^{i\langle \boldsymbol{\omega}, \boldsymbol{p} \rangle}, \tag{4.120}$$

where H_{X} denotes the phase of $F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}}$ as introduced in (4.93). On the one hand, $Z_{\mathrm{X}}(\boldsymbol{x_n})$ is the phase (represented on the unit circle $\mathbb{S}^1$) of the complex wavelet transform $F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}}$ at location $\boldsymbol{x_n}$. On the other hand, $Z_{\boldsymbol{p}}(m\boldsymbol{\theta})$ approximates the phase shift between any two evaluations of $F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}}$ at locations $\boldsymbol{x}$, $\boldsymbol{x}'$ such that $\boldsymbol{x}' - \boldsymbol{x} = \boldsymbol{h_p}$. This however is

only true if we assume that Ψ_{W} exhibits slow amplitude variations. Then, $G_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p})$ approximates the cosine of the phase of $F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}}$ at location $\boldsymbol{x_n} + \boldsymbol{h_p}$.

According to (4.91), $\max_{\|\boldsymbol{p}\|_\infty \leq q} G_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p})$ approximates the ratio between ℝMax and ℂMod outputs at discrete location $\boldsymbol{n} \in \mathbb{Z}^2$. The intuition behind this is that max pooling seeks a point in a discrete grid around $\boldsymbol{x_n}$ where the phase of $F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}}$ is the closest to 1, thereby maximizing the amount of energy on the real part of the signal. Assuming slow amplitude variations of Ψ_{W}, the result therefore approximates the modulus of the complex coefficients.

To get an estimation of $\delta_{m,\,q}\mathrm{X}[\boldsymbol{n}]$ (4.95), which approximates the difference between ℂMod and ℝMax outputs at a given location $\boldsymbol{n} \in \mathbb{Z}^2$, we will exploit the following property. If $\{Z_{\boldsymbol{p}}(m\boldsymbol{\theta})\}_{\boldsymbol{p} \in \{-q..q\}^2}$ is well distributed on the unit circle, then the values of $G_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h_p})$ are evenly spread out on $[-1, 1]$. Therefore, its maximum value is more likely to be close to 1, and (4.95) becomes

$$\delta_{m,\,q}\mathrm{X}[\boldsymbol{n}] \ll U_m^{\mathrm{mod}}\mathrm{X}[\boldsymbol{n}] \qquad \forall \boldsymbol{n} \in \mathbb{Z}^2. \tag{4.121}$$

Let $n_q := (2q+1)^2$ denote the number of evaluation points for the max pooling operator. For any $\boldsymbol{\omega} \in \mathbb{R}^2$, we consider a sequence of values on $\mathbb{S}^1$, denoted by $\left(Z_i^{(q)}(\boldsymbol{\omega})\right)_{i \in \{0..n_q-1\}}$, obtained by sorting $\{Z_{\boldsymbol{p}}(\boldsymbol{\omega})\}_{\boldsymbol{p} \in \{-q..q\}^2}$ (4.120) in ascending order of their arguments:

$$0 = H_0^{(q)}(\boldsymbol{\omega}) \leq \cdots \leq H_{n_q-1}^{(q)}(\boldsymbol{\omega}) < 2\pi, \tag{4.122}$$

where $H_i^{(q)}(\boldsymbol{\omega})$ denotes the phase of $Z_i^{(q)}(\boldsymbol{\omega})$. Besides, we close the loop with $H_{n_q}^{(q)}(\boldsymbol{\omega}) := 2\pi$ and $Z_{n_q}^{(q)}(\boldsymbol{\omega}) := 1$. Then, we split $\mathbb{S}^1$ into n_q arcs delimited by $Z_i^{(q)}(\boldsymbol{\omega})$:

$$\mathfrak{A}_i^{(q)}(\boldsymbol{\omega}) := \begin{cases} \left[Z_i^{(q)}(\boldsymbol{\omega}),\, Z_{i+1}^{(q)}(\boldsymbol{\omega})\right]_{\mathbb{S}^1} & \text{if } H_{i+1}^{(q)}(\boldsymbol{\omega}) - H_i^{(q)}(\boldsymbol{\omega}) < 2\pi; \\ \mathbb{S}^1 & \text{otherwise.} \end{cases} \tag{4.123}$$

Finally, for any $i \in \{0\,..\,n_q - 1\}$, we denote by

$$\delta H_i^{(q)} : \boldsymbol{\omega} \mapsto H_{i+1}^{(q)}(\boldsymbol{\omega}) - H_i^{(q)}(\boldsymbol{\omega}) \tag{4.124}$$

the function computing the angular measure of arc $\mathfrak{A}_i^{(q)}(\boldsymbol{\omega})$, for any $\boldsymbol{\omega} \in \mathbb{R}^2$.

4.3.4 Probabilistic Framework

From now on, input X is considered as discrete 2D stochastic processes. In order to "randomize" F_{X} introduced in (4.30), we define a continuous stochastic process from X, denoted by F_{X}, such that

$$\forall \boldsymbol{x} \in \mathbb{R}^2,\ \mathsf{F}_{\mathrm{X}}(\boldsymbol{x}) := \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathrm{X}[\boldsymbol{n}]\, \boldsymbol{\Phi}_{\boldsymbol{n}}^{(s)}(\boldsymbol{x}). \tag{4.125}$$

Now, we consider the following stochastic processes, which are parameterized by X:

$$\mathsf{M}_{\mathrm{X}} := |\mathsf{F}_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}}|; \qquad \mathsf{H}_{\mathrm{X}} := \angle(\mathsf{F}_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}}); \qquad \mathsf{Z}_{\mathrm{X}} := \mathrm{e}^{i\mathsf{H}_{\mathrm{X}}}, \tag{4.126}$$

and, for any $\boldsymbol{p} \in \{-q\,..\,q\}^2$,

$$\mathsf{G}_{\mathrm{X},\,\boldsymbol{p}} := \mathrm{Re}(\mathsf{Z}_{\mathrm{X}}^*\, Z_{\boldsymbol{p}}(m\boldsymbol{\theta})); \qquad \mathsf{G}_{\mathrm{X}}^{\max} := \max_{\|\boldsymbol{p}\|_\infty \leq q} \mathsf{G}_{\mathrm{X},\,\boldsymbol{p}}, \tag{4.127}$$

where the deterministic function $Z_{\boldsymbol{p}}$ has been defined in (4.120).

Remark 4.6. $\mathsf{H}_{\mathrm{X}}(\boldsymbol{x})$ is ill-defined if $\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) = 0$. To overcome this, it is designed to follow a uniform conditional probability distribution on $[0,\, 2\pi[$, given $\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) = 0$. Moreover, we impose the following conditional independence, for any $n \in \mathbb{N} \setminus \{0\}$ and $\boldsymbol{x},\, \boldsymbol{y}_0,\, \ldots,\, \boldsymbol{y}_{n-1} \in \mathbb{R}^2$:

$$\mathsf{H}_{\mathrm{X}}(\boldsymbol{x}) \perp\!\!\!\perp \mathbf{M} \mid \mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) = 0, \qquad \text{with} \qquad \mathbf{M} := \big(\mathsf{M}_{\mathrm{X}}(\boldsymbol{y}_0),\, \cdots,\, \mathsf{M}_{\mathrm{X}}(\boldsymbol{y}_{n-1})\big)^{\top}. \tag{4.128}$$

Finally, we impose the following relationship between $\mathsf{H}_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}}$ and H_{X}, for any $\boldsymbol{u} \in \mathbb{R}^2$:

$$\mathsf{M}_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}}(\boldsymbol{x}) = 0 \implies \mathsf{H}_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}}(\boldsymbol{x}) = \mathcal{T}_{s\boldsymbol{u}}\mathsf{H}_{\mathrm{X}}(\boldsymbol{x}). \tag{4.129}$$

For any $\boldsymbol{x} \in \mathbb{R}^2$, $F_{\mathrm{X}}(\boldsymbol{x})$ (4.30) and $H_{\mathrm{X}}(\boldsymbol{x})$ (4.93) are respectively drawn from $\mathsf{F}_{\mathrm{X}}(\boldsymbol{x})$ and $\mathsf{H}_{\mathrm{X}}(\boldsymbol{x})$. Then, $Z_{\mathrm{X}}(\boldsymbol{x})$ (4.120) is a realization of $\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x})$. Consequently, according to (4.119), $G_{\mathrm{X}}(\boldsymbol{x},\, \boldsymbol{h_p})$ is a realization of $\mathsf{G}_{\mathrm{X},\,\boldsymbol{p}}(\boldsymbol{x})$. Besides, according to the definition of $\mathbb{C}$Mod in (4.3) and $\boldsymbol{x_n}$ in (4.90), Proposition 4.2 with $m \leftarrow 2m$ implies that

$$\mathsf{M}_{\mathrm{X}}(\boldsymbol{x_n}) = U_m^{\mathrm{mod}}\mathrm{X}[\boldsymbol{n}]. \tag{4.130}$$

We remind that $\boldsymbol{\theta} \in [-\pi,\, \pi]^2$ and $\kappa \in\,]0,\, 2\pi]$ respectively denote the center and size of the Fourier support of the complex kernel $\mathrm{W} \in \mathcal{J}(\boldsymbol{\theta},\, \kappa)$. To compute the expected discrepancy between $\mathrm{Y}^{\mathrm{max}}$ and $\mathrm{Y}^{\mathrm{mod}}$, we assume that

$$\|\boldsymbol{\theta}\|_2 \gg 2\pi/N; \tag{4.131}$$

$$\|\boldsymbol{\theta}\|_2 \gg \kappa, \tag{4.132}$$

where $N \in \mathbb{N} \setminus \{0\}$ denotes the support size of input images. These assumptions exclude low-frequency filters from the scope of our study. We then state the following hypotheses, for which a justification is provided in Section 4.A.

Hypothesis 4.2. *For any $\boldsymbol{x} \in \mathbb{R}^2$, $\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x})$ is uniformly distributed on $\mathbb{S}^1$.*

Hypothesis 4.3. *For any $n \in \mathbb{N} \setminus \{0\}$ and $\boldsymbol{x},\, \boldsymbol{y}_0,\, \ldots,\, \boldsymbol{y}_{n-1} \in \mathbb{R}^2$, the random variables $\mathsf{M}_{\mathrm{X}}(\boldsymbol{y}_i)$ for $i \in \{0\,..\,n-1\}$ are jointly independent of $\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x})$.*

4.3.5 Expected Quadratic Error between $\mathbb{R}$Max and $\mathbb{C}$Mod

In this section, we propose to estimate the expected value of the stochastic quadratic error $\widetilde{\mathsf{P}}_{\mathrm{X}}^2$, defined such that

$$\widetilde{\mathsf{P}}_{\mathrm{X}} := \|U_m^{\mathrm{mod}}\mathrm{X} - U_{m,\,q}^{\mathrm{max}}\mathrm{X}\|_2 / \|U_m^{\mathrm{mod}}\mathrm{X}\|_2. \tag{4.133}$$

According to (4.84), this is an estimation of the relative error between $\mathrm{Y}^{\mathrm{mod}}$ and $\mathrm{Y}^{\mathrm{max}}$.

First, let us reformulate $\delta_{m,\,q}\mathrm{X}$, introduced in (4.95), using the probabilistic framework. According to (4.119) and (4.127), we have, for any $\boldsymbol{n} \in \mathbb{Z}^2$,

$$\delta_{m,\,q}\mathrm{X}[\boldsymbol{n}] := U_m^{\mathrm{mod}}\mathrm{X}[\boldsymbol{n}]\,(1 - \mathsf{G}_{\mathrm{X}}^{\mathrm{max}}(\boldsymbol{x_n})). \tag{4.134}$$

We now consider the stochastic process

$$\mathsf{Q}_{\mathrm{X}} := 1 - \mathsf{G}_{\mathrm{X}}^{\mathrm{max}}, \tag{4.135}$$

and the random variable

$$\widetilde{\mathsf{Q}}_{\mathrm{X}} := \|\delta_{m,\,q}\mathrm{X}\|_2 \,/\, \|U_m^{\mathrm{mod}}\mathrm{X}\|_2. \tag{4.136}$$

The next steps are as follows: (1) at the pixel level, show that $\mathbb{E}[\mathsf{Q}_{\mathrm{X}}(\boldsymbol{x})^2]$ depends on the subsampling factor m and the filter frequency $\boldsymbol{\theta}$, and remains close to zero with some exceptions; (2) at the image level, show that the expected value of $\widetilde{\mathsf{Q}}_{\mathrm{X}}^2$ is equal to the latter quantity; (3) use Proposition 4.5, which states that $\widetilde{\mathsf{P}}_{\mathrm{X}} \approx \widetilde{\mathsf{Q}}_{\mathrm{X}}$, to deduce an upper bound on the expected value of $\widetilde{\mathsf{P}}_{\mathrm{X}}^2$.

The first point is established in Proposition 4.6 below, and the two remaining ones are the purpose of Theorem 4.2.

Proposition 4.6. *Assuming Hypothesis 4.2, the expected value of* $\mathsf{Q}_{\mathrm{X}}(\boldsymbol{x})^2$ *is independent from the choice of* $\boldsymbol{x} \in \mathbb{R}^2$*, and*

$$\mathbb{E}\left[\mathsf{Q}_{\mathrm{X}}(\boldsymbol{x})^2\right] = \gamma_q(m\boldsymbol{\theta})^2, \tag{4.137}$$

where we have defined

$$\gamma_q : \boldsymbol{\omega} \mapsto \sqrt{\frac{3}{2} + \frac{1}{4\pi}\sum_{i=0}^{n_q-1}\left(\sin \delta H_i^{(q)}(\boldsymbol{\omega}) - 8\sin\frac{\delta H_i^{(q)}(\boldsymbol{\omega})}{2}\right)}, \tag{4.138}$$

with $\delta H_i^{(q)}(\boldsymbol{\omega}) \in [0,\, 2\pi]$ (4.124) *being the length of arc* $\mathfrak{A}_i^{(q)}(\boldsymbol{\omega})$.

Proof. For the sake of readability, in this proof we omit the argument of functions $Z_{\boldsymbol{p}}$ (4.120), $Z_i^{(q)}$, $H_i^{(q)}$ (4.122), $\mathfrak{A}_i^{(q)}$ (4.123), and $\delta H_i^{(q)}$ (4.124); we assume they are evaluated at $\boldsymbol{\omega} \leftarrow m\boldsymbol{\theta}$. We consider the "Lebesgue" Borel σ-algebra on $\mathbb{S}^1$ generated by $\{[z,\, z']_{\mathbb{S}^1} \mid z,\, z' \in \mathbb{S}^1\} \cup \{\mathbb{S}^1\}$, on which we have defined the angular measure ϑ such that $\vartheta(\mathbb{S}^1) := 2\pi$, and

$$\forall z,\, z' \in \mathbb{S}^1,\, \vartheta\left([z,\, z']_{\mathbb{S}^1}\right) := \angle(z^* z'). \tag{4.139}$$

For any $p \in \mathbb{N} \setminus \{0\}$, we compute the p-th moment of $\mathsf{G}_{\mathrm{X}}^{\max}(\boldsymbol{x})$ defined in (4.127). By considering

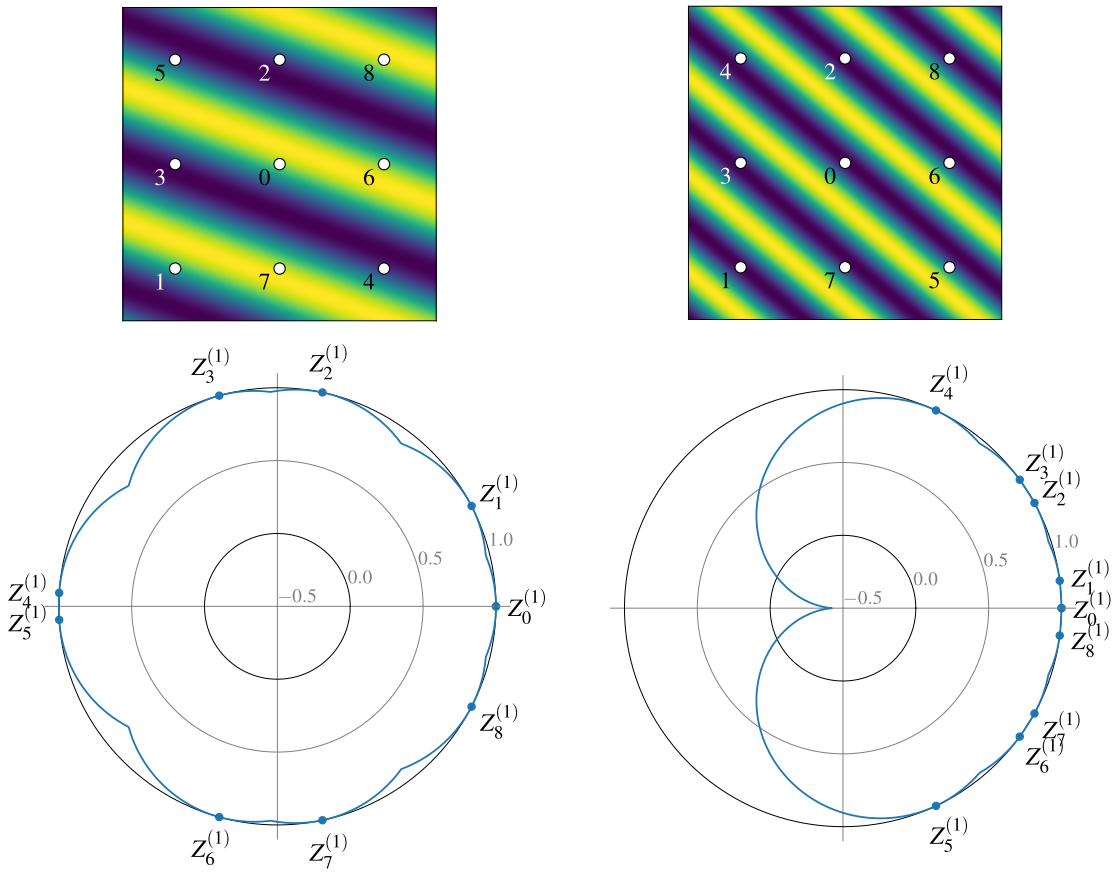
$$\begin{aligned} g_{\max} : \mathbb{S}^1 &\to [-1,\, 1] \\ z &\mapsto \max_{\|\boldsymbol{p}\|_\infty \le q} \mathrm{Re}(z^* Z_{\boldsymbol{p}}), \end{aligned} \tag{4.140}$$

we get $\mathsf{G}_{\mathrm{X}}^{\max}(\boldsymbol{x}) = g_{\max}(\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}))$. A visual representation of $g_{\max}$ is provided in Figure 4.2, for two different values of $\boldsymbol{\theta}$. According to Hypothesis 4.2, $\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x})$ follows a uniform distribution on $\mathbb{S}^1$. Therefore,

$$\mathbb{E}\left[\mathsf{G}_{\mathrm{X}}^{\max}(\boldsymbol{x})^p\right] = \frac{1}{2\pi}\int_{\mathbb{S}^1} g_{\max}(z)^p \,\mathrm{d}\vartheta(z), \tag{4.141}$$

which proves that $\mathbb{E}\left[\mathsf{G}_{\mathrm{X}}^{\max}(\boldsymbol{x})^p\right]$ does not depend on $\boldsymbol{x}$. Let us split the unit circle $\mathbb{S}^1$ into the arcs $\mathfrak{A}_0^{(q)}, \ldots, \mathfrak{A}_{n_q-1}^{(q)}$ such as introduced in (4.123):

$$\mathbb{E}\left[\mathsf{G}_{\mathrm{X}}^{\max}(\boldsymbol{x})^p\right] = \frac{1}{2\pi}\sum_{i=0}^{n_q-1}\int_{\mathfrak{A}_i^{(q)}} g_{\max}(z)^p \,\mathrm{d}\vartheta(z). \tag{4.142}$$



(a) General case

(b) Pathological case

Figure 4.2. Top: 2D representation of $\boldsymbol{h} \mapsto G_{\mathrm{X}}(\boldsymbol{x_n}, \boldsymbol{h})$ (4.92), for two different values of $\boldsymbol{\theta} \in \mathbb{R}^2$, $q = 1$ and arbitrary values of $m \in \mathbb{N} \setminus \{0\}$ and $s \in \mathbb{R} \setminus \{0\}$. Assuming the plots are centered around $\boldsymbol{h} = \boldsymbol{0}$, each point materializes a location $\boldsymbol{h_p}$ in the max pooling grid, for $\boldsymbol{p} \in \{-q \,..\, q\}^2$. The desirable situation occurs when one of these locations falls near a ridge (bright areas), in which case the outputs produced by $\mathbb{R}$Max and $\mathbb{C}$Mod are similar—see (4.91). Each number $i \in \{0 \,..\, 8\}$ represents the rank of $Z_{\boldsymbol{p}} \in \mathbb{S}^1$ (4.120), when these values are sorted by ascending order of their arguments (4.122). If rank i is affected to location $\boldsymbol{h_p}$, then we have $Z_{\boldsymbol{p}} = Z_i^{(q)}$. Bottom: polar representations of $g_{\max} : \mathbb{S}^1 \to [-1, 1]$ (4.140), corresponding to the same settings. The closer the curve is from the outer ring, the more likely some points $\boldsymbol{h_p}$ will fall near a ridge of G_{X}. (a) Case where the values $Z_{\boldsymbol{p}}$ are roughly evenly distributed on $\mathbb{S}^1$. (b) Case where these values are concentrated in a small portion of the unit circle. The most extreme cases occurs when $Z_{\boldsymbol{p}} = 1$ for any $\boldsymbol{p}$. Figure 4.1 (middle and right) depicts two such situations.

Let $i \in \{0\,..\,n_q - 1\}$. We show that

$$\forall z \in \mathfrak{A}_i^{(q)},\ g_{\max}(z) = \max\left(\mathrm{Re}(z^* Z_i^{(q)}),\ \mathrm{Re}(z^* Z_{i+1}^{(q)})\right). \tag{4.143}$$

Let $z \in \mathfrak{A}_i^{(q)}$ and $i' \notin \{i,\, i+1\}$. We prove that

$$\mathrm{Re}(z^* Z_{i'}^{(q)}) \le \mathrm{Re}(z^* Z_i^{(q)}) \qquad \text{or} \qquad \mathrm{Re}(z^* Z_{i'}^{(q)}) \le \mathrm{Re}(z^* Z_{i+1}^{(q)}). \tag{4.144}$$

On the one hand, we assume that $\angle(z^* Z_{i'}^{(q)}) \le \pi$. By design of $(Z_i^{(q)})_{i\in\{0..n_q-1\}}$, we have

$$Z_{i+1}^{(q)} \in [z,\, Z_{i'}^{(q)}]_{\mathbb{S}^1}. \tag{4.145}$$

Therefore, by definition of arcs on the unit circle (4.118), we get

$$\angle(z^* Z_{i+1}^{(q)}) \le \angle(z^* Z_{i'}^{(q)}). \tag{4.146}$$

Then, since cos is non-increasing on $[0,\, \pi]$, we get

$$\cos\angle(z^* Z_{i+1}^{(q)}) \ge \cos\angle(z^* Z_{i'}^{(q)}), \tag{4.147}$$

which yields the right part of (4.144). On the other hand, if $\angle(z^* Z_{i'}^{(q)}) \ge \pi$, a similar reasoning yields the left part of (4.144). Then, (4.143) holds.

Now, we show that, as observed in Figure 4.2, $g_{\max}$ is piecewise-symmetric with respect to the center value of each arc $\mathfrak{A}_i^{(q)}$, denoted by

$$\overline{Z}_i^{(q)} := \sqrt{Z_i^{(q)} Z_{i+1}^{(q)}}. \tag{4.148}$$

Let $z_1,\, z_2 \in \mathfrak{A}_i^{(q)}$ which are symmetric with respect to $\overline{Z}_i^{(q)}$. Therefore, there exists $z' \in \mathbb{S}^1$ such that $z_1 = \overline{Z}_i^{(q)} z'$ and $z_2 = \overline{Z}_i^{(q)} z'^*$. We now prove that

$$g_{\max}(z_1) = g_{\max}(z_2). \tag{4.149}$$

A simple calculation yields

$$z_1^* Z_{i+1}^{(q)} = z'^* \widetilde{Z}_i^{(q)} \qquad \text{and} \qquad z_2^* Z_i^{(q)} = (z'^* \widetilde{Z}_i^{(q)})^*, \tag{4.150}$$

with

$$\widetilde{Z}_i^{(q)} := (Z_i^{(q)^*}\, \overline{Z}_i^{(q)}) = (\overline{Z}_i^{(q)^*}\, Z_{i+1}^{(q)}). \tag{4.151}$$

Therefore,

$$\mathrm{Re}(z_1^* Z_{i+1}^{(q)}) = \mathrm{Re}(z_2^* Z_i^{(q)}). \tag{4.152}$$

Since $z_1,\, z_2$ both belong to $\mathfrak{A}_i^{(q)}$, $g_{\max}(z_1)$ and $g_{\max}(z_2)$ satisfy (4.143). Then, by symmetry, (4.152) implies (4.149). One can observe from Figure 4.2 that $g_{\max}$ reaches its local minimum at the center of arc $\mathfrak{A}_i^{(q)}$, *i.e.*, $\overline{Z}_i^{(q)}$. This corresponds to a point where $g_{\max}$ is non-differentiable.

We denote by $\overline{\mathfrak{A}}_i^{(q)} := [Z_i^{(q)},\, \overline{Z}_i^{(q)}]_{\mathbb{S}^1}$ the first half of arc $\mathfrak{A}_i^{(q)}$. Then,

$$\forall z \in \overline{\mathfrak{A}}_i^{(q)},\ g_{\max}(z) = \mathrm{Re}(z^* Z_i^{(q)}). \tag{4.153}$$

As a consequence, using symmetry, we get

$$\begin{aligned}\int_{\mathfrak{A}_i^{(q)}} g_{\max}(z)^p \,\mathrm{d}\vartheta(z) &= 2\int_{\overline{\mathfrak{A}}_i^{(q)}} g_{\max}(z)^p \,\mathrm{d}\vartheta(z) \\ &= 2\int_{\overline{\mathfrak{A}}_i^{(q)}} \mathrm{Re}(z^* Z_i^{(q)})^p \,\mathrm{d}\vartheta(z).\end{aligned}$$

By using the change of variable formula (Athreya and Lahiri, 2006, p. 81) with $z \leftarrow \mathrm{e}^{i\eta}$, we get

$$\int_{\mathfrak{A}_i^{(q)}} g_{\max}(z)^p \,\mathrm{d}\vartheta(z) = 2\int_{H_i^{(q)}}^{\overline{H}_i^{(q)}} \cos^p(\eta - H_i^{(q)})\,\mathrm{d}\eta, \tag{4.154}$$

where $\overline{H}_i^{(q)} := (H_i^{(q)} + H_{i+1}^{(q)})/2$ denotes the argument of $\overline{Z}_i^{(q)}$. Then, the change of variable $\eta' \leftarrow \eta - H_i^{(q)}$ yields

$$\int_{\mathfrak{A}_i^{(q)}} g_{\max}(z)^p \,\mathrm{d}\vartheta(z) = 2\int_0^{\delta H_i^{(q)}/2} \cos^p \eta' \,\mathrm{d}\eta'. \tag{4.155}$$

Now, we insert (4.155) into (4.142), and compute $\mathbb{E}\left[\mathsf{G}_{\mathrm{X}}^{\max}(\boldsymbol{x})^p\right]$ for $p \leftarrow 1$ and $p \leftarrow 2$:

$$\mathbb{E}\left[\mathsf{G}_{\mathrm{X}}^{\max}(\boldsymbol{x})\right] = \frac{1}{\pi}\sum_{i=0}^{n_q-1} \sin\frac{\delta H_i^{(q)}}{2};$$

$$\mathbb{E}\left[\mathsf{G}_{\mathrm{X}}^{\max}(\boldsymbol{x})^2\right] = \frac{1}{2} + \frac{1}{4\pi}\sum_{i=0}^{n_q-1} \sin\delta H_i^{(q)}.$$

We recall that $\mathsf{Q}_{\mathrm{X}} := 1 - \mathsf{G}_{\mathrm{X}}^{\max}$. By linearity of $\mathbb{E}$, we get

$$\mathbb{E}\left[\mathsf{Q}_{\mathrm{X}}(\boldsymbol{x})^2\right] := \frac{3}{2} + \frac{1}{4\pi}\sum_{i=0}^{n_q-1}\left(\sin\delta H_i^{(q)} - 8\sin\frac{\delta H_i^{(q)}}{2}\right), \tag{4.156}$$

which concludes the proof. □

We consider an ideal scenario where $\left(Z_i^{(q)}(m\boldsymbol{\theta})\right)_{i\in\{0..n_q-1\}}$ are evenly spaced on $\mathbb{S}^1$. Then, an order 2 Taylor expansion yields

$$\gamma_q(m\boldsymbol{\theta}) = o(1/q^2), \tag{4.157}$$

providing an order-two-polynomial decay rate for $\mathsf{Q}_{\mathrm{X}}(\boldsymbol{x})$, when the grid half-size q increases. Figure 4.3 displays $\boldsymbol{\theta} \mapsto \gamma_q(m\boldsymbol{\theta})^2$ for $\boldsymbol{\theta} \in [-\pi, \pi]^2$, with $m = 4$ and $q = 1$ as in AlexNet. We notice that, for the major part of the Fourier domain, γ_q remains close to 0. However, we observe a regular pattern of dark regions, which correspond to pathological frequencies where the repartition of $\left(Z_i^{(q)}(m\boldsymbol{\theta})\right)_{i\in\{0..n_q-1\}}$ is unbalanced.

So far, we established a result at the pixel level. Before stating Theorem 4.2, which extends the result to the image level, we need the following intermediate statement.

Proposition 4.7. *We consider the random variable*

$$\widetilde{\mathsf{S}}_{\mathrm{X}} := \left\|U_m^{\mathrm{mod}}\mathrm{X}\right\|_2. \tag{4.158}$$

Under Hypothesis 4.3, for any $\boldsymbol{x} \in \mathbb{R}^2$,

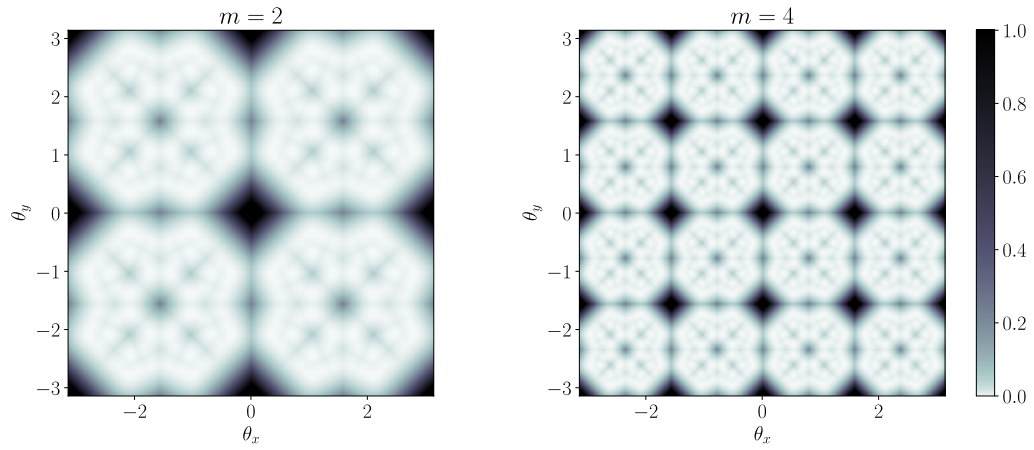


Figure 4.3. $\gamma(m\boldsymbol{\theta})^2$ as a function of the kernel characteristic frequency $\boldsymbol{\theta} \in [-\pi, \pi]^2$. According to Theorem 4.2, this quantity provides an approximate bound for the expected quadratic error between ℝMax and ℂMod outputs. The subsampling factor m has been set to 2 as in ResNet (left), and 4 as in AlexNet (right). The bright regions correspond to frequencies for which the two outputs are expected to be similar. However, in the dark regions, pathological cases such as illustrated in Figure 4.1 are more likely to occur.

- $\mathsf{Z}_\mathrm{X}(\boldsymbol{x})$ *is independent of* $\widetilde{\mathsf{S}}_\mathrm{X}$*;*
- $\mathsf{Z}_\mathrm{X}(\boldsymbol{x})$*,* $\mathsf{M}_\mathrm{X}(\boldsymbol{x})$ *are conditionally independent given* $\widetilde{\mathsf{S}}_\mathrm{X}$*.*

Proof. We suppose that Hypothesis 4.3 is satisfied and we consider $\boldsymbol{x} \in \mathbb{R}^2$. For a given $n \in \mathbb{N} \setminus \{0\}$, we introduce the random variable

$$\widetilde{\mathsf{S}}_{\mathrm{X},\,n} := \sqrt{\sum_{\|\boldsymbol{p}\|_\infty \leq n} \mathsf{M}_\mathrm{X}(\boldsymbol{x}_{\boldsymbol{p}})^2}. \tag{4.159}$$

According to Hypothesis 4.3, $\mathsf{Z}_\mathrm{X}(\boldsymbol{x})$ is jointly independent of $\mathsf{M}_\mathrm{X}(\boldsymbol{x}_{\boldsymbol{p}})$ for $\boldsymbol{p} \in \{-n \,..\, n\}^2$. Therefore, by composition, $\mathsf{Z}_\mathrm{X}(\boldsymbol{x})$ is also independent of $\widetilde{\mathsf{S}}_{\mathrm{X},\,n}$. Moreover, according to (4.130) and (4.158), $\widetilde{\mathsf{S}}_{\mathrm{X},\,n}$ converges almost surely towards $\widetilde{\mathsf{S}}_\mathrm{X}$, which proves independence between $\mathsf{Z}_\mathrm{X}(\boldsymbol{x})$ and $\widetilde{\mathsf{S}}_\mathrm{X}$.

Now, we prove conditional independence between $\mathsf{Z}_\mathrm{X}(\boldsymbol{x})$ and $\mathsf{M}_\mathrm{X}(\boldsymbol{x})$ given $\widetilde{\mathsf{S}}_\mathrm{X}$. According to Hypothesis 4.3,

$$\left(\mathsf{M}_\mathrm{X}(\boldsymbol{x}),\, \widetilde{\mathsf{S}}_{\mathrm{X},\,n}\right) \perp\!\!\!\perp \mathsf{Z}_\mathrm{X}(\boldsymbol{x}), \tag{4.160}$$

where $\perp\!\!\!\perp$ stands for independence. This is because $\widetilde{\mathsf{S}}_{\mathrm{X},\,n}$ only depends on a finite number of $\mathsf{M}_\mathrm{X}(\boldsymbol{x}_{\boldsymbol{p}})$. Therefore,

$$\mathsf{Z}_\mathrm{X}(\boldsymbol{x}) \perp\!\!\!\perp \mathsf{M}_\mathrm{X}(\boldsymbol{x}) \,\big|\, \widetilde{\mathsf{S}}_{\mathrm{X},\,n}. \tag{4.161}$$

Finally, since $\widetilde{\mathsf{S}}_{\mathrm{X},\,n}$ converges almost surely towards $\widetilde{\mathsf{S}}_\mathrm{X}$, it comes that $\mathsf{Z}_\mathrm{X}(\boldsymbol{x})$ and $\mathsf{M}_\mathrm{X}(\boldsymbol{x})$ are conditionally independent given $\widetilde{\mathsf{S}}_\mathrm{X}$. □

Finally, Propositions 4.6 and 4.7 yield the following theorem. It provides an upper bound on the expected value of the normalized mean squared error $\widetilde{\mathsf{P}}^2_\mathrm{X}$, such as defined in (4.133).

Theorem 4.2 (Discrepancy between $\mathbb{C}$Mod and $\mathbb{R}$Max)**.** *Let* $\mathrm{W} \in \mathcal{J}(\boldsymbol{\theta}, \kappa)$ *denote a discrete Gabor-like filter,* $m \in \mathbb{N} \setminus \{0\}$ *a subsampling factor and* $q \in \mathbb{N} \setminus \{0\}$ *a grid half-size. We consider a stochastic process* X *whose realizations are elements of* $l^2_{\mathbb{R}}(\mathbb{Z}^2)$*. We assume that condition* (4.60) *is satisfied:* $\kappa \leq \pi/m$*. Then, under Hypotheses 4.1 to 4.3,*[3]

$$\mathbb{E}[\widetilde{\mathsf{P}}^2_{\mathrm{X}}] \leq \left(\beta_q(m\kappa) + \gamma_q(m\boldsymbol{\theta})\right)^2, \tag{4.162}$$

where $\widetilde{\mathsf{P}}^2_{\mathrm{X}}$ (4.133) *denotes the stochastic quadratic error between* $\mathbb{C}$*Mod and* $\mathbb{R}$*Max outputs. We remind that* β_q *and* γ_q *have been introduced in* (4.112) *and* (4.138)*, respectively.*

Proof. We consider $\boldsymbol{n} \in \mathbb{Z}^2$. By construction, $\mathsf{Q}_{\mathrm{X}}(\boldsymbol{x_n}) := 1 - \mathsf{G}^{\max}_{\mathrm{X}}(\boldsymbol{x_n})$ only depends on $\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x_n})$. Therefore, under Hypothesis 4.3, Proposition 4.7 implies

$$\mathsf{Q}_{\mathrm{X}}(\boldsymbol{x_n}) \perp\!\!\!\perp \mathsf{M}_{\mathrm{X}}(\boldsymbol{x_n}) \mid \widetilde{\mathsf{S}}^2_{\mathrm{X}} \qquad \text{and} \qquad \mathsf{Q}_{\mathrm{X}}(\boldsymbol{x_n}) \perp\!\!\!\perp \widetilde{\mathsf{S}}^2_{\mathrm{X}}. \tag{4.163}$$

Besides, we introduce

$$\widetilde{\Delta}_{\mathrm{X}} := \|\delta_{m,\,q}\mathrm{X}\|_2\,, \tag{4.164}$$

where $\delta_{m,\,q}\mathrm{X}$ is defined in (4.134). Then, using the linearity of $\mathbb{E}$, we get

$$\begin{aligned}
\mathbb{E}\Big[\widetilde{\Delta}^2_{\mathrm{X}} \;\Big|\; \widetilde{\mathsf{S}}^2_{\mathrm{X}} = \sigma\Big] &= \sum_{\boldsymbol{n}\in\mathbb{Z}^2} \mathbb{E}\left[\delta_{m,\,q}[\boldsymbol{n}]^2 \;\middle|\; \widetilde{\mathsf{S}}^2_{\mathrm{X}} = \sigma\right] \\
&= \sum_{\boldsymbol{n}\in\mathbb{Z}^2} \mathbb{E}\left[U^{\mathrm{mod}}_{m,\,l}\mathrm{X}[\boldsymbol{n}]^2 \left(1 - \mathsf{G}^{\max}_{\mathrm{X}}(\boldsymbol{x_n})\right)^2 \;\middle|\; \widetilde{\mathsf{S}}^2_{\mathrm{X}} = \sigma\right] \\
&= \sum_{\boldsymbol{n}\in\mathbb{Z}^2} \mathbb{E}\left[\mathsf{M}_{\mathrm{X}}(\boldsymbol{x_n})^2\, \mathsf{Q}_{\mathrm{X}}(\boldsymbol{x_n})^2 \;\middle|\; \widetilde{\mathsf{S}}^2_{\mathrm{X}} = \sigma\right] \quad \text{(acc. to (4.130) and (4.135))} \\
&= \sum_{\boldsymbol{n}\in\mathbb{Z}^2} \mathbb{E}\left[\mathsf{M}_{\mathrm{X}}(\boldsymbol{x_n})^2 \;\middle|\; \widetilde{\mathsf{S}}^2_{\mathrm{X}} = \sigma\right] \mathbb{E}[\mathsf{Q}_{\mathrm{X}}(\boldsymbol{x_n})^2] \quad \text{(acc. to (4.163))}.
\end{aligned}$$

According to (4.130) and (4.158), we have

$$\sum_{\boldsymbol{n}\in\mathbb{Z}^2} \mathsf{M}_{\mathrm{X}}(\boldsymbol{x_n})^2 = \|U^{\mathrm{mod}}_m \mathrm{X}\|_2^2 = \widetilde{\mathsf{S}}^2_{\mathrm{X}}. \tag{4.165}$$

Therefore, using again the linearity of $\mathbb{E}$, we get

$$\begin{aligned}
\mathbb{E}\Big[\widetilde{\Delta}^2_{\mathrm{X}} \;\Big|\; \widetilde{\mathsf{S}}^2_{\mathrm{X}} = \sigma\Big] &= \mathbb{E}\left[\widetilde{\mathsf{S}}^2_{\mathrm{X}} \;\middle|\; \widetilde{\mathsf{S}}^2_{\mathrm{X}} = \sigma\right] \mathbb{E}[\mathsf{Q}_{\mathrm{X}}(\boldsymbol{x_n})^2] \\
&= \sigma \cdot \mathbb{E}[\mathsf{Q}_{\mathrm{X}}(\boldsymbol{x_n})^2].
\end{aligned}$$

Under Hypothesis 4.2, Proposition 4.6 yields

$$\mathbb{E}\Big[\widetilde{\Delta}^2_{\mathrm{X}} \;\Big|\; \widetilde{\mathsf{S}}^2_{\mathrm{X}} = \sigma\Big] = \sigma \cdot \gamma_q(m\boldsymbol{\theta})^2. \tag{4.166}$$

Besides, we can reformulate $\widetilde{\mathsf{Q}}_{\mathrm{X}}$ such as defined in (4.136): $\widetilde{\mathsf{Q}}_{\mathrm{X}} = \widetilde{\Delta}_{\mathrm{X}}/\widetilde{\mathsf{S}}_{\mathrm{X}}$. Therefore,

$$\mathbb{E}\Big[\widetilde{\mathsf{Q}}^2_{\mathrm{X}} \;\Big|\; \widetilde{\mathsf{S}}^2_{\mathrm{X}} = \sigma\Big] = \frac{1}{\sigma}\,\mathbb{E}\Big[\widetilde{\Delta}^2_{\mathrm{X}} \;\Big|\; \widetilde{\mathsf{S}}^2_{\mathrm{X}} = \sigma\Big] = \gamma_q(m\boldsymbol{\theta})^2. \tag{4.167}$$

[3] We can easily prove that these properties are independent from the choice of sampling interval $s > 0$.

According to (4.167), the conditional expected value of $\widetilde{\mathsf{Q}}_{\mathrm{X}}^2$ remains the same whatever the outcome of $\widetilde{\mathsf{S}}_{\mathrm{X}}^2$. Thus, the law of total expectation states that

$$\mathbb{E}[\widetilde{\mathsf{Q}}_{\mathrm{X}}^2] = \mathbb{E}\left[\mathbb{E}[\widetilde{\mathsf{Q}}_{\mathrm{X}}^2 \mid \widetilde{\mathsf{S}}_{\mathrm{X}}^2]\right] = \gamma_q(m\boldsymbol{\theta})^2. \tag{4.168}$$

Since we have assumed Hypothesis 4.1, we can apply Proposition 4.5. Using the definition of $\widetilde{\mathsf{P}}_{\mathrm{X}}$ (4.133) and $\widetilde{\mathsf{Q}}_{\mathrm{X}}$ (4.136), we get

$$\widetilde{\mathsf{P}}_{\mathrm{X}} \leq \widetilde{\mathsf{Q}}_{\mathrm{X}} + \beta_q(m\kappa). \tag{4.169}$$

Then,

$$\mathbb{E}[\widetilde{\mathsf{P}}_{\mathrm{X}}^2] \leq \mathbb{E}[\widetilde{\mathsf{Q}}_{\mathrm{X}}^2] + 2\beta_q(m\kappa)\,\mathbb{E}[\widetilde{\mathsf{Q}}_{\mathrm{X}}] + \beta_q(m\kappa)^2. \tag{4.170}$$

According to Jensen's inequality,

$$\mathbb{E}[\widetilde{\mathsf{Q}}_{\mathrm{X}}] \leq \sqrt{\mathbb{E}[\widetilde{\mathsf{Q}}_{\mathrm{X}}^2]} = \gamma_q(m\boldsymbol{\theta}). \tag{4.171}$$

Thus,

$$\mathbb{E}[\widetilde{\mathsf{P}}_{\mathrm{X}}^2] \leq \gamma_q(m\boldsymbol{\theta})^2 + 2\beta_q(m\kappa)\gamma_q(m\boldsymbol{\theta}) + \beta_q(m\kappa)^2, \tag{4.172}$$

which yields (4.162). □

Let us analyze the bound obtained in (4.162). The first term, $\beta_q(m\kappa)$, accounts for the localized property of the convolution filter W. Assumably, it decreases linearly with the product $m\kappa$. In the limit case where $\kappa = 0$ (infinite, nonlocal filter), we get $\beta_q(m\kappa) = 0$. Note that a smaller subsampling factor m allows for a larger bandwidth κ. Besides, $\beta_q(m\kappa)$ increases with the size of the max pooling grid, which is characterized by q. The second term, $\gamma_q(m\boldsymbol{\theta})$, accounts for the discrete nature of the max pooling grid. It strongly depends on the characteristic frequency $\boldsymbol{\theta}$, as illustrated in Figure 4.3. According to (4.157), this term has a polynomial decay when q increases. However, increasing the size of the max pooling grid also results in increasing the term $\beta_q(m\kappa)$, as explained above. Therefore, a tradeoff must be found to get an optimal bound.

4.4 Shift Invariance of ℝMax Outputs

In this section, we present the main theoretical claim of this chapter. Based on the previous results, we provide a probabilistic measure of shift invariance for ℝMax operators. First, we consider the following lemma.

Lemma 4.6. *If Hypotheses 4.2 and 4.3 are satisfied, then they are also true with* $\mathrm{X} \leftarrow \mathcal{T}_{\boldsymbol{u}}\mathrm{X}$*, for any* $\boldsymbol{u} \in \mathbb{R}^2$.

Proof. First, we show that, for any $\boldsymbol{x} \in \mathbb{R}^2$,

$$\mathsf{M}_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}}(\boldsymbol{x}) = \mathcal{T}_{s\boldsymbol{u}}\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}); \tag{4.173}$$

$$\mathsf{Z}_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}}(\boldsymbol{x}) = \mathcal{T}_{s\boldsymbol{u}}\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}). \tag{4.174}$$

According to Lemma 4.3, and since the convolution product commutes with translations, we have

$$(F_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}} * \overline{\Psi}_{\mathrm{W}})(\boldsymbol{x}) = \mathcal{T}_{s\boldsymbol{u}}(F_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}})(\boldsymbol{x}). \tag{4.175}$$

Then, using (4.126), the above expression becomes

$$\mathsf{M}_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}}(\boldsymbol{x}) \times \mathsf{Z}_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}}(\boldsymbol{x}) = (\mathcal{T}_{s\boldsymbol{u}}\mathsf{M}_{\mathrm{X}})(\boldsymbol{x}) \times (\mathcal{T}_{s\boldsymbol{u}}\mathsf{Z}_{\mathrm{X}})(\boldsymbol{x}). \tag{4.176}$$

Therefore, we necessarily have (4.173). On the one hand, if $\mathsf{M}_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}}(\boldsymbol{x}) > 0$, then (4.174) is satisfied, by uniqueness of the magnitude-phase decomposition. On the other hand, if $\mathsf{M}_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}}(\boldsymbol{x}) = 0$, then (4.129) also guarantees (4.174), by design.

Finally, we remind that

$$\mathcal{T}_{s\boldsymbol{u}}\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) = \mathsf{M}_{\mathrm{X}}(\boldsymbol{x} - s\boldsymbol{u}) \qquad \text{and} \qquad \mathcal{T}_{s\boldsymbol{u}}\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) = \mathsf{Z}_{\mathrm{X}}(\boldsymbol{x} - s\boldsymbol{u}). \tag{4.177}$$

Then, considering hypotheses Hypotheses 4.2 and 4.3 with $\boldsymbol{x} \leftarrow \boldsymbol{x} - s\boldsymbol{u}$ concludes the proof. □

We are now ready to state the main result about shift invariance of $\mathbb{R}$Max outputs.

Theorem 4.3 (Shift invariance of $\mathbb{R}$Max)**.** *We assume that the requirements stated in Theorem 4.2 are satisfied. Besides, given a translation vector $\boldsymbol{u} \in \mathbb{R}^2$, we consider the following random variable:*

$$\widetilde{\mathsf{R}}_{\mathrm{X},\,\boldsymbol{u}} := \left\|U_{m,\,q}^{\max}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X}) - U_{m,\,q}^{\max}\mathrm{X}\right\|_2 / \left\|U_m^{\mathrm{mod}}\mathrm{X}\right\|_2. \tag{4.178}$$

Then, under condition (4.60), we have

$$\mathbb{E}[\widetilde{\mathsf{R}}_{\mathrm{X},\,\boldsymbol{u}}] \leq 2\left(\beta_q(m\kappa) + \gamma_q(m\boldsymbol{\theta})\right) + \alpha(\kappa\boldsymbol{u}), \tag{4.179}$$

where α, β_q and γ_q are defined in (4.21), (4.112) and (4.138), respectively.

Proof. Using the triangle inequality, we compute

$$\begin{aligned}\left\|U_{m,\,q}^{\max}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X}) - U_{m,\,q}^{\max}\mathrm{X}\right\|_2 \\ \leq \left\|U_m^{\mathrm{mod}}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X})\right\|_2 \widetilde{\mathsf{P}}_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}} + \left\|U_m^{\mathrm{mod}}\mathrm{X}\right\|_2 \widetilde{\mathsf{P}}_{\mathrm{X}} + \left\|U_m^{\mathrm{mod}}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X}) - U_m^{\mathrm{mod}}\mathrm{X}\right\|_2,\end{aligned} \tag{4.180}$$

where $\widetilde{\mathsf{P}}_{\mathrm{X}}$ and $\widetilde{\mathsf{P}}_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}}$ are defined in (4.133). According to (4.60), we can apply Proposition 4.3 on the first term of (4.180):

$$\left\|U_m^{\mathrm{mod}}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X})\right\|_2 = \left\|U_m^{\mathrm{mod}}\mathrm{X}\right\|_2. \tag{4.181}$$

Moreover, we can apply Theorem 4.1 to the third term of (4.180):

$$\left\|U_m^{\mathrm{mod}}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X}) - U_m^{\mathrm{mod}}\mathrm{X}\right\|_2 \leq \alpha(\kappa\boldsymbol{u})\left\|U_m^{\mathrm{mod}}\mathrm{X}\right\|_2. \tag{4.182}$$

We therefore get

$$\left\|U_{m,\,q}^{\max}(\mathcal{T}_{\boldsymbol{u}}\mathrm{X}) - U_{m,\,q}^{\max}\mathrm{X}\right\|_2 \leq \left[\widetilde{\mathsf{P}}_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}} + \widetilde{\mathsf{P}}_{\mathrm{X}} + \alpha(\kappa\boldsymbol{u})\right]\left\|U_m^{\mathrm{mod}}\mathrm{X}\right\|_2. \tag{4.183}$$

Then, by linearity of $\mathbb{E}$, we get

$$\mathbb{E}[\widetilde{\mathsf{R}}_{\mathrm{X},\,\boldsymbol{u}}] \leq \mathbb{E}[\widetilde{\mathsf{P}}_{\mathcal{T}_{\boldsymbol{u}}\mathrm{X}}] + \mathbb{E}[\widetilde{\mathsf{P}}_{\mathrm{X}}] + \alpha(\kappa\boldsymbol{u}), \tag{4.184}$$

where $\widetilde{\mathsf{R}}_{\mathrm{X},\,\boldsymbol{u}}$ has been introduced in (4.179).

For any stochastic process X' satisfying Hypotheses 4.2 and 4.3, Theorem 4.2 and Jensen's inequality yield:

$$\mathbb{E}[\widetilde{\mathsf{P}}_{\mathrm{X}'}] \leq \beta_q(m\kappa) + \gamma_q(m\boldsymbol{\theta}). \tag{4.185}$$

According to Lemma 4.6, Hypotheses 4.2 and 4.3 are also satisfied for $\mathrm{X} \leftarrow \mathcal{T}_{\boldsymbol{u}}\mathrm{X}$. Therefore, (4.185) is valid for both $\mathrm{X}' \leftarrow \mathrm{X}$ and $\mathrm{X}' \leftarrow \mathcal{T}_{\boldsymbol{u}}\mathrm{X}$, and plugging it into (4.184) concludes the proof. □

In the bound established in (4.179), the sum $\beta_q(m\kappa) + \gamma_q(m\boldsymbol{\theta})$ accounts for the discrepancy between $\mathbb{R}$Max and $\mathbb{C}$Mod outputs, as stated in Theorem 4.2, whereas the term $\alpha(\kappa \boldsymbol{u})$ characterizes the stability of $\mathbb{C}$Mod outputs, as stated in Theorem 4.1. If κ is sufficiently small, then $\alpha(\kappa \boldsymbol{u})$ and $\beta_q(m\kappa)$ become negligible with respect to $\gamma_q(m\boldsymbol{\theta})$, and the bound can be approximated by $2\,\gamma_q(m\boldsymbol{\theta})$. Theorem 4.3 therefore provides a validity domain for shift invariance of $\mathbb{R}$Max operators, as illustrated in Figure 4.3 with $q = 1$.

Remark 4.7. The stochastic discrepancy introduced in (4.178) is estimated relatively to the $\mathbb{C}$Mod output. This choice is motivated by the perfect shift invariance of its norm, as shown in Proposition 4.3.

Remark 4.8. In practice, most of the time max pooling is performed on a grid of size 3×3; therefore $q = 1$. For the sake of conciseness, we shall sometimes drop q in the notations, which implicitly means $q = 1$.

4.5 Adaptation to Multichannel Convolution Operators

In this section, we adapt Theorems 4.1 to 4.3 to multichannel inputs (*e.g.*, RGB images), employed in conventional CNNs such as AlexNet or ResNet. A detailed description of CNNs is provided in Chapter 2.

First, we define multichannel $\mathbb{C}$Mod and $\mathbb{R}$Max operators relatively to (4.3) and (4.1). We denote by K and $L \in \mathbb{N} \setminus \{0\}$ the number of input and output channels, respectively. Besides, we consider a *multichannel convolution tensor*

$$\mathbf{W} := (\mathrm{W}_{lk})_{l \in \{0..L-1\},\, k \in \{0..K-1\}} \in \left(l^2_{\mathbb{C}}(\mathbb{Z}^2)\right)^{L \times K}. \tag{4.186}$$

Multichannel $\mathbb{R}$Max and $\mathbb{C}$Mod operators take as input images, denoted by

$$\mathbf{X} := (\mathrm{X}_k)_{k \in \{0..K-1\}} \in \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^{K}. \tag{4.187}$$

They are defined, for any given output channel $l \in \{0\,..\,L-1\}$, by

$$U^{\max}_{m,\,q,\,l}[\mathbf{W}] : \mathbf{X} \mapsto \mathrm{MaxPool}_q \left(\sum_{k=0}^{K-1} \left(\mathrm{X}_k * \mathrm{Re}\, \overline{\mathrm{W}}_{lk} \right) \downarrow m \right); \tag{4.188}$$

$$U^{\mathrm{mod}}_{m,\,l}[\mathbf{W}] : \mathbf{X} \mapsto \left| \sum_{k=0}^{K-1} (\mathrm{X}_k * \overline{\mathrm{W}}_{lk}) \downarrow (2m) \right|, \tag{4.189}$$

where m, $q \in \mathbb{N} \setminus \{0\}$ respectively denote a subsampling factor and the max pooling grid half-size. Analogously to (4.84) for single-channel inputs, we now consider

$$\mathrm{Y}^{\max}_l := U^{\max}_{m,\,q,\,l}[\mathbf{W}]\,(\mathbf{X}) \qquad \text{and} \qquad \mathrm{Y}^{\mathrm{mod}}_l := U^{\mathrm{mod}}_{m,\,l}[\mathbf{W}]\,(\mathbf{X}). \tag{4.190}$$

Again, in what follows we omit the parameter between square brackets. To apply Theorems 4.1 to 4.3 to the current setting on the l-th output channel, we need the following hypotheses.

Hypothesis 4.4 (Monochorome filters)**.** *Let*

$$\widetilde{\mathrm{W}}_l := \frac{1}{K} \sum_{k=0}^{K-1} \mathrm{W}_{lk} \tag{4.191}$$

denote the mean kernel of the l-th output channel. Then, there exists $\boldsymbol{\mu}_l \in \mathbb{R}^K$ such that

$$\forall k \in \{0\,..\,K-1\}\,,\ \mathrm{W}_{lk} = \mu_{lk}\widetilde{\mathrm{W}}_l. \tag{4.192}$$

Hypothesis 4.5 (Gabor-like filters)**.** *There exists a bandwidth $\kappa > 0$ satisfying $\kappa \leq \pi/m$ and a frequency vector $\boldsymbol{\theta}_l \in [-\pi,\,\pi]^2$ such that*

$$\widetilde{\mathrm{W}}_l \in \mathcal{J}(\boldsymbol{\theta}_l,\,\kappa). \tag{4.193}$$

Note that the bandwidth κ is not indexed by l, because it shall later be assumed to be shared across the output channels. Then, under Hypothesis 4.4, $\mathrm{Y}_l^{\max}$ and $\mathrm{Y}_l^{\mathrm{mod}}$ are the outputs of single-channel $\mathbb{R}$Max and $\mathbb{C}$Mod operators, as introduced in (4.1) and (4.3):

$$\mathrm{Y}_l^{\max} = U_{m,\,q}^{\max}\big[\widetilde{\mathrm{W}}_l\big](\mathrm{X}_l^{\mathrm{lum}}) \qquad \text{and} \qquad \mathrm{Y}_l^{\mathrm{mod}} = U_m^{\mathrm{mod}}\big[\widetilde{\mathrm{W}}_l\big](\mathrm{X}_l^{\mathrm{lum}}), \tag{4.194}$$

where $\mathrm{X}_l^{\mathrm{lum}} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ ("luminance" image) is defined as the following linear combination:

$$\mathrm{X}_l^{\mathrm{lum}} := \sum_{k=0}^{K-1} \mu_{lk}\mathrm{X}_k. \tag{4.195}$$

The results established for single-channel inputs can therefore be extended to multichannel operators. Specifically, we get the following corollaries to Theorems 4.1 to 4.3.

Corollary 4.2 (Shift invariance of $\mathbb{C}$Mod)**.** *For a given output channel $l \in \{0\,..\,L-1\}$, we postulate Hypotheses 4.4 and 4.5. Then, for any input image $\mathbf{X} \in \big(l^2_{\mathbb{R}}(\mathbb{Z}^2)\big)^K$ with finite support and any translation vector $\boldsymbol{u} \in \mathbb{R}^2$,*

$$\big\|U_{m,\,l}^{\mathrm{mod}}(\mathcal{T}_{\boldsymbol{u}}\mathbf{X}) - U_{m,\,l}^{\mathrm{mod}}\mathbf{X}\big\|_2 \leq \alpha(\kappa\boldsymbol{u})\,\big\|U_{m,\,l}^{\mathrm{mod}}\mathbf{X}\big\|_2, \tag{4.196}$$

where α has been defined in (4.21).

Corollary 4.3 (Discrepancy between $\mathbb{C}$Mod and $\mathbb{R}$Max)**.** *As in Corollary 4.2, we postulate Hypotheses 4.4 and 4.5. Again, we assume that condition (4.60) is satisfied: $\kappa \leq \pi/m$. Besides, we consider $\mathbf{X}$ as a stack of K discrete stochastic processes, and assume Hypotheses 4.1 to 4.3 with $\mathrm{X} \leftarrow \mathrm{X}_l^{\mathrm{lum}}$ and $\mathrm{W} \leftarrow \widetilde{\mathrm{W}}_l$. Then,*

$$\mathbb{E}\big[\widetilde{\mathsf{P}}^2_{\mathbf{X},\,l}\big] \leq \big(\beta_q(m\kappa) + \gamma_q(m\boldsymbol{\theta}_l)\big)^2, \tag{4.197}$$

where we have defined the following random variable:

$$\widetilde{\mathsf{P}}_{\mathbf{X},\,l} := \big\|U_{m,\,l}^{\mathrm{mod}}\mathbf{X} - U_{m,\,l}^{\max}\mathbf{X}\big\|_2 / \big\|U_{m,\,l}^{\mathrm{mod}}\mathbf{X}\big\|_2. \tag{4.198}$$

Corollary 4.4 (Shift invariance of $\mathbb{R}$Max)**.** *We assume that the requirements stated in Corollary 4.3 are satisfied. Then, for any translation vector $\boldsymbol{u} \in \mathbb{R}^2$,*

$$\mathbb{E}\big[\widetilde{\mathsf{R}}_{\mathbf{X},\,\boldsymbol{u},\,l}\big] \leq 2\,\big(\beta_q(m\kappa) + \gamma_q(m\boldsymbol{\theta}_l)\big) + \alpha(\kappa\boldsymbol{u}), \tag{4.199}$$

where we have defined the following random variable:

$$\widetilde{\mathsf{R}}_{\mathbf{X},\,\boldsymbol{u},\,l} := \big\|U_{m,\,l}^{\max}(\mathcal{T}_{\boldsymbol{u}}\mathbf{X}) - U_{m,\,l}^{\max}\mathbf{X}\big\|_2 / \big\|U_{m,\,l}^{\mathrm{mod}}\mathbf{X}\big\|_2. \tag{4.200}$$

Remark 4.9. In the above results, we used a translation operator on multichannel tensors, obtained by applying $\mathcal{T}_{\boldsymbol{u}}$, as defined in (4.41), to each channel X_k.

In Section 5.3, we shall examine the applicability of the above results to freely-trained models such as AlexNet or ResNet. Specifically, the validity of Hypotheses 4.4 and 4.5 will be evaluated experimentally.

4.6 A Case Study Implementing the Dual-Tree Complex Wavelet Packet Transform

In this section, we experimentally validate the results stated in Theorems 4.1 to 4.3. To this end, we consider a fully-deterministic setting based on the dual-tree complex wavelet packet transform (DT-ℂWPT), for which a detailed description and the principles upon which it is designed is provided in Section 3.3. As formally established in Sections 4.6.1 and 4.6.2, for a given decomposition depth $J \in \mathbb{N} \setminus \{0\}$, DT-ℂWPT achieves subsampled convolutions with oriented band-pass filters, tiling the Fourier domain into 4×4^J overlapping square windows of size $\kappa_J := \pi/m_J$, where we have denoted $m_J := 2^{J-1}$. Based on this, in Section 4.6.3, we build ℂMod and ℝMax operators, respectively satisfying (4.3) and (4.1) with $m \leftarrow m_J$. Note that increasing the decomposition depth J, and therefore the subsampling factor m_J, results in a decreased Fourier support size κ_J, therefore matching the condition stated in (4.60) with $\kappa \leftarrow \kappa_J$ and $m \leftarrow m_J$. DT-ℂWPT thus provides a convenient framework to experimentally validate Theorems 4.1 to 4.3 in a controlled environment. As will be evidenced in Section 5.3, the initial convolution layer in CNNs such as AlexNet or ResNet behaves in a similar way.

4.6.1 Convolution Operators

We first examine the real-valued WPT algorithm in its fully-decomposed version, such as introduced in Section 3.2.5. In this context, we use the notations introduced in Section 3.2 (real-valued wavelet transforms). Considering $J \in \mathbb{N} \setminus \{0\}$ as the decomposition depth, input images $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ are decomposed in a pseudo-local cosine basis $\mathbf{E}^{(J)}$ (3.64). The following lemma introduces an array of convolution kernels $\mathbf{V}^{(J)} := \big(\mathrm{V}_l^{(J)}\big)_{l \in \{0..4^J-1\}}$ characterizing WPT. It is a simple reformulation of the well-known result that two successive convolutions can be written as another convolution with a wider kernel. A visual representation of these kernels is provided in Figure 4.4 (left) with $J = 2$.

Lemma 4.7. *For any $l \in \{0\,..\,4^J - 1\}$,*

$$\mathrm{D}_l^{(J)} = \Big(\mathrm{X} * \overline{\mathrm{V}}_l^{(J)}\Big) \downarrow 2^J, \qquad \textit{with} \qquad \mathrm{V}_l^{(J)} := \mathrm{E}_{l,\mathbf{0}}^{(J)}, \tag{4.201}$$

where $\mathrm{E}_{l,\mathbf{0}}^{(J)}$ (atom of the discrete wavelet packet basis) and $\mathrm{D}_l^{(J)} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ (feature map of wavelet packet coefficients) have been introduced in (3.62) and (3.63), respectively.

Proof. We use the decomposition formula in an orthonormal basis. For any $\boldsymbol{n} \in \mathbb{Z}^2$,

$$\mathrm{D}_l^{(J)}[\boldsymbol{n}] = \Big\langle \mathrm{X},\, \mathrm{E}_{l,\boldsymbol{n}}^{(J)} \Big\rangle. \tag{4.202}$$

Besides, employing an inductive reasoning approach to (3.62) demonstrates that each basis function $\mathrm{E}_{l,\boldsymbol{n}}^{(J)}$ can be derived from $\mathrm{V}_l^{(J)} := \mathrm{E}_{l,\mathbf{0}}^{(J)}$ by shifting it by the vector $2^J\boldsymbol{n}$:

$$\mathrm{E}_{l,\boldsymbol{n}}^{(J)}[\boldsymbol{p}] = \mathrm{V}_l^{(J)}[\boldsymbol{p} - 2^J\boldsymbol{n}]. \tag{4.203}$$

Therefore, (4.202) can be rewritten as a convolution product with $\overline{\mathrm{V}}_l^{(J)}$, which concludes the proof. □

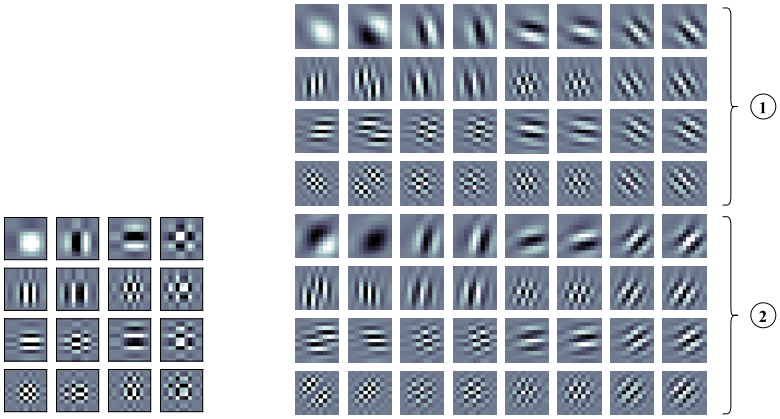


Figure 4.4. Convolution kernels $\mathbf{V}^{(J)}$ for WPT (left) and $\mathbf{W}^{\nearrow(J)}$, $\mathbf{W}^{\searrow(J)}$ for DT-$\mathbb{C}$WPT (right ① and ②, respectively). The number of decomposition stages J is set to 2. In both cases, the kernels have been computed using Q-shift orthogonal QMFs of length 10 (Kingsbury, 2003). The kernels have been cropped to size 11×11 for the sake of legibility. Both $\mathbf{W}^{\nearrow(J)}$ and $\mathbf{W}^{\searrow(J)}$ contain 16 complex filters, alternatively represented by their real and imaginary parts.

Despite interesting properties such as sparsity of image representations, WPT is unstable with respect to small shifts and suffers from a poor directional selectivity. To overcome this, Kingsbury (1999) designed a discrete wavelet transform where images are decomposed in a redundant frame of nearly-analytic, complex-valued waveforms. It was later extended to the wavelet packet framework (Bayram and I. W. Selesnick, 2008), giving birth to the dual-tree complex wavelet packet transform (DT-$\mathbb{C}$WPT), described in Section 3.3.4. In the following, we use the notations introduced in Section 3.3 (complex wavelet transforms).

Based on Lemma 4.7, the following proposition introduces two arrays of complex kernels $\mathbf{W}^{\nearrow(J)} := (\mathrm{W}_l^{\nearrow(J)})_{l\in\{0..4^J-1\}}$ and $\mathbf{W}^{\searrow(J)} := (\mathrm{W}_l^{\searrow(J)})_{l\in\{0..4^J-1\}}$, characterizing DT-$\mathbb{C}$WPT when the decomposition is performed in a pseudo-local cosine frame (3.101). A graphical representation of these kernels is provided in Figure 4.4 (right) with $J = 2$.

Proposition 4.8. *For any* $l \in \{0\,..\,4^J - 1\}$,

$$\mathrm{D}_l^{\nearrow(J)} = \left(\mathrm{X} * \overline{\mathrm{W}}_l^{\nearrow(J)^*}\right) \downarrow 2^J, \qquad \textit{with} \qquad \mathrm{W}_l^{\nearrow(J)} := \mathrm{E}_{l,\mathbf{0}}^{\nearrow(J)}, \tag{4.204}$$

where $\mathrm{E}_{l,\mathbf{0}}^{\nearrow(J)}$ *(atom of the discrete frame) and* $\mathrm{D}_l^{\nearrow(J)} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ *(feature map of complex wavelet packet coefficients) satisfy* (3.103) *and* (3.100)*, respectively, with* $j \leftarrow J$*. Identical results are obtained with the three other Fourier quadrants.*

Proof. For each of the four filter banks $m \in \{0\,..\,3\}$, and any channel $l \in \{0\,..\,4^J - 1\}$, Lemma 4.7 provides a convolution kernel $\mathrm{V}_l^{[m](J)} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ such that

$$\mathrm{D}_l^{[m](J)} = \left(\mathrm{X} * \overline{\mathrm{V}}_l^{[m](J)}\right) \downarrow 2^J. \tag{4.205}$$

Then, we insert (4.205) into (3.100) for all $m \in \{0\,..\,3\}$. Finally, the expression of $\mathrm{E}_{l,\mathbf{0}}^{\nearrow(J)}$ in (3.103) yields the result. □

Remark 4.10. To ease implementation, we have considered a variant of DT-ℂWPT where the low-frequency atoms $\mathrm{E}_{0,\boldsymbol{n}}^{\nearrow(J)}$ and $\mathrm{E}_{0,\boldsymbol{n}}^{\searrow(J)}$ are complex-valued. This is different from Section 3.3.4, where the low-frequency atoms $\mathrm{E}_{0,\boldsymbol{n}}^{(J)}$ remained real-valued (3.101).

Remark 4.11. A similar result can be obtained in the continuous framework:

$$\mathrm{D}_l^{\nearrow(J)}[\boldsymbol{n}] = \Big(F * \overline{\Psi}_l^{\nearrow(J)^*}\Big)(2^J\boldsymbol{n}), \qquad \text{with} \qquad \Psi_l^{\nearrow(J)} := \Psi_{l,\mathbf{0}}^{\nearrow(J)}, \tag{4.206}$$

where, as introduced in Section 3.3.3, $F \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ denotes the continuous function from which $\mathrm{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ is uniformly sampled, and $\Psi_{l,\mathbf{0}}^{\nearrow(J)}$, satisfying (3.104), is an atom of the wavelet packet tight frame $\boldsymbol{\Psi}_{\mathbb{C}}^{(J)}$ introduced in (3.102).

Remark 4.12. Since input images are real-valued, we can discard the kernels $\mathrm{W}_l^{\swarrow(J)}$ and $\mathrm{W}_l^{\nwarrow(J)}$ without loss of information, because the corresponding feature maps of wavelet packet coefficients, $\mathrm{D}_l^{\swarrow(J)}$ and $\mathrm{D}_l^{\nwarrow(J)}$, are simply the complex conjugates of $\mathrm{D}_l^{\nearrow(J)}$ and $\mathrm{D}_l^{\searrow(J)}$, respectively.

4.6.2 Gabor-Like Convolution Kernels

We now show that the convolution kernels $\mathrm{W}_l^{\nearrow(J)}$ and $\mathrm{W}_l^{\searrow(J)}$, introduced in (4.204), approximately behave as Gabor-like filters, as defined in (4.12). To begin with, we assume that $\mathrm{h}^{[0]}$ is a Shannon filter, satisfying (3.58). Let $J \in \mathbb{N} \setminus \{0\}$ denote the number of decomposition stages. The following proposition states that DT-ℂWPT tiles the frequency plane with square windows.

Proposition 4.9. *There exists a permutation* $(\boldsymbol{\sigma}_l^{(J)})_{l\in\{0..4^J-1\}}$ *of* $\{0\,..\,2^J-1\}^2$ *such that, for any* $l \in \{0\,..\,4^J-1\}$,

$$\Psi_l^{\nearrow(J)} \in \mathcal{V}(\boldsymbol{\theta}_l^{(J)}, \kappa_J), \tag{4.207}$$

where $\Psi_l^{\nearrow(J)}$ *has been introduced in Remark 4.11, and where we have defined*

$$\boldsymbol{\theta}_l^{(J)} := \left(\boldsymbol{\sigma}_l^{(J)} + \frac{1}{2}\right)\frac{\pi}{2^J} \qquad \textit{and} \qquad \kappa_J := \frac{\pi}{2^J}. \tag{4.208}$$

We remind the reader that $\mathcal{V}(\boldsymbol{\nu}, \varepsilon)$, *defined in* (4.7), *denotes a space of Gabor-like filters in the continuous framework.*

Proof. From the construction of the wavelet packet tight frame presented in Section 3.3.4, we can show that $\Psi_l^{\nearrow(J)}$ is the tensor product of two 1D wavelet packets:

$$\Psi_l^{\nearrow(J)} = \psi_{l_1}^{(J)} \otimes \psi_{l_2}^{(J)}, \tag{4.209}$$

for some indices l_1 and $l_2 \in \{0\,..\,2^J-1\}$. Moreover, for any $l' \in \{0\,..\,2^J-1\}$, we have

$$\psi_{l'}^{(J)} = \psi_{l'}^{[0](J)} + i\,\psi_{l'}^{[1](J)}, \tag{4.210}$$

where $\psi_{l'}^{[0](J)} \in L^2_{\mathbb{R}}(\mathbb{R})$ is an atom of the standard Shannon wavelet packet orthonormal basis, and $\psi_{l'}^{[1](J)}$ is the Hilbert transform of $\psi_{l'}^{[0](J)}$, as defined in (3.76). Therefore, since the Hilbert transform suppresses negative frequencies, we get

$$\widehat{\psi}_{l'}^{(J)} = 2\,\widehat{\psi}_{l'}^{[0](J)}\,\mathbb{1}_{\mathbb{R}_+}. \tag{4.211}$$

Consequently, according to the Coifman-Wickerhauser theorem (Mallat, 2009, pp. 384-385), there exists $k \in \{0\,..\,2^J - 1\}$ such that

$$\operatorname{supp} \widehat{\psi}_{l'}^{(J)} \subset \left[\frac{k\pi}{2^J}, \frac{(k+1)\pi}{2^J}\right]. \tag{4.212}$$

Finally, the tensor product (4.209) yields the result. □

According to Proposition 4.9, each atom $\Psi_l^{\nearrow(J)}$, for $l \in \{0\,..\,4^J - 1\}$, is supported in a square window of size $\kappa_J \times \kappa_J$ included in the top-right quadrant of the Fourier domain. Similar results can be obtained for the three remaining quadrants, with $\Psi_l^{\searrow(J)}$, $\Psi_l^{\swarrow(J)}$ and $\Psi_l^{\nwarrow(J)}$. Using the analogy between (4.204) and (4.206), we would like to deduce from Proposition 4.9 that the discrete filter $\mathrm{W}_l^{\nearrow(J)} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ satisfies

$$\mathrm{W}_l^{\nearrow(J)} \in \mathcal{J}(\boldsymbol{\theta}_l^{(J)}, \kappa_J), \tag{4.213}$$

which is a space of Gabor-like filters in the discrete framework (4.12). However, as mentioned in Remark 3.6 (p. 52), a "perfect" dual-tree transform should be initialized with four different inputs $\mathrm{X}^{[0-3]}$. Instead, all four WPT decompositions are performed on the same input X. Consequently, DT-ℂWPT is not perfectly analytic and (4.213) is only valid asymptotically, when J goes to ∞. In fact, the Fourier support of $\mathrm{W}_l^{\nearrow(J)}$ is contained in four square regions of size κ_J (one in each quadrant), its energy becoming negligible outside the top-right quadrant when J increases. Nevertheless, employing, in the first stage, a specific pair of low-pass filters satisfying the one-sample delay condition (3.93) yields near-analytic solutions even for small values of J. We therefore consider (4.213) as a reasonable approximation if $J \geq 2$.

Remark 4.13. Proposition 4.9 tiles the top-right Fourier quadrant with 4^J square cells of size $\kappa_J := \pi/2^J$. However, as explained in Section 3.2.4, the Shannon wavelet is poorly suited for sparse image representations, because of its slow rate of decay. Moreover, it deviates from what is typically observed in freely-trained CNNs, because $\mathrm{W}_l^{\nearrow(J)}$ must be approximated with very large filters to avoid numerical instabilities. Practical implementations of DT-ℂWPT use fast-decaying filters such as these associated to Meyer wavelets (3.59), or finite-length filters that approximate the half-sample delay condition (I. W. Selesnick et al., 2005). Therefore, energy is leaking outside the square cells tiling the Fourier domain. To counterbalance this, we increase the window size up to

$$\kappa_J := \frac{\pi}{2^{J-1}} = \pi/m_J, \tag{4.214}$$

and consider that (4.213) remains a reasonable approximation. Therefore, the conditions to apply Theorems 4.1 to 4.3 are approximately satisfied in this context.

Depth J	Bandwidth κ_J	Mean	Std
2	$\pi/2$	0.98	0.00
3	$\pi/4$	0.95	0.02

Table 4.1. Energy concentration of the DT-ℂWPT filters within a Fourier window of size $\kappa_J \times \kappa_J$, with $\kappa_J := \pi/2^{J-1}$.

In order to numerically assess this assumption, we measured the maximum percentage of energy within a square window of size $\kappa_J \times \kappa_J$ in the Fourier domain:

$$\rho_l^{\nearrow} := \frac{\max_{\boldsymbol{\theta} \in [-\pi,\,\pi]^2} \left\| \mathbb{1}_{B_\infty(\boldsymbol{\theta},\,\kappa_J/2)} \widehat{\mathrm{W}}_l^{\nearrow(J)} \right\|_{L^2}^2}{\left\| \widehat{\mathrm{W}}_l^{\nearrow(J)} \right\|_{L^2}^2}, \tag{4.215}$$

where the l^∞-ball $B_\infty(\boldsymbol{\theta},\, \kappa_J/2)$ is defined in the quotient space $[-\pi,\, \pi]^2 / (2\pi\mathbb{Z}^2)$, as explained in Remark 4.2 (p. 69). If (4.213) is perfectly satisfied, then $\rho_l^{\nearrow} = 1$. The statistics computed over the collection $(\rho_l^{\nearrow},\, \rho_l^{\searrow})_{l \in \{0..4^J-1\}}$ are reported in Table 4.1.

Remark 4.14. For "boundary filters", *i.e.*, when $\|\boldsymbol{\theta}_l^{(J)}\|_\infty = (1 - 2^{-(J+1)})\,\pi$, Remark 4.2 states that a small fraction of the filter's energy remains located at the far end of the Fourier domain—see also Bayram and I. W. Selesnick (2008). Therefore, these filters do not strictly comply with the conditions of Theorems 4.1 to 4.3. We nevertheless include them in our experiments.

4.6.3 DT-ℂWPT-Based ℝMax and ℂMod Operators

According to (4.204), (4.213) and (4.214), we can apply Theorems 4.1 to 4.3 to the dual-tree framework. More precisely, for any output channel $l \in \{0\,..\,4^J - 1\}$, we consider the following ℝMax and ℂMod operators:

$$U_l^{\mathrm{max}\nearrow} : \mathrm{X} \mapsto \mathrm{MaxPool}\left(\left(\mathrm{X} * \mathrm{Re}\,\overline{\mathrm{W}}_l^{\nearrow(J)}\right) \downarrow 2^{J-1}\right); \tag{4.216}$$

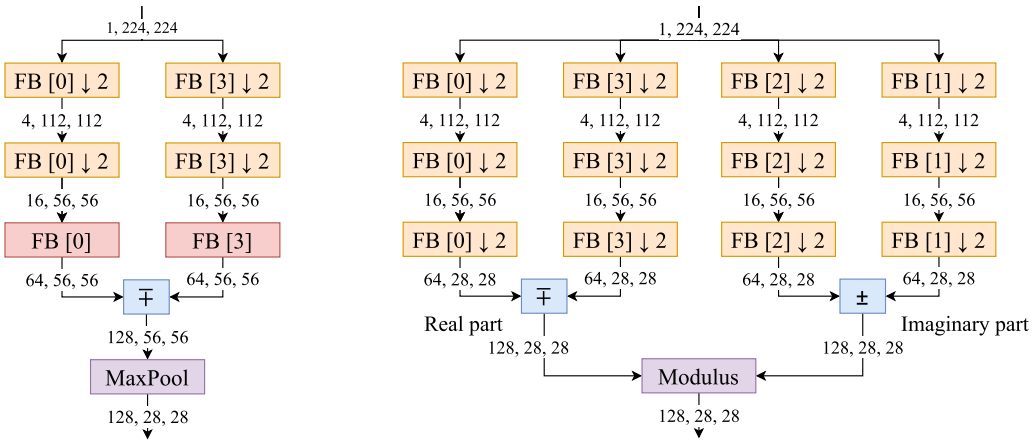
$$U_l^{\mathrm{mod}\nearrow} : \mathrm{X} \mapsto \left|\left(\mathrm{X} * \overline{\mathrm{W}}_l^{\nearrow(J)}\right) \downarrow 2^{J}\right|. \tag{4.217}$$

Using the notations introduced in (4.3) and (4.1), we have

$$U_l^{\mathrm{max}\nearrow} = U_{m_J}^{\mathrm{max}}[\mathrm{W}_l^{\nearrow(J)}] \qquad \text{and} \qquad U_l^{\mathrm{mod}\nearrow} := U_{m_J}^{\mathrm{mod}}[\mathrm{W}_l^{\nearrow(J)}], \tag{4.218}$$

where we have defined $m_J := 2^{J-1}$. Note that, following Remark 4.8, we have omitted the grid half-size q, which is equal to 1 (max pooling operates on a grid of size 3×3). Furthermore, for the sake of brevity, we have omitted the depth J in the above notations.

Remark 4.15. Both $U_l^{\mathrm{max}\nearrow}$ and $U_l^{\mathrm{mod}\nearrow}$ are implemented using DT-ℂWPT with J decomposition stages. However, in (4.216), the subsampling factor is equal to 2^{J-1}, instead of 2^J, as stated in Proposition 4.8. In order to accommodate this property of ℝMax operators, the last stage of DT-ℂWPT decomposition is carried out without subsampling, resulting in higher redundancy. This is similar to the concept of stationary wavelet transform as described by Nason and Silverman (1995). Furthermore, only the real component



(a) $\mathbb{R}$Max (b) $\mathbb{C}$Mod

Figure 4.5. Detailed illustration of the $\mathbb{R}$Max (a) and $\mathbb{C}$Mod (b) operators based on DT-$\mathbb{C}$WPT, with $J = 3$ decomposition stages. The numbers between modules correspond to the number of feature maps, height and width. The orange modules represent subsampled convolutions using one of the four 2D filter banks $\mathbf{G}^{[0-3]}$, such as introduced in (3.20). The FB index is indicated between square brackets. The $\mathbb{R}$Max model (a) only computes the real part of the dual-tree coefficients, and the last stage of decomposition is performed without subsampling (red modules). Additionally, the blue modules represent linear combinations of feature maps such as described in (3.100).

of the wavelet feature maps is preserved. On the other hand, $U_l^{\mathrm{mod}\nearrow}$ implements a fully-decimated wavelet packet transform, and keeps both real and imaginary parts. Figure 4.5 illustrates these technical details.

4.6.4 Experiments and Results

We implemented the $\mathbb{R}$Max and $\mathbb{C}$Mod operators $U_l^{\mathrm{max}\nearrow}$ and $U_l^{\mathrm{mod}\nearrow}$, as introduced in (4.216) and (4.217), with both $J = 2$ and 3 stages of wavelet packet decomposition. To cover the whole frequency plane, we also implemented similar operators, denoted by $U_l^{\mathrm{max}\searrow}$ and $U_l^{\mathrm{mod}\searrow}$. They are associated with the convolution filters $\mathrm{W}_l^{\searrow(J)}$, introduced in Proposition 4.8, with energy being located in the bottom-right quadrant. However, as explained in Remark 4.12, we did not need to deal with the two other quadrants (negative x-values), since input images are real-valued. Using the validation set of ImageNet-1K (Russakovsky et al., 2015), ($N := 50\,000$ images), we measured the mean discrepancy between $\mathbb{R}$Max and $\mathbb{C}$Mod outputs, and evaluated the shift invariance of both models. Dual-tree decompositions have been performed with Q-shift orthogonal filters of length 10 (Kingsbury, 2003), which approximately meets the half-sample delay condition (3.91).

Discrepancies between $\mathbb{R}$Max and $\mathbb{C}$Mod. Each image $n \in \{0\,..\,N-1\}$ in the dataset was converted to grayscale, from which a center crop of size 224×224 was extracted. We denote by $\mathrm{X}_n \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ the resulting input feature map. For any $l \in \{0\,..\,4^J-1\}$, we denote by

$$\mathrm{Y}_{nl}^{\mathrm{max}\nearrow} := U_l^{\mathrm{max}\nearrow}(\mathrm{X}_n) \qquad \text{and} \qquad \mathrm{Y}_{nl}^{\mathrm{mod}\nearrow} := U_l^{\mathrm{mod}\nearrow}(\mathrm{X}_n) \tag{4.219}$$

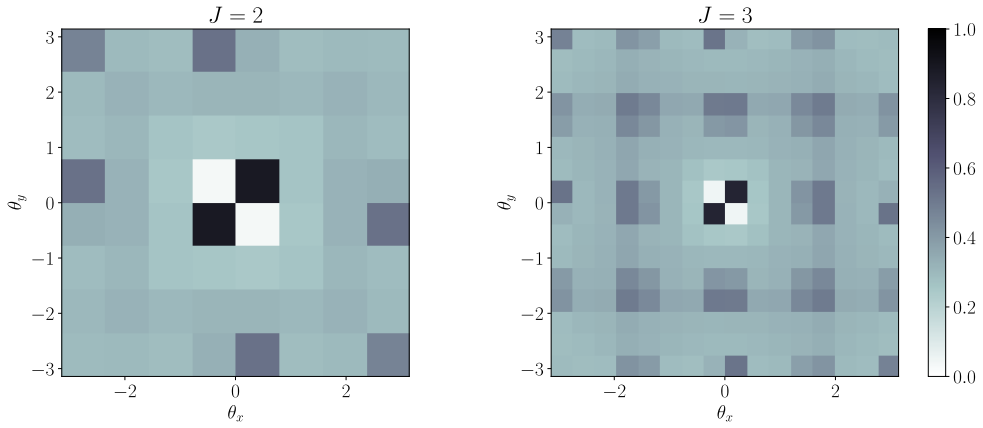
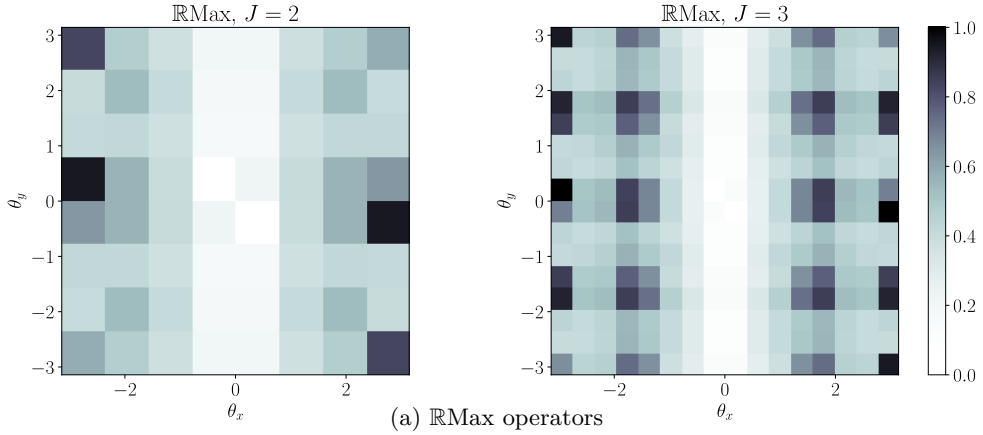
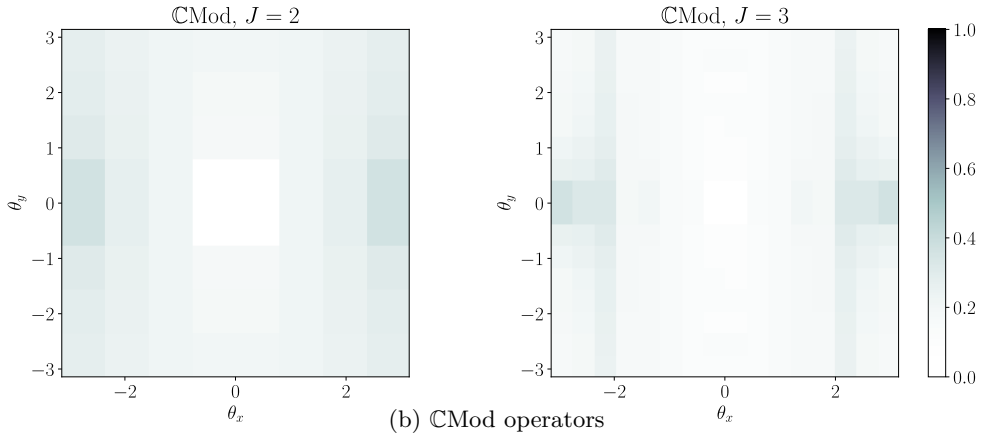


Figure 4.6. Empirical estimates of the normalized mean squared error between ℝMax and ℂMod outputs, computed on ImageNet-1K (validation set). For each channel $l \in \{0\,..\,4^J - 1\}$, $\widetilde{\rho}_l^{\nearrow 2}$ is plotted as a grayscale pixel centered in $\boldsymbol{\theta}_l^{(J)}$ such as introduced in Proposition 4.9 (top-right quadrant). Similarly, $\widetilde{\rho}_l^{\searrow 2}$ is plotted in the bottom-right quadrant. Finally, the bottom- and top-left quadrants ($\widetilde{\rho}_l^{\swarrow 2}$ and $\widetilde{\rho}_l^{\nwarrow 2}$) are simply obtained by symmetrizing the figures. Since the subsampling factor m_J is equal to 2^{J-1}, these experimental results can be compared with the left and right parts of Figure 4.3.



(a) ℝMax operators



(b) ℂMod operators

Figure 4.7. Shift invariance of ℝMax and ℂMod outputs, computed on ImageNet 2012 (validation set). For each $l \in \{0\,..\,4^J - 1\}$, $\widetilde{\rho}_l^{\max\nearrow}$ (Figure 4.7a) and $\widetilde{\rho}_l^{\mathrm{mod}\nearrow}$ (Figure 4.7b) are plotted by applying the same procedure as in Figure 4.6.

the outputs of the l-th $\mathbb{R}\mathrm{Max}$ and $\mathbb{C}\mathrm{Mod}$ operators as defined in (4.216) and (4.217), respectively. We adopt similar notations for the bottom-right Fourier quadrant. Then, the normalized mean squared error between $\mathrm{Y}_{nl}^{\mathrm{mod}\nearrow}$ and $\mathrm{Y}_{nl}^{\mathrm{max}\nearrow}$ was computed. It is defined by the square of

$$\rho_{nl}^{\nearrow} := \|\mathrm{Y}_{nl}^{\mathrm{mod}\nearrow} - \mathrm{Y}_{nl}^{\mathrm{max}\nearrow}\|_2 / \|\mathrm{Y}_{nl}^{\mathrm{mod}\nearrow}\|_2. \tag{4.220}$$

Finally, the for each output channel l, an empirical estimate for $\mathbb{E}[\widetilde{\mathsf{P}}_{\mathrm{X}}^2]$, introduced in (4.133), was obtained by averaging $\rho_{nl}^{\nearrow 2}$ over the whole dataset. We denote by $\widetilde{\rho}_l^{\nearrow 2}$ the corresponding quantity.

Since $U_l^{\mathrm{max}\nearrow}$ and $U_l^{\mathrm{mod}\nearrow}$ are parameterized by $\mathrm{W}_l^{\nearrow(J)}$, it follows that $\widetilde{\rho}_l^{\nearrow 2}$ depends on the filter's characteristic frequency $\boldsymbol{\theta}_l^{(J)}$ (4.213). According to Proposition 4.9, these frequencies form a regular grid in the top-right quadrant of Fourier domain. This provides a visual representation of $\widetilde{\rho}_l^{\nearrow 2}$, as shown in Figure 4.6. This figure also displays $\widetilde{\rho}_l^{\searrow 2}$, corresponding to the bottom-right quadrant. The half-plane of negative x-values has simply been symmetrized, following Remark 4.12. We can observe a regular pattern of dark spots. More precisely, high discrepancies between max pooling and modulus seem to occur when the energy of $\mathrm{W}_l^{\nearrow(J)}$ or $\mathrm{W}_l^{\searrow(J)}$ overlaps a dark region of Figure 4.3. This result corroborates Theorem 4.3, which states that high discrepancies are expected for certain pathological frequencies, due to the search for a maximum value over a discrete grid.

Shift invariance. For each input image previously converted to grayscale, two crops of size 224×224 were extracted, such that the corresponding sequences X_n and X'_n are shifted by one pixel along the x-axis. From these inputs, the following quantity was then computed:

$$\rho_{nl}^{\mathrm{max}\nearrow} := \|\mathrm{Y}_{nl}^{\mathrm{max}'\nearrow} - \mathrm{Y}_{nl}^{\mathrm{max}\nearrow}\|_2 \,/\, \|\mathrm{Y}_{nl}^{\mathrm{mod}\nearrow}\|_2, \tag{4.221}$$

where $\mathrm{Y}_{nl}^{\mathrm{max}'\nearrow}$ satisfies (4.218) with $\mathrm{X}_n \leftarrow \mathrm{X}'_n$. Finally, for each output channel $l \in \{0\,..\,4^J - 1\}$, an empirical estimate for $\mathbb{E}[\widetilde{\mathsf{R}}_{\mathrm{X},\boldsymbol{u}}]$, satisfying (4.178) with $\boldsymbol{u} = (1,\,0)^\top$, was obtained by averaging $\rho_{nl}^{\mathrm{max}\nearrow}$ over the whole dataset. We denote by $\widetilde{\rho}_l^{\mathrm{max}\nearrow}$ the corresponding quantity. We point out that shift invariance is measured relatively to the norm of the $\mathbb{C}\mathrm{Mod}$ output, as explained in Remark 4.7.

On the other hand, the same procedure was applied to the $\mathbb{C}\mathrm{Mod}$ operators:

$$\rho_{nl}^{\mathrm{mod}\nearrow} := \|\mathrm{Y}_{nl}^{\mathrm{mod}'\nearrow} - \mathrm{Y}_{nl}^{\mathrm{mod}\nearrow}\|_2 \,/\, \|\mathrm{Y}_{nl}^{\mathrm{mod}\nearrow}\|_2, \tag{4.222}$$

and $\widetilde{\rho}_l^{\mathrm{mod}\nearrow}$ was obtained as before by averaging $\rho_{nl}^{\mathrm{mod}\nearrow}$ over the whole dataset.

A visual representation of $\widetilde{\rho}_l^{\mathrm{max}\nearrow}$ and $\widetilde{\rho}_l^{\mathrm{mod}\nearrow}$ are provided in Figure 4.7 (as well as the other Fourier quadrants). Two observations can be drawn here. (1) When the filter is horizontally oriented, the corresponding output is highly stable with respect to horizontal shifts. This can be explained by noticing that such kernels perform low-pass filtering along the x-axis. The exact transposed phenomenon occurs for vertical shifts. (2) Elsewhere, we observe that high discrepancies between $\mathbb{R}\mathrm{Max}$ and $\mathbb{C}\mathrm{Mod}$ outputs (Figure 4.6) are correlated with shift instability of $\mathbb{R}\mathrm{Max}$ (Figure 4.7, top). This is in line with (4.162) and (4.179) in Theorems 4.2 and 4.3. Note that $\mathbb{C}\mathrm{Mod}$ outputs are nearly shift invariant regardless the characteristic frequency $\boldsymbol{\theta}_l^{(J)}$ (Figure 4.7, bottom), as predicted by Theorem 4.1 (4.61).

4.7 Concluding Remarks

In this chapter, we explored the shift invariance properties captured by the max pooling operator, when applied on top of a convolution layer with Gabor-like kernels. We established a validity domain for near-shift invariance and confirmed our predictions through an experimental setting based on the dual-tree complex wavelet packet transform. Our results indicate that the $\mathbb{C}$Mod operator can serve as a proxy for $\mathbb{R}$Max, extracting comparable, yet more stable features. This suggests a promising approach for improving shift invariance in CNNs while preserving high-frequency information. This is the main focus of Chapter 5, in which we apply these "antialiasing" principles to real-life architectures.

4.A Appendix: Theoretical Foundations for our Hypotheses

In this section, we provide theoretical arguments for justifying Hypotheses 4.2 and 4.3. Given $n \in \mathbb{N} \setminus \{0\}$, we define *n-th order stationarity* of a given stochastic process F as stated by K. I. Park and M. Park (2018, p. 152): for any $n' \in \{0\,..\,n-1\}$, $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n'}) \in (\mathbb{R}^2)^{n'}$ and $\boldsymbol{h} \in \mathbb{R}^2$, the joint distribution of $(\mathsf{F}(\boldsymbol{x}_1), \ldots, \mathsf{F}(\boldsymbol{x}_{n'}))$ is identical to the one of $(\mathsf{F}(\boldsymbol{x}_1 + \boldsymbol{h}), \ldots, \mathsf{F}(\boldsymbol{x}_{n'} + \boldsymbol{h}))$. Besides, *strict-sense stationarity* is defined as n-th order stationarity for any $n \in \mathbb{N} \setminus \{0\}$.

We recall that $\boldsymbol{\nu} := \boldsymbol{\theta}/s$. We then state the following results.

Proposition 4.10. *We assume that F_X is first-order stationary. If, for any $\boldsymbol{x} \in \mathbb{R}^2$ and any $\boldsymbol{h} \in B_2(2\pi/\|\boldsymbol{\nu}\|_2)$,*

$$(\mathcal{T}_{\boldsymbol{h}}\mathsf{F}_\mathrm{X} * \overline{\Psi}_\mathrm{W})(\boldsymbol{x}) = \mathrm{e}^{i\langle \boldsymbol{\nu}, \boldsymbol{h}\rangle}(\mathsf{F}_\mathrm{X} * \overline{\Psi}_\mathrm{W})(\boldsymbol{x}), \tag{4.223}$$

then Hypothesis 4.2 is satisfied.

Proof. Let $\boldsymbol{x} \in \mathbb{R}^2$. By design (see Remark 4.6), $\mathsf{Z}_\mathrm{X}(\boldsymbol{x})$ follows a uniform conditional probability distribution on $\mathbb{S}^1$, given $\mathsf{M}_\mathrm{X}(\boldsymbol{x}) = 0$. In any other cases, we show that the conditional probability measure of $\mathsf{Z}_\mathrm{X}(\boldsymbol{x})$ given $\mathsf{M}_\mathrm{X}(\boldsymbol{x}) > 0$ is invariant with respect to phase shifts, and is therefore equal to the uniform probability measure on $\mathbb{S}^1$. Specifically, we show that, for any measurable set $\mathfrak{A} \subset \mathbb{S}^1$,

$$\forall \omega \in [0,\, 2\pi]\,,\ \mu(\mathfrak{A}) = \mu(\mathrm{e}^{i\omega}\mathfrak{A}), \tag{4.224}$$

where we have denoted

$$\mu : \mathfrak{A} \mapsto \mathbb{P}\left\{\mathsf{Z}_\mathrm{X}(\boldsymbol{x}) \in \mathfrak{A} \mid \mathsf{M}_\mathrm{X}(\boldsymbol{x}) > 0\right\}. \tag{4.225}$$

Let $\boldsymbol{h} \in B_2(2\pi/\|\boldsymbol{\nu}\|_2)$. According to (4.223), and assuming $\mathsf{M}_\mathrm{X}(\boldsymbol{x}) > 0$, we get

$$\mathsf{Z}_\mathrm{X}(\boldsymbol{x}) \in \mathfrak{A} \iff \mathcal{T}_{\boldsymbol{h}}\mathsf{Z}_\mathrm{X}(\boldsymbol{x}) \in \mathrm{e}^{i\langle \boldsymbol{\nu}, \boldsymbol{h}\rangle}\mathfrak{A}. \tag{4.226}$$

Therefore,

$$\mathbb{P}\left\{\mathsf{Z}_\mathrm{X}(\boldsymbol{x}) \in \mathfrak{A} \mid \mathsf{M}_\mathrm{X}(\boldsymbol{x}) > 0\right\} = \mathbb{P}\big\{\mathcal{T}_{\boldsymbol{h}}\mathsf{Z}_\mathrm{X}(\boldsymbol{x}) \in \mathrm{e}^{i\langle \boldsymbol{\nu}, \boldsymbol{h}\rangle}\mathfrak{A} \mid \mathsf{M}_\mathrm{X}(\boldsymbol{x}) > 0\big\}. \tag{4.227}$$

Since F_{X} is first-order stationary, $\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x})$ and $\mathcal{T}_{\boldsymbol{h}}\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x})$ have the same conditional probability distribution given $\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0$. Thus we get

$$\mathbb{P}\left\{\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) \in \mathfrak{A} \mid \mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0\right\} = \mathbb{P}\{\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) \in \mathrm{e}^{i\langle \boldsymbol{\nu}, \boldsymbol{h}\rangle}\mathfrak{A} \mid \mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0\}. \tag{4.228}$$

Let $\omega \in [0, 2\pi]$. Considering $\boldsymbol{h} := \omega\,\boldsymbol{\nu}/\left\|\boldsymbol{\nu}\right\|_2^2$, we have

$$\boldsymbol{h} \in B_2(2\pi/\left\|\boldsymbol{\nu}\right\|_2) \qquad \text{and} \qquad \langle \boldsymbol{\nu}, \boldsymbol{h}\rangle = \omega. \tag{4.229}$$

Therefore,

$$\forall \omega \in [0, 2\pi]\,, \mathbb{P}\left\{\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) \in \mathfrak{A} \mid \mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0\right\} = \mathbb{P}\{\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) \in \mathrm{e}^{i\omega}\mathfrak{A} \mid \mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0\}, \tag{4.230}$$

which yields (4.224).

Any probability measure defined on $\mathbb{S}^1$ is a Radon measure. Therefore, according to Haar's theorem (Halmos, 2013), there exists a unique probability measure on $\mathbb{S}^1$ satisfying (4.224). Since the uniform probability measure is also invariant to phase shifts, we deduce that $\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x})$ is uniformly distributed on $\mathbb{S}^1$, conditionally to $\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0$, which concludes the proof. □

Proposition 4.11. *We assume the conditions of Proposition 4.10 are met. If, moreover, F_{X} is strict-sense stationary, then Hypothesis 4.3 is satisfied.*

Proof. Let $n \in \mathbb{N} \setminus \{0\}$ and $\boldsymbol{x}, \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{n-1} \in \mathbb{R}^2$. To alleviate notations, we consider the random vector $\mathbf{M} = \left(\mathsf{M}_{\mathrm{X}}(\boldsymbol{y}_0), \ldots, \mathsf{M}_{\mathrm{X}}(\boldsymbol{y}_{n-1})\right)^\top$ with outcomes in $\mathbb{R}_+^n$. According to (4.128), $\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x})$ is conditionally independent of $\mathbf{M}$ given $\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) = 0$. Therefore, it remains to prove conditional independence given $\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0$.

The proof is organized as follows. Using a similar reasoning as Proposition 4.10, we show that, for any measurable subset $\mathfrak{S} \subset \mathbb{R}_+^n$, Z_{X} follows a uniform probability distribution conditionally to $\mathbf{M} \in \mathfrak{S}$ and $\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0$. Since we already know that Z_{X} follows a uniform distribution conditionally to $\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0$ alone, we deduce that Z_{X} and $\mathbf{M}$ are conditionally independent given $\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0$.

Let $\mathfrak{A} \subset \mathbb{S}^1$ and $\mathfrak{S} := (\mathfrak{S}_i)_{i\in\{0..n-1\}} \subset \mathbb{R}_+^n$ denote measurable sets. According to (4.223), and assuming $\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0$, we get, for any $\boldsymbol{h} \in B_2(2\pi/\left\|\boldsymbol{\nu}\right\|_2)$,

$$\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) \in \mathfrak{A} \iff \mathcal{T}_{\boldsymbol{h}}\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) \in \mathrm{e}^{i\langle \boldsymbol{\nu}, \boldsymbol{h}\rangle}\mathfrak{A}; \tag{4.231}$$

$$\mathsf{M}_{\mathrm{X}}(\boldsymbol{y}_i) \in \mathfrak{S}_i \iff \mathcal{T}_{\boldsymbol{h}}\mathsf{M}_{\mathrm{X}}(\boldsymbol{y}_i) \in \mathfrak{S}_i \quad \forall i \in \{0\,..\,n-1\}\,. \tag{4.232}$$

Therefore,

$$\begin{aligned}\mathbb{P}\Big\{(\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) \in \mathfrak{A}) \,\&\, (\mathbf{M} \in \mathfrak{S}) \;\Big|\; \mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0\Big\} \\ = \mathbb{P}\left\{(\mathcal{T}_{\boldsymbol{h}}\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) \in \mathrm{e}^{i\langle \boldsymbol{\nu}, \boldsymbol{h}\rangle}\mathfrak{A}) \,\&\, (\mathcal{T}_{\boldsymbol{h}}\mathbf{M} \in \mathfrak{S}) \;\middle|\; \mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0\right\}.\end{aligned} \tag{4.233}$$

Since F_{X} is strict-sense stationary, the joint conditional probability density of

$$\mathcal{T}_{\boldsymbol{h}}\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}),\, \mathcal{T}_{\boldsymbol{h}}\mathsf{M}_{\mathrm{X}}(\boldsymbol{y}_0),\, \ldots,\, \mathcal{T}_{\boldsymbol{h}}\mathsf{M}_{\mathrm{X}}(\boldsymbol{y}_{n-1}) \tag{4.234}$$

is identical to the one of

$$\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}),\, \mathsf{M}_{\mathrm{X}}(\boldsymbol{y}_0),\, \ldots,\, \mathsf{M}_{\mathrm{X}}(\boldsymbol{y}_{n-1}). \tag{4.235}$$

Therefore we get

$$\mathbb{P}\Big\{(\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) \in \mathfrak{A}) \,\&\, (\mathbf{M} \in \mathfrak{S}) \;\Big|\; \mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0\Big\} = \mathbb{P}\Big\{(\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) \in \mathrm{e}^{i\langle \boldsymbol{\nu}, \boldsymbol{h}\rangle}\mathfrak{A}) \,\&\, (\mathbf{M} \in \mathfrak{S}) \;\Big|\; \mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0\Big\}. \quad (4.236)$$

We assume that $\mathbb{P}(\mathbf{M} \in \mathfrak{S}) > 0$. According to the above expression, and similarly to the proof of Proposition 4.10, we get,

$$\forall \omega \in [0,\, 2\pi]\,,\, \mathbb{P}\Big\{\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) \in \mathfrak{A} \;\Big|\; (\mathbf{M} \in \mathfrak{S}) \,\&\, (\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0)\Big\} = \mathbb{P}\Big\{\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) \in \mathrm{e}^{i\omega}\mathfrak{A} \;\Big|\; (\mathbf{M} \in \mathfrak{S}) \,\&\, (\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0)\Big\}. \quad (4.237)$$

Then, the above conditional probability measure satisfies phase shift invariance (4.224). Therefore, as in the proof of Proposition 4.10, Haar's theorem implies that $\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x})$ follows a uniform conditional distribution given $\mathbf{M} \in \mathfrak{S}$ and $\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0$.

Moreover, strict-sense implies first-order stationarity, and thus, according to the proof of Proposition 4.10, $\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x})$ follows a uniform distribution conditionally to $\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0$. Therefore we get, for any measurable sets $\mathfrak{A} \subset \mathbb{S}^1$ and $\mathfrak{S} \subset \mathbb{R}^n_+$ such that $\mathbb{P}(\mathbf{M} \in \mathfrak{S}) > 0$,

$$\mathbb{P}\{\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) \in \mathfrak{A} \mid (\mathbf{M} \in \mathfrak{S}) \,\&\, (\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0)\} = \mathbb{P}\{\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x}) \in \mathfrak{A} \mid \mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0\}, \quad (4.238)$$

which proves conditional independence between $\mathsf{Z}_{\mathrm{X}}(\boldsymbol{x})$ and $\mathbf{M}$ given $\mathsf{M}_{\mathrm{X}}(\boldsymbol{x}) > 0$, and concludes the proof. □

Remark 4.16 (Stationarity hypothesis)**.** Strict-sense stationarity suggests that any translated version of a given image is equally likely. In reality, this statement is too strong, for several reasons. First, by construction, X has all its realizations in $L^2_{\mathbb{R}}(\mathbb{R}^2)$. In that context, a stationary process yields outcomes which are zero almost everywhere. Besides, depending on which category the image belongs to, the pixel distribution is likely to vary across various regions. For instance, we can expect the main subject to be located at the center of the image. More details on statistical properties of images from natural versus man-made objects can be found in a paper by Torralba and Oliva (2003). Nevertheless, this hypothesis will be considered as a reasonable approximation if the shift is much smaller than the image "characteristic" size in the continuous domain; *i.e.*, if

$$\|\boldsymbol{h}\|_2 \ll sN, \quad (4.239)$$

where, as a reminder, N denotes the support size of input images. We refer the reader to Tygert et al. (2016) for a related notion of local stationarity. As it turns out, the proofs of Propositions 4.10 and 4.11 only requires shifts with $\|\boldsymbol{h}\|_2 \leq 2\pi/\|\boldsymbol{\nu}\|_2$. Therefore, the constraint on $\|\boldsymbol{\theta}\|_2$ stated in (4.131) implies (4.239), and the stationarity hypothesis holds.

Remark 4.17 (Justification for (4.223))**.** We consider

$$\Phi_{\mathrm{W}} : \boldsymbol{x} \mapsto \Psi_{\mathrm{W}}(\boldsymbol{x})\mathrm{e}^{-i\langle \boldsymbol{\nu}, \boldsymbol{x}\rangle}. \quad (4.240)$$

Similarly to Lemma 4.1, we can show that Φ_{W} is a low-pass filter, with $\operatorname{supp}\widehat{\Phi}_{\mathrm{W}} \subset B_\infty(\varepsilon/2)$. For all $\boldsymbol{h} \in \mathbb{R}^2$ such that $\|\boldsymbol{h}\|_2 \leq 2\pi/\|\boldsymbol{\nu}\|_2$, we have

$$\begin{aligned}(\mathcal{T}_{\boldsymbol{h}}\mathsf{F}_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}})(\boldsymbol{x}) &= \iint_{\mathbb{R}^2} \mathcal{T}_{\boldsymbol{h}}\mathsf{F}_{\mathrm{X}}(\boldsymbol{x}-\boldsymbol{y})\,\overline{\Phi}_{\mathrm{W}}(\boldsymbol{y})\,\mathrm{e}^{-i\langle \boldsymbol{\nu}, \boldsymbol{y}\rangle}\,\mathrm{d}^2\boldsymbol{y} \\ &= \mathrm{e}^{i\langle \boldsymbol{\nu}, \boldsymbol{h}\rangle} \iint_{\mathbb{R}^2} \mathsf{F}_{\mathrm{X}}(\boldsymbol{x}-\boldsymbol{y}')\,\overline{\Phi}_{\mathrm{W}}(\boldsymbol{y}'-\boldsymbol{h})\,\mathrm{e}^{-i\langle \boldsymbol{\nu}, \boldsymbol{y}'\rangle}\,\mathrm{d}^2\boldsymbol{y}'.\end{aligned}$$

Since $\operatorname{supp} \widehat{\Phi}_{\mathrm{W}} \subset B_\infty\left(\frac{\kappa}{2s}\right)$, we can define a "minimal wavelength" $\lambda_{\Phi_{\mathrm{W}}} := 2\pi s/\kappa$. Then, if $\|\boldsymbol{h}\|_2 \ll \lambda_{\Phi_{\mathrm{W}}}$, we can approximate $\overline{\Phi}_{\mathrm{W}}(\boldsymbol{y}' - \boldsymbol{h}) \approx \overline{\Phi}_{\mathrm{W}}(\boldsymbol{y}')$. This sufficient condition is actually met, because $\|\boldsymbol{h}\|_2 \leq 2\pi/\|\boldsymbol{\nu}\|_2$ and, according to (4.132), $\|\boldsymbol{\nu}\|_2 \gg \kappa/s$. Therefore,

$$(\mathcal{T}_{\boldsymbol{h}} \mathsf{F}_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}})(\boldsymbol{x}) \approx \mathrm{e}^{i\langle \boldsymbol{\nu}, \boldsymbol{h} \rangle} (\mathsf{F}_{\mathrm{X}} * \overline{\Psi}_{\mathrm{W}})(\boldsymbol{x}). \tag{4.241}$$

As explained in Remarks 4.16 and 4.17, the sufficient conditions outlined in Propositions 4.10 and 4.11 are not strictly met. Nevertheless, we consider that Hypotheses 4.2 and 4.3 still provide a reasonable description of the distribution from which input images are drawn.

Chapter 5

Shift-Invariant Twin Models Based on Complex Wavelets

Translation invariance in CNNs is often taken for granted. This misconception comes from the equivariance property of convolutions in the continuous domain: convolutions commute with translations. In this context, shifting an input will result in an equally shifted output. Under this assumption, Wiatowski and Bölcskei (2018) showed that a wide variety of models becomes progressively more shift invariant with increasing network depth. However, the situation changes when working on discrete sequences. Subsampling operations, typically found in convolution and pooling layers, are an important source of instability—a phenomenon known as aliasing, as discussed in Section 2.4.2 and illustrated in Figure 3.3 in the context of wavelet transforms.

This chapter introduces a novel antialiasing method to increase shift invariance and prediction accuracy in CNNs. Specifically, we replace the first-layer combination "real-valued convolutions → max pooling" ($\mathbb{R}$Max) by "complex-valued convolutions → modulus" ($\mathbb{C}$Mod), which is stable to translations. This approach is justified by our findings from Chapter 4. Specifically, $\mathbb{C}$Mod and $\mathbb{R}$Max produce comparable outputs when the convolution kernel is band-pass and oriented (Gabor-like filter). In this context, $\mathbb{C}$Mod can be considered as a stable alternative to $\mathbb{R}$Max. Thus, prior to antialiasing, we force the convolution kernels to adopt such a Gabor-like structure. The corresponding architecture is called mathematical twin, because it employs a well-defined mathematical operator to mimic the behavior of the original, freely-trained model. By retaining high-frequency details, our antialiasing approach achieves a better balance between shift invariance and information preservation, compared to concurrent methods based on low-pass filtering. This results in improved classification accuracy on ImageNet and CIFAR-10, with a lower computational cost and memory footprint.

This contribution chapter is a synthesis of the following two papers:

- H. Leterme, K. Polisano, V. Perrier, and K. Alahari (2021). "Modélisation Parcimonieuse de CNNs Avec Des Paquets d'Ondelettes Dual-Tree". In: *ORASIS*.

- H. Leterme, K. Polisano, V. Perrier, and K. Alahari (2023). "From CNNs to Shift-Invariant Twin Models Based on Complex Wavelets". arXiv: 2212.00394.

5.1 Existing vs Proposed Antialiasing Methods

Blurpooled CNNs. To improve stability to translations in CNNs, a line of work is focused on designing antialiasing methods based on low-pass filtering, also called *blur pooling* in this context. As discussed in Section 2.4.2, this approach is inspired by the Nyquist-Shannon sampling theorem (Shannon, 1949), which implies that high-frequency signals must be blurred before subsampling, in order to avoid artifacts in reconstruction. R. Zhang (2019) was the first paper to introduce blur pooling in CNNs, using linear convolutions with fixed filters. Later, X. Zou et al. (2023) enhanced the method by using a nonlinear, adaptive blur pooling layer, thus preserving high-frequency information when necessary. In this nonlinear setting, the blurring filter varies across channels and image locations, therefore preserving high-frequency information in strategic zones. Albeit achieving higher prediction accuracy, this approach remains fundamentally based on low-pass filtering. Consequently, features that are not blurred may still be unstable to translations. Furthermore, adaptive blur pooling requires additional memory, computational resources, and trainable parameters.

Proposed Approach. In this chapter, we propose an alternative antialiasing approach based on complex-valued convolutions, extracting high-frequency features that are stable to translations. We observed improved accuracy for ImageNet and CIFAR-10 classification, compared to the two above antialiasing methods. Furthermore, our approach offers significant advantages in terms of computational efficiency and memory usage, and does not induce any additional training, unlike adaptive blur pooling.

Our proposed method replaces the first layers of a CNN:

$$\text{Conv} \to \text{Sub} \to \text{Bias} \to \text{ReLU} \to \text{MaxPool}, \tag{5.1}$$

equivalently written as

$$\text{Conv} \to \text{Sub} \to \text{MaxPool} \to \text{Bias} \to \text{ReLU}, \tag{5.2}$$

by the following combination:

$$\mathbb{C}\text{Conv} \to \text{Sub} \to \text{Modulus} \to \text{Bias} \to \text{ReLU}, \tag{5.3}$$

where $\mathbb{C}$Conv denotes a convolution operator with a complex-valued kernel, whose real and imaginary parts approximately form a 2D Hilbert transform pair (Havlicek et al., 1997). From (5.2) and (5.3), we introduce the two following operators:

$$\mathbb{R}\text{Max} \ : \ \text{Conv} \to \text{Sub} \to \text{MaxPool}; \tag{5.4}$$

$$\mathbb{C}\text{Mod} \ : \ \mathbb{C}\text{Conv} \to \text{Sub} \to \text{Modulus}. \tag{5.5}$$

This method is motivated by our theoretical contribution presented in Chapter 4: when the convolution kernel is band-pass and oriented (Gabor-like filter), the $\mathbb{C}$Mod operator can be considered as a proxy for $\mathbb{R}$Max, extracting comparable, yet more stable features. In compliance with these results, the $\mathbb{R}$Max-$\mathbb{C}$Mod substitution is only applied to the output channels associated with Gabor-like filters which, as explained in Section 2.4.1, arise spontaneously in the first layer of CNNs trained on image datasets. In this chapter, we

enforce this property by applying additional constraints to the original model, prior to antialiasing. Specifically, a predefined number of convolution kernels are guided to adopt Gabor-like structures, instead of letting the network learn them from scratch. For this purpose, we rely on the dual-tree complex wavelet packet transform (DT-ℂWPT), presented in Section 3.3. Throughout the chapter, we refer to this constrained model as a *mathematical twin*, because it employs a well-defined mathematical operator to mimic the behavior of the original model. In this context, replacing ℝMax by ℂMod is straightforward, since the complex-valued filters are provided by DT-ℂWPT.

Other Related Work. As mentioned in Section 2.4.2, Chaman and Dokmanic (2021) designed a perfectly shift-invariant architecture using an adaptive, input-dependent subsampling grid. However, this approach is not intended to compete with other antialiasing methods, but rather to complement them at the subsampling stages.

Comparable to our work, wavelet scattering networks (ScatterNets), described in Section 3.4, also take advantage of complex-valued convolutions to produce shift-invariant image representations that are stable to deformation and preserve high-frequency information. However, as mentioned in Section 3.5.3, comparing complex-valued ScatterNets and real-valued CNNs is not straightforward, due to the differences in their architectural design. In contrast, owing to the proximity between the ℝMax and ℂMod operators, our models are enhanced versions of existing networks, rather than ad hoc constructions. This allows drawing a bond between the real and complex worlds, and to compare ℝMax- and ℂMod-based models in terms of prediction accuracy and shift invariance.

Finally, we draw the reader's attention to the family of complex-valued convolutional neural networks (ℂVCNNs), presented in Section 2.3.3. These models are tailored for tasks where preserving the phase information is essential to achieve high performances, but do not perform better than standard CNNs on image classification tasks. Conversely, our approach discards the phase information by computing the modulus. In summary, ℂVCNNs and our models, although both employing complex-valued convolutions, are suited for different contexts.

5.2 Subject of Study: First Layers in CNNs

Throughout the chapter, we use the notations introduced in Section 4.2.1. Our study deals with the initial layers of a CNN as sketched in (5.1). A convolution layer with K input channels, L output channels and subsampling factor $m \in \mathbb{N} \setminus \{0\}$ is parameterized by a weight tensor $\mathbf{V} := (\mathrm{V}_{lk})_{l\in\{0..L-1\},\, k\in\{0..K-1\}} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)^{L\times K}$. For any multichannel input $\mathbf{X} := (\mathrm{X}_k)_{k\in\{0..K-1\}} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)^K$, the corresponding output $\mathbf{Y} := (\mathrm{Y}_l)_{l\in\{0..L-1\}} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)^L$ is defined such that, for any output channel $l \in \{0\,..\,L-1\}$,

$$\mathrm{Y}_l := \sum_{k=0}^{K-1} (\mathrm{X}_k * \overline{\mathrm{V}_{lk}}) \downarrow m = \sum_{k=0}^{K-1} (\mathrm{X}_k \star \mathrm{V}_{lk}) \downarrow m. \tag{5.6}$$

From now on, we shall employ the *cross-correlation product* $\star$, instead of the convolution product $*$, for the sake of convenience.

In AlexNet and ResNet, $K = 3$ (RGB input images), $L = 64$. Furthermore, in AlexNet, the weight tensor $\mathbf{V}$ is supported in a region of size 11×11 and the subsampling factor

m (stride) is equal to 4. ResNet models use kernels of size 7×7 with $m = 2$. Next, a bias $\boldsymbol{b} := (b_1, \cdots, b_L)^\top \in \mathbb{R}^L$ is applied to $\mathbf{Y}$, which is then transformed through nonlinear ReLU and max pooling operators. The activated outputs satisfy

$$\mathrm{A}_l^{\max} := \mathrm{MaxPool}\left(\mathrm{ReLU}(\mathrm{Y}_l + b_l)\right), \tag{5.7}$$

where MaxPool has been introduced in (4.2) with $q = 1$ (see Remark 4.8), and ReLU is defined such that, for any $\mathrm{Y} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ and any $\boldsymbol{n} \in \mathbb{Z}^2$,

$$\mathrm{ReLU}(\mathrm{Y})[\boldsymbol{n}] := \max\bigl(0, \mathrm{Y}[\boldsymbol{n}]\bigr). \tag{5.8}$$

Expression (5.7) also employs the bias notation introduced in (2.48).

For any output channel $l \in \{0\,..\,L-1\}$, we denote by

$$\widetilde{\mathrm{V}}_l := \frac{1}{K} \sum_{k=0}^{K-1} \mathrm{V}_{lk} \tag{5.9}$$

the mean kernel computed over the input channels. This operation is justified by the monochrome hypothesis stated in Hypothesis 4.4 for Gabor-like kernels, and experimentally assessed in Section 5.3.2. We now consider, as in Section 4.2.2, an analytic complex-valued companion of $\widetilde{\mathrm{V}}_l$, denoted by $\widetilde{\mathrm{W}}_l \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$, obtained by setting to zero the kernel's discrete-time Fourier transform on half of the frequency plane. More specifically, we denote by $\boldsymbol{u}_l \in \mathbb{R}^2$ the dominant eigenvector of the *structure tensor* associated with $\widetilde{\mathrm{V}}_l$ (Bigun et al., 1991). This unit vector is, by design, orthogonal to the filter's orientation. Then, $\widetilde{\mathrm{W}}_l$ is defined by

$$\widetilde{\mathrm{W}}_l := \widetilde{\mathrm{V}}_l + i\mathcal{H}_{\boldsymbol{u}_l}(\widetilde{\mathrm{V}}_l), \tag{5.10}$$

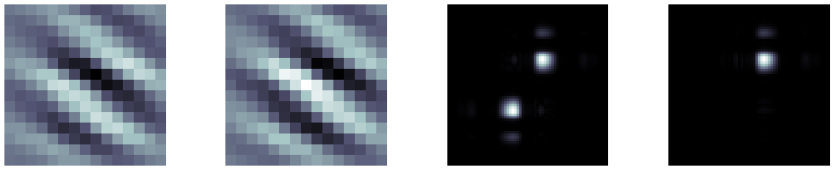
where $\mathcal{H}_{\boldsymbol{u}_l}$ denotes a two-dimensional *Hilbert transform*, adapted from Havlicek et al. (1997). It satisfies, for any unit vector $\boldsymbol{u} \in \mathbb{R}^2$, real-valued sequence $\mathrm{V} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ and frequency $\boldsymbol{\omega} \in [-\pi, \pi]^2$,

$$\widehat{\mathcal{H}_{\boldsymbol{u}}(\mathrm{V})}(\boldsymbol{\omega}) := -i\,\mathrm{sgn}\langle \boldsymbol{\omega}, \boldsymbol{u}\rangle \cdot \widehat{\mathrm{V}}(\boldsymbol{\omega}). \tag{5.11}$$

The Hilbert transform is designed such that the Fourier transform of $\widetilde{\mathrm{W}}_l$ is entirely supported in the half-plane $\{\boldsymbol{\omega} \in [-\pi, \pi]^2 \mid \langle \boldsymbol{\omega}, \boldsymbol{u}_l\rangle \geq 0\}$. Therefore, if $\widetilde{\mathrm{V}}_l$ is band-pass and oriented (Gabor-like filter), then the energy of $\widetilde{\mathrm{W}}_l$ is concentrated in a small window in the Fourier domain, as depicted in Figure 5.1d. Finally, for any $k \in \{0\,..\,K-1\}$ and $l \in \{0\,..\,L-1\}$, we construct the complex-valued companions of $\mathrm{V}_{lk} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$:

$$\mathrm{W}_{lk} := \mathrm{V}_{lk} + i\mathcal{H}_{\boldsymbol{u}_l}(\mathrm{V}_{lk}). \tag{5.12}$$

Remark 5.1. The 2D Hilbert transform as defined in (5.11) spawns a linear boundary which is orthogonal to $\boldsymbol{u}$, referred to as the *cut-off boundary*. If the Fourier support of $\widetilde{\mathrm{V}}_l$ overlaps this boundary, then $\widetilde{\mathrm{W}}_l$ will not have the "well-behaved" frequency-localization property observed in Figure 5.1d. To avoid this shortcoming, we have chosen $\boldsymbol{u} := \boldsymbol{u}_l$ to be orthogonal to the filter's orientation.



(a) (b) (c) (d)

Figure 5.1. (a), (b): Real and imaginary parts of the Gabor-like filter $\widetilde{\mathrm{W}}_l$ from AlexNet, satisfying (5.10) with $l = 14$ (see also Figure 5.2a). (c), (d): Magnitude spectra (modulus of the Fourier transform) of $\widetilde{\mathrm{V}}_l$ and $\widetilde{\mathrm{W}}_l$, respectively.

5.3 Applicability of our Theoretical Results to CNNs

Before delving deeper into our antialiasing approach, we assess whether the theoretical results established in Section 4.5 are applicable to freely-trained CNNs. Considering AlexNet and ResNet-34 trained with ImageNet ILSVRC 2012-2017 (Russakovsky et al., 2015), also called ImageNet-1K, we experimentally verify that a substantial number of convolution kernels in the first layer are monochrome and Gabor-like. Specifically, we check that, for a certain number of output channels $l \in \{0\,..\,63\}$, Hypotheses 4.4 and 4.5 are approximately satisfied.

5.3.1 Identifying the Gabor-like Kernels

To conduct our experiments, the first action was to identify the Gabor-like kernels in the first layer. To this end, we applied the following procedure. For any output channel $l \in \{0\,..\,63\}$, we denote by $\widetilde{\mathrm{V}}_l := \frac{1}{3}\sum_{k=0}^{2} \mathrm{V}_{lk}$ the mean kernel computed over the RGB input channels. Whether $\widetilde{\mathrm{V}}_l$ has a clearly-defined orientation can be determined by the *coherence index* $c_l \in [0,\,1]$ (Bigun et al., 1991), defined by

$$c_l := (\lambda_{l,\,0} - \lambda_{l,\,1})/(\lambda_{l,\,0} + \lambda_{l,\,1}) \in [0,\,1]\,, \tag{5.13}$$

where $\lambda_{l,\,0}$ and $\lambda_{l,\,1}$ denote the eigenvalues of the structure tensor associated to $\widetilde{\mathrm{V}}_l$ (see Remark 5.1), sorted by descending order of their absolute values. To select convolution kernels with well-defined orientations, we applied a lower threshold on c_l, arbitrarily chosen to $c_{\mathrm{inf}} := 0.8$. Furthermore, we removed the low-pass filters by applying an upper threshold, set to $s_{\mathrm{sup}} := 0.1$, on the normalized Fourier transform at frequency $\boldsymbol{\omega} = \mathbf{0}$, denoted by

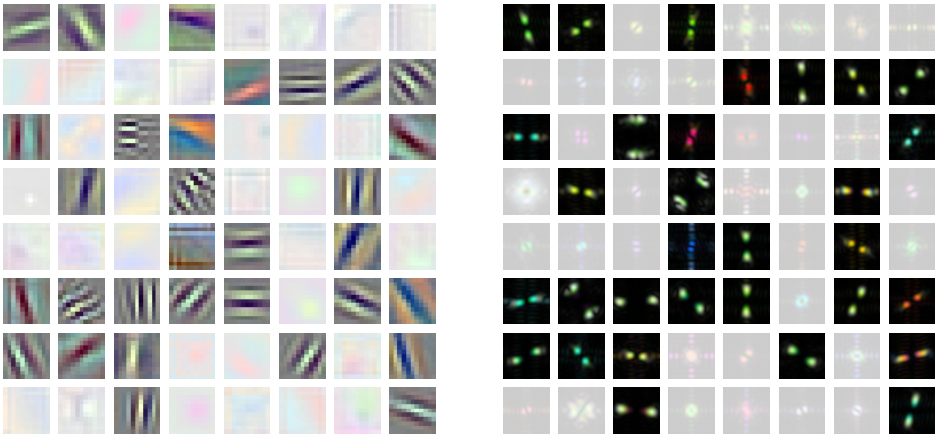
$$s_l := \big|\widehat{\widetilde{\mathrm{V}}}_l(\mathbf{0})\big| / \big\|\widetilde{\mathrm{V}}_l\big\|_1 \in [0,\,1]\,. \tag{5.14}$$

Subsequently, we obtained a subset $\mathcal{G} \subset \{0\,..\,63\}$ of *Gabor channels*, defined by

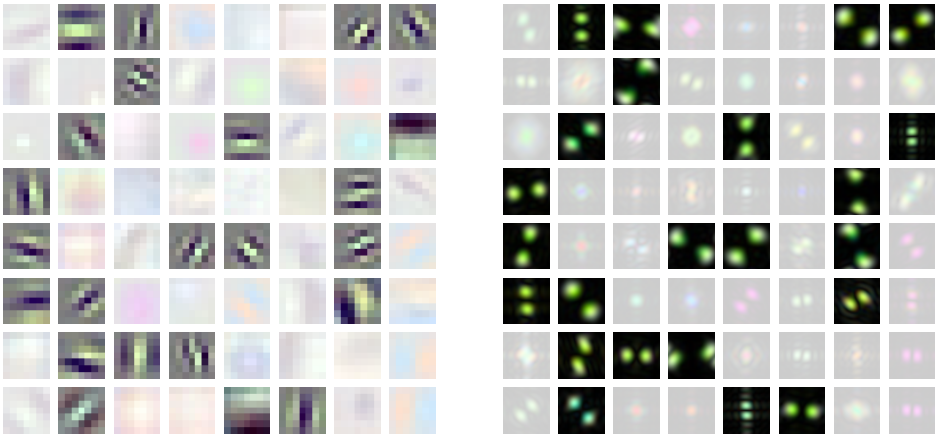
$$\mathcal{G} := \{l \in \{0\,..\,63\} \mid (c_l \geq c_{\mathrm{inf}}) \text{ and } (s_l \leq s_{\mathrm{sup}})\}\,. \tag{5.15}$$

Figure 5.2 highlights the convolution kernels selected by this method. AlexNet and ResNet-34 models were trained on ImageNet-1K, following the standard procedure provided by PyTorch (Paszke et al., 2017).[1] The number of Gabor channels is equal to 31 for AlexNet and 23 for ResNet. We notice that the selected kernels have various frequencies

[1]The PyTorch "examples" repository is available at https://github.com/pytorch/examples/tree/main/imagenet



(a) AlexNet (31 Gabor channels)



(b) ResNet (23 Gabor channels)

Figure 5.2. Convolution kernels in the first layer of AlexNet (a) and ResNet-34 (b), after training with ImageNet-1K. For each output channel $l \in \{0\,..\,63\}$, the corresponding convolution kernel $(\mathrm{V}_{lk})_{k\in\{0..2\}}$ is displayed as an RGB image in the spatial domain (left), and its associated magnitude spectrum in the Fourier domain (right). The highlighted kernels correspond to the Gabor channels $l \in \mathcal{G}$, empirically determined using the method from Section 5.3.1.

and orientations. Furthermore, by looking at the Fourier transforms, most of them appear monochrome (not necessarily grayscale), which supports Hypothesis 4.4.[2] A more formal assessment of this hypothesis is presented in Section 5.3.2. Besides, the bandwidth κ appears larger for ResNet than for AlexNet. Since the latter model has a larger subsampling factor m than the former ($m = 4$ vs $m = 2$), this observation is in line with Hypothesis 4.5, which states that $\kappa \leq \pi/m$. Again, more precise measurements are presented in Section 5.3.1.

As evidenced by Figure 5.2, the method has some limitations. First, it is fundamentally prone to false positives. An example is the kernel in the 5-th row and 4-th column in Figure 5.2a. We can see from the Fourier transform that it has a well-defined orientation

[2]The spatial representation of V_{lk} for $l \in \mathcal{G}$ sometimes displays two colors. However, this is misleading, as they correspond to positive and negative values.

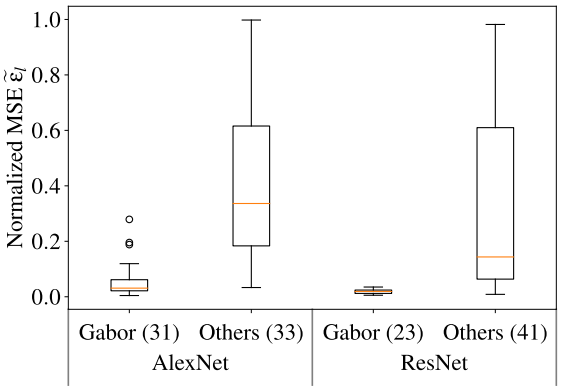


Figure 5.3. Box plots representing the distribution of the normalized mean squared errors $\varepsilon_{lk} \in [0, 1]$ between the convolution kernels V_{lk} and their projection on $\mathbb{R}\widetilde{\mathrm{V}}_l$, averaged over the input RGB channels ($\widetilde{\varepsilon}_l$), for the first convolution layers of AlexNet and ResNet-34 trained on ImageNet-1K. The data have been split between the Gabor channels $l \in \mathcal{G}$ and the others. The size of each statistical sample is indicated between parentheses.

but no clear absolute frequency. Moreover, depending on the chosen thresholds c_{inf} and s_{sup}, more false positives or false negatives may appear.

5.3.2 Monochrome Kernels

We now experimentally assess Hypothesis 4.4. We consider, for any output channel $l \in \{0\,..\,63\}$ and any RGB input channel $k \in \{0\,..\,2\}$, the value of $\mu \in \mathbb{R}$ minimizing $\left\|\mu\widetilde{\mathrm{V}}_l - \mathrm{V}_{lk}\right\|_2^2$, denoted by μ_{lk}. We then denote by $\varepsilon_{lk} := \left\|\mu_{lk}\widetilde{\mathrm{V}}_l - \mathrm{V}_{lk}\right\|_2^2 / \left\|\mathrm{V}_{lk}\right\|_2^2$ the normalized mean squared error between V_{lk} and its projection on $\mathbb{R}\widetilde{\mathrm{V}}_l$. We get

$$\varepsilon_{lk} = 1 - \frac{\langle \widetilde{\mathrm{V}}_l,\, \mathrm{V}_{lk} \rangle^2}{\left\|\widetilde{\mathrm{V}}_l\right\|_2^2 \cdot \left\|\mathrm{V}_{lk}\right\|_2^2}. \tag{5.16}$$

Then, we compute a weighted average over the RGB channels:

$$\widetilde{\varepsilon}_l := \frac{\sum_{k=0}^{2} \left\|\mathrm{V}_{lk}\right\|_2^2 \cdot \varepsilon_{lk}}{\sum_{k=0}^{2} \left\|\mathrm{V}_{lk}\right\|_2^2}. \tag{5.17}$$

For a given output channel $l \in \{0\,..\,63\}$, Hypothesis 4.4 holds if and only if $\widetilde{\varepsilon}_l = 0$. Figure 5.3 displays the distributions of $\widetilde{\varepsilon}_l$ for $l \in \mathcal{G}$ on the one hand and $l \in \{0\,..\,63\} \setminus \mathcal{G}$ on the other hand. We observe that $\widetilde{\varepsilon}_l \ll 1$ for a large majority of Gabor channels, with some exceptions for AlexNet. Consequently, we will consider Hypothesis 4.4 as a reasonable approximation for any Gabor channel $l \in \mathcal{G}$.

5.3.3 Kernel Bandwidth

In practice, Hypothesis 4.5 cannot be exactly satisfied. This is because $\widetilde{\mathrm{V}}_l = \mathrm{Re}\,\widetilde{\mathrm{W}}_l$ is finitely supported, and thus its power spectrum cannot be exactly zero on a region with non-zero measure. To evaluate how close we are to this ideal situation, we measured the

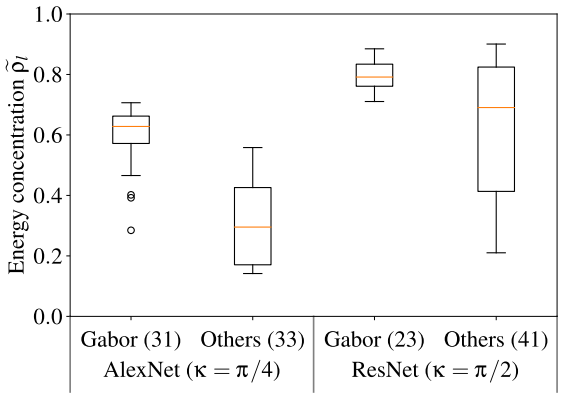


Figure 5.4. Energy concentration of $\widetilde{\mathrm{W}}_l$ within a Fourier window of size $\kappa \times \kappa$, with $\kappa := \pi/m$. If Hypothesis 4.5 is perfectly satisfied, then $\widetilde{\rho}_l = 1$. Conversely, the lowest possible energy concentration corresponds to $\widetilde{\rho}_l = 1/(2m^2)$, *i.e.*, $\widetilde{\rho}_l = 1/32$ for AlexNet and $\widetilde{\rho}_l = 1/8$ for ResNet.

maximum percentage of energy within a square window of size $\kappa \times \kappa$ in the Fourier domain:

$$\widetilde{\rho}_l := \frac{\max_{\boldsymbol{\theta} \in [-\pi,\,\pi]^2} \left\| \mathbb{1}_{B_\infty(\boldsymbol{\theta},\,\kappa/2)} \widehat{\widetilde{\mathrm{W}}}_l \right\|_{L^2}^2}{\left\| \widehat{\widetilde{\mathrm{W}}}_l \right\|_{L^2}^2}, \tag{5.18}$$

where the l^∞-ball $B_\infty(\boldsymbol{\theta},\, \kappa/2)$ is defined in the quotient space $[-\pi,\, \pi]^2 / (2\pi\mathbb{Z}^2)$, as explained in Remark 4.2. We denote by $\boldsymbol{\theta}_l$ the *characteristic frequency* of $\widetilde{\mathrm{W}}_l$, for which the maximum value in (5.18) is reached:

$$\boldsymbol{\theta}_l := \operatorname{argmax}_{\boldsymbol{\theta} \in [-\pi,\,\pi]^2} \left\| \mathbb{1}_{B_\infty(\boldsymbol{\theta},\,\kappa/2)} \widehat{\widetilde{\mathrm{W}}}_l \right\|_{L^2}^2. \tag{5.19}$$

The statistical distribution of $(\widetilde{\rho}_l)_{k \in \mathcal{G}}$ on the one hand (Gabor channels), and $(\widetilde{\rho}_l)_{l \in \{0..63\} \setminus \mathcal{G}}$ on the other hand, are shown in Figure 5.4, for AlexNet and ResNet after training with ImageNet-1K. The window size κ has been set to its highest admissible value, *i.e.*, π/m.

Gabor channels exhibit a substantial percentage of energy outside the window of interest, particularly in the case of AlexNet. Such behavior can make the networks more unstable, since the conditions of Section 4.5 are no longer satisfied. This observation can be attributed to the kernel size imposed by the network's design, which mechanically sets a limit on the Fourier resolution. In Figure 5.2, we can indeed notice that some Gabor-like filters seem truncated, providing a visual evidence for the previous statement. However, increasing the kernel size would make the networks harder to train, with possible downsides on their prediction accuracy. Nevertheless, consistent with Hypothesis 4.5, the Fourier resolution of the Gabor-like kernels is higher in AlexNet, compared to ResNet, as the subsampling factor m is also twice larger. This is visually evidenced in Figure 5.2.

In Section 5.4.2, we shall construct a mathematical twin of standard CNNs, where Hypothesis 4.4 (monochrome filters) and Hypothesis 4.5 (Gabor-like filters with maximum bandwidth) are enforced for a predefined number of output channels.

5.4 Design of the ℂMod-based Antialiased Models

In this section, we describe the general principles of our antialiasing approach based on complex convolutions. We then provide some details about the mathematical twin based on DT-ℂWPT, and explain how our method has been benchmarked against blur-pooling-based antialiased models.

5.4.1 Antialiasing Principle

In this section, we focus on the Gabor channels $\mathcal{G} \subset \{0\,..\,L-1\}$, as described in Section 5.3.1. The main idea is to substitute, for any $l \in \mathcal{G}$, ℝMax by ℂMod, as explained hereafter. Following (5.2), expression (5.7) can be rewritten

$$\mathrm{A}_l^{\max} = \mathrm{ReLU}(\mathrm{Y}_l^{\max} + b_l), \tag{5.20}$$

where $\mathrm{Y}_l^{\max}$ is the output of an ℝMax operator as introduced in (5.4). More formally,

$$\mathrm{Y}_l^{\max} := \mathrm{MaxPool}\left(\sum_{k=0}^{K-1} (\mathrm{X}_k \star \mathrm{V}_{lk}) \downarrow m\right). \tag{5.21}$$

Then, following (5.3), the ℝMax-ℂMod substitution yields

$$\mathrm{A}_l^{\mathrm{mod}} = \mathrm{ReLU}(\mathrm{Y}_l^{\mathrm{mod}} + b_l), \tag{5.22}$$

where $\mathrm{Y}_l^{\mathrm{mod}}$ is the output of a ℂMod operator (5.5), satisfying

$$\mathrm{Y}_l^{\mathrm{mod}} := \left|\sum_{k=0}^{K-1} (\mathrm{X}_k \star \mathrm{W}_{lk}) \downarrow (2m)\right|, \tag{5.23}$$

where the complex-valued filter $\mathrm{W}_{lk} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ has been introduced in (5.12). Then, assuming Hypotheses 4.4 and 4.5 for any $l \in \mathcal{G}$, the stability results of Section 4.5 are applicable to the Gabor channels. In particular, for any input channel $k \in \{0\,..\,K-1\}$, the energy of W_{lk} is concentrated in a small window in the Fourier domain, similar to Figure 5.1d. Due to this property, the modulus operator provides a smooth envelope for complex-valued cross-correlations with W_{lk} (Kingsbury and Magarey, 1998). This leads to the output $\mathrm{Y}_l^{\mathrm{mod}}$ (5.23) being nearly invariant to translations. Furthermore, $\mathrm{Y}_l^{\max} \approx \mathrm{Y}_l^{\mathrm{mod}}$ (except for a set of pathological frequencies regularly scattered across the Fourier domain), thus establishing the ℂMod operator as a stable alternative to ℝMax.

As covered in Section 5.3.1, the Fourier resolution of the Gabor-like filters does not quite match the requirements needed to consider Hypothesis 4.5 as a reasonable approximation, especially for AlexNet. This is partly due to the limited size of the convolution kernels, which leads to truncated filters. However, in the next section, we present a mathematical twin of standard, freely-trained architectures, in which both hypotheses are enforced.

5.4.2 Wavelet-Based Twin Models (WCNNs)

As explained in Section 5.4.1, our antialiasing method restricts to the Gabor channels $l \in \mathcal{G} \subset \{0\,..\,L-1\}$. However, $\mathcal{G}$ is unknown a priori: for a given output channel

$l \in \{0\,..\,L-1\}$, whether V_{lk} will become band-pass and oriented after training is unpredictable. Thus, we need a way to automatically separate the set $\mathcal{G}$ of Gabor channels from the set of remaining channels, denoted by $\mathcal{F} := \{0\,..\,L-1\} \setminus \mathcal{G}$. To this end, we built "mathematical twins" of standard CNNs, based on the dual-tree wavelet packet transform (DT-$\mathbb{C}$WPT), such as described in Section 3.3. These models, which we call WCNNs in short, are intended to reproduce the behavior of freely-trained architectures with a higher degree of control and fewer trainable parameters. In this section, we present their general structure; a more detailed description is provided in Section 5.A.1. For the purpose of readability, we assume that $K = 3$ (RGB input images).

We denote by $L_{\mathrm{gab}} := \mathrm{card}(\mathcal{G})$ and $L_{\mathrm{free}} := \mathrm{card}(\mathcal{F})$ the number of Gabor and remaining channels, respectively. They are determined empirically from the trained CNNs, following the procedure described in Section 5.3.1—see Table 5.1 for a summary of the experiment details. In a twin WCNN architecture, the two groups of output channels are organized such that $\mathcal{F} = \{0\,..\,L_{\mathrm{free}}-1\}$ and $\mathcal{G} = \{(L_{\mathrm{free}}+1)\,..\,L\}$. The first L_{free} channels, which are outside the scope of our antialiasing approach, remain freely-trained as in the standard architecture. Regarding the L_{gab} remaining (Gabor) channels, the convolution kernels V_{lk} are constrained to satisfy Hypotheses 4.4 and 4.5, as explained below.

Monochrome Filters. In WCNNs, Hypothesis 4.4 (monochrome filters) is enforced with a trainable 1×1 convolution layer (M. Lin et al., 2014), parameterized by $\boldsymbol{\mu}$, computing the following luminance image:

$$\mathrm{X}^{\mathrm{lum}} := \sum_{k=0}^{2} \mu_k \mathrm{X}_k. \tag{5.24}$$

Gabor-Like Kernels. To guarantee that Hypothesis 4.5 is met (Gabor-like kernels with bandwidth no larger than π/m), we implemented DT-$\mathbb{C}$WPT, which is achieved through a series of subsampled convolutions. The number of decomposition stages $J \in \mathbb{N} \setminus \{0\}$ was chosen such that

$$m = 2^{J-1}, \tag{5.25}$$

where, as a reminder, m denotes the subsampling factor as introduced in (5.6). As proven in Proposition 4.8, DT-$\mathbb{C}$WPT generates a set of filters $(\mathrm{W}_{k'}^{\nearrow(J)}, \mathrm{W}_{k'}^{\searrow(J)}, \mathrm{W}_{k'}^{\swarrow(J)}, \mathrm{W}_{k'}^{\nwarrow(J)})_{k' \in \{0..4^J-1\}}$, which tiles the Fourier domain $[-\pi,\, \pi]^2$ into 4×4^J overlapping square windows of size π/m. Their real and imaginary parts approximately form a 2D Hilbert transform pair such as defined in (5.11). For the sake of convenience, this family of filters is re-indexed and renamed $(\mathrm{W}_{k'}^{\mathrm{dt}})_{k' \in \{0..4\times 4^J-1\}}$.

The WCNN architecture is designed such that, for any Gabor channel $l \in \mathcal{G}$,

$$\widetilde{\mathrm{V}}_l = \mathrm{Re}(\widetilde{\mathrm{W}}_l), \tag{5.26}$$

where $\widetilde{\mathrm{W}}_l \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ is selected among the 4×4^J DT-$\mathbb{C}$WPT filters:

$$\exists k' \in \{0\,..\,4 \times 4^J - 1\} : \widetilde{\mathrm{W}}_l := \mathrm{W}_{k'}^{\mathrm{dt}}. \tag{5.27}$$

The output Y_l introduced in (5.6) then becomes

$$\mathrm{Y}_l = (\mathrm{X}^{\mathrm{lum}} \star \widetilde{\mathrm{V}}_l) \downarrow 2^{J-1}. \tag{5.28}$$

To summarize, rather than the freely-trained convolution (5.6), a WCNN employs a combination of (5.24) and (5.28) as a substitute, for any Gabor output channel $l \in \mathcal{G}$. This combination is wrapped into a *wavelet block*, also referred to as WBlock in short. Technical details about its exact design are provided in Section 5.A.1. The resulting convolution kernel then satisfies $\mathrm{V}_{lk} = \mu_k \widetilde{\mathrm{V}}_l$, where $\widetilde{\mathrm{V}}_l$ has been introduced in (5.26).

Schematic representations of the WCNN architecture based on AlexNet ($m = 4$, $J = 3$) and ResNet ($m = 2$, $J = 2$) are provided in Figures 5.5b and 5.6b, respectively (top).

5.4.3 Antialiased WCNNs with ℂMod

Using the antialiasing principles presented in Section 5.4.1, we replace ℝMax (5.21) by ℂMod (5.23) for all Gabor channels $l \in \mathcal{G}$. In the corresponding model, referred to as ℂWCNN, the wavelet block is replaced by a *complex wavelet block* (ℂWBlock), in which (5.28) becomes

$$\mathrm{Z}_l = (\mathrm{X}^{\mathrm{lum}} \star \widetilde{\mathrm{W}}_l) \downarrow 2^{J}, \tag{5.29}$$

where $\widetilde{\mathrm{W}}_l \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ has been introduced in (5.27). Then, a modulus is applied to Z_l, which yields $\mathrm{Y}_l^{\mathrm{mod}}$ such as defined in (5.23), with $\mathrm{W}_{lk} := \mu_k \widetilde{\mathrm{W}}_l$ for any RGB channel $k \in \{0\,..\,2\}$. Finally, we apply a bias and ReLU to $\mathrm{Y}_l^{\mathrm{mod}}$, following (5.22).

A schematic representation ℂWAlexNet is provided in Figure 5.5c (top).

Remark 5.2. In contrast with (5.12), the 2D Hilbert transform does not need to be explicitly computed to get the complex-valued kernel W_{lk}, since it is directly provided by DT-ℂWPT. Nonetheless, by design, $\mathrm{Re}(\mathrm{W}_{lk})$ and $\mathrm{Im}(\mathrm{W}_{lk})$ approximately form a Hilbert transform pair.

Remark 5.3. Unlike freely-trained models, the size of the convolution kernels is not limited, thus allowing Hypothesis 4.5 to be more accurately satisfied, to the extent allowed by DT-ℂWPT. Similar to Figure 5.4, the statistical distribution of $(\widetilde{\rho}_l)_{k \in \mathcal{G}}$ on the one hand (Gabor channels), and $(\widetilde{\rho}_l)_{l \in \mathcal{F}}$ on the other hand, are shown in Figure 5.14.

Remark 5.4. The constraint (5.24) could be relaxed by authorizing a specific luminance vector $\boldsymbol{\mu}_l$ for each Gabor channel $l \in \mathcal{G}$, while still satisfying Hypotheses 4.4 and 4.5. Numerical experiments on this matter are left outside the scope of this thesis, but the corresponding (untrained) models are available in the Python package released on GitHub.

5.4.4 WCNNs with Blur Pooling

We benchmark our approach against the antialiasing methods proposed by R. Zhang (2019) and X. Zou et al. (2023). To this end, we first consider a WCNN antialiased with static or adaptive blur pooling, respectively referred to as BlurWCNN and ABlurWCNN. A schematic representation of BlurWAlexNet is provided in Figure 5.5b (bottom). Then, we substitute the blurpooled Gabor channels (right branch of the diagram) with our own ℂMod-based approach. The corresponding models are respectively referred to as ℂBlurWCNN and ℂABlurWCNN. Again, a schematic representation of ℂBlurWAlexNet

	WAlexNet	**WResNet**
m (subsampling factor)	4	2
J (decomposition depth)	3	2
L_{free}, L_{gab} (output channels)	32, 32	40, 24

Table 5.1. Experimental settings for our WCNN twin models. Other details are provided in Section 5.A.3.

can be found in Figure 5.5c (bottom). Note that, for a fair comparison, all models use blur pooling in the freely-trained channels as well as deeper layers. Therefore, several antialiasing methods are employed in different parts of the network.

5.4.5 Adaptation to ResNet: Batch Normalization

In many architectures including ResNet, the bias is computed after an operation called *batch normalization* (BN) (Ioffe and Szegedy, 2015). In this context, (5.1) becomes

$$\text{Conv} \to \text{Sub} \to \text{BN} \to \text{Bias} \to \text{ReLU} \to \text{MaxPool}. \tag{5.30}$$

As detailed in Section 5.A.2, the $\mathbb{R}$Max-$\mathbb{C}$Mod substitution yields, analogously to (5.3),

$$\mathbb{C}\text{Conv} \to \text{Sub} \to \text{Modulus} \to \text{BN0} \to \text{Bias} \to \text{ReLU}, \tag{5.31}$$

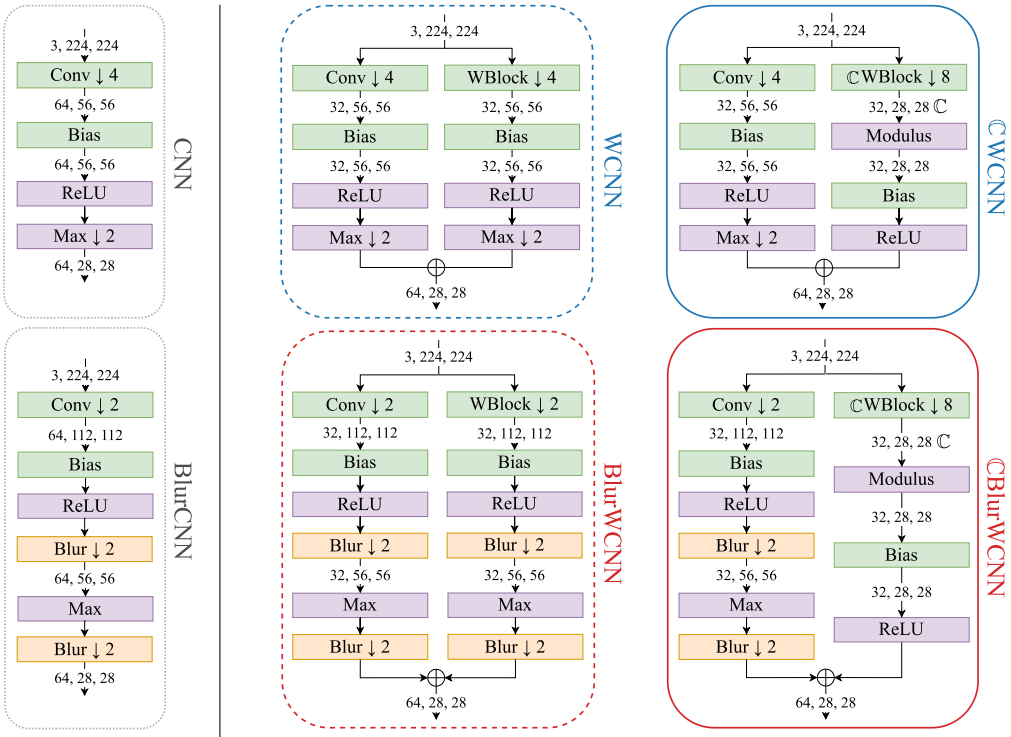
where BN0 refers to a special type of batch normalization without mean centering. A schematic representation of $\mathbb{C}$WResNet models is provided in Figure 5.6c (top).

5.5 Experiments

5.5.1 Experiment Details

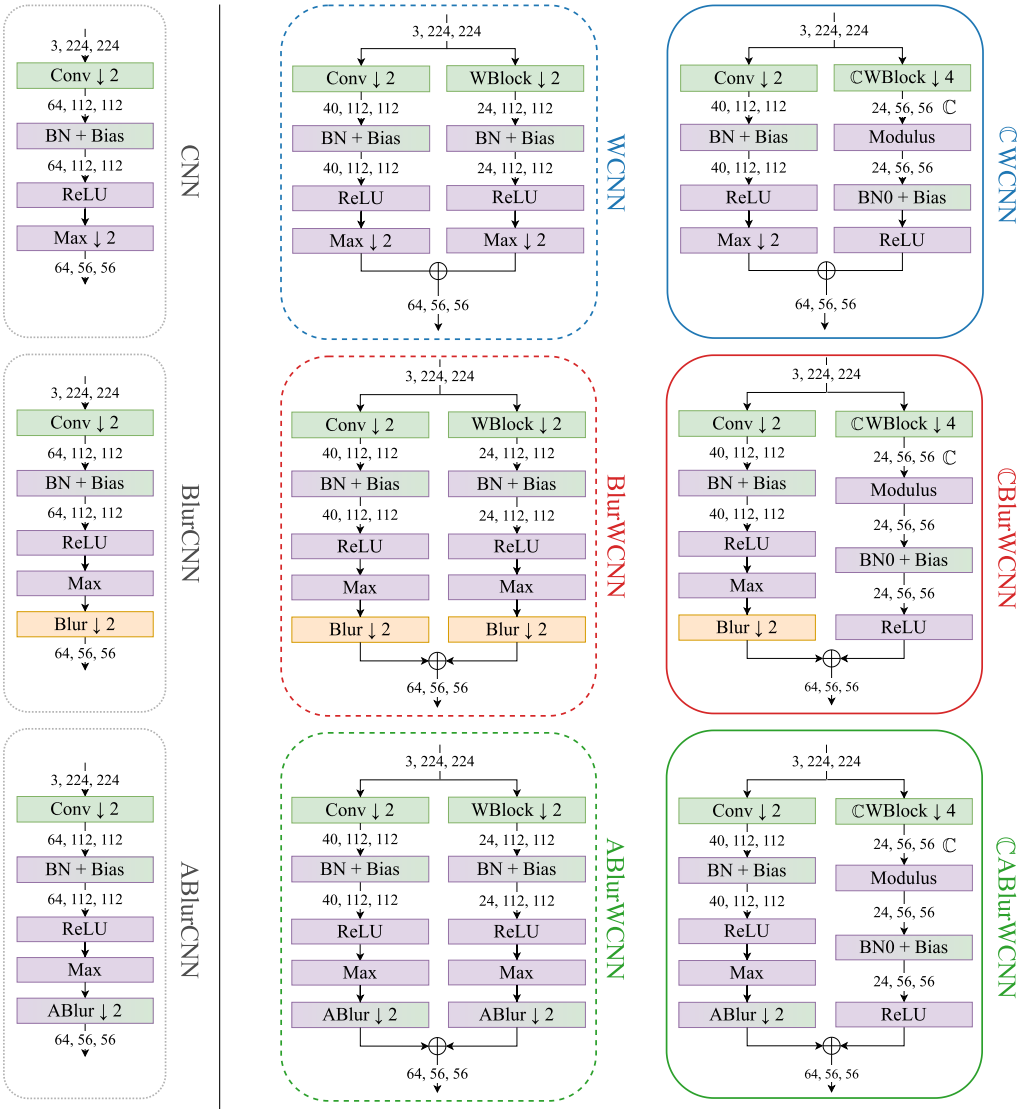
ImageNet. We built our WCNN and $\mathbb{C}$WCNN mathematical twins based on AlexNet (Krizhevsky et al., 2017) and ResNet-34 (He et al., 2016). Their overall design is described in Section 5.4, along with setting details in Table 5.1. The values of L_{free} and L_{gab} were determined empirically from the freely-trained AlexNet and ResNet-34, following the procedure described in Section 5.3.1. More precisely, we respectively found 31 and 23 Gabor channels in AlexNet and ResNet, that we rounded to 32 and 24 to ease implementation. Zhang's static blur pooling approach (2019) is tested on both AlexNet and ResNet, whereas Zou et al.'s adaptive approach (2023) is only tested on ResNet. The latter was indeed not implemented on AlexNet in the original paper, and we could not make it work on this architecture.

As mentioned above, we compare blur-pooling-based antialiasing approach (Figure 5.5b, bottom) with ours (Figure 5.5c, bottom). To apply static or adaptive blur pooling to the WCNNs, we proceed as follows. Following Zhang's implementation, the wavelet block is not antialiased if $m = 2$ as in ResNet, for computational reasons. However, when $m = 4$ as in AlexNet, a blur pooling layer is placed after ReLU, and the wavelet block's subsampling factor is divided by 2. Moreover, max pooling is replaced by max-blur pooling. The size of the blurring filters is set to 3, as recommended by R. Zhang (2019). Besides, DT-$\mathbb{C}$WPT decompositions are performed with Q-shift orthogonal filters of length 10 as introduced by Kingsbury (2003).



(a) AlexNet (b) WAlexNet (baseline) (c) **ℂWAlexNet**

Figure 5.5. First layers of AlexNet and its variants, corresponding to a convolution layer followed by ReLU and max pooling (5.1). The models are framed according to the same colors and line styles as in Figures 5.9a and 5.10a. The green modules are the ones containing trainable parameters; the orange and purple modules represent static linear and nonlinear operators, respectively. The numbers between each module represent the depth (number of channels), height and width of each output. Figure 5.5a: freely-trained models. Top: standard AlexNet. Bottom: Zhang's "blurpooled" AlexNet. Figure 5.5b: mathematical twins (WAlexNet) reproducing the behavior of standard (top) and blurpooled (bottom) AlexNet. The left side of each diagram corresponds to the $L_{\text{free}} := 32$ freely-trained output channels, whereas the right side displays the $L_{\text{gab}} := 32$ remaining channels, where freely-trained convolutions have been replaced by a wavelet block (WBlock) as described in Section 5.4.2. Figure 5.5c: ℂMod-based antialiased WAlexNet, where WBlock has been replaced by ℂWBlock, and max pooling by a modulus. The bias and ReLU are placed after the modulus, following (5.3). In the bottom models, we compare Zhang's antialiasing approach (Figure 5.5b) with ours (Figure 5.5c) in the Gabor channels.



(a) ResNet (b) WResNet (baseline) (c) ℂ**WResNet**

Figure 5.6. First layers of ResNet and its variants, corresponding to a convolution layer followed by ReLU and max pooling. The models are framed according to the same colors and line styles as in Figures 5.9b and 5.10b. The bias module from Figure 5.5 has been replaced by an affine batch normalization layer ("BN + Bias", or "BN0 + Bias" when placed after Modulus—see Section 5.4.5). Top: ResNet without blur pooling. Middle: Zhang's approach, using static blur pooling. Bottom: Zou et al.'s approach, using adaptive blur pooling.

As for standard AlexNet and ResNet, our models were trained on the ImageNet-1K dataset, following the standard procedure provided by PyTorch. Moreover, we set aside 100K images from the training set—100 per class—in order to compute the top-1 error rate after each training epoch ("validation set").

CIFAR-10. We also trained ResNet-18, ResNet-34 and their variants on the CIFAR-10 dataset. Training was performed on 300 epochs, with an initial learning rate equal to 0.1, decreased by a factor of 10 every 100 epochs. As for ImageNet, we set aside 5 000 images out of 50K to compute accuracy during the training phase. Given the images of small size in this dataset (32×32 pixels), feature extraction can be performed efficiently with a reduced number of layers. For this reason, the first layers (5.1) arguably have a higher influence on the overall predictive power. We therefore expect to clearly highlight the benefits of our approach on this specific task.

5.5.2 Evaluation Metrics

Classification Accuracy. Classification accuracy was computed on the standard ImageNet evaluation set (50K images). We followed the *ten-crops* procedure (Krizhevsky et al., 2017): predictions are made over 10 patches extracted from each input image, and the softmax outputs are averaged to get the overall prediction. We also considered center crops of size 224 for *one-crop* evaluation. In both cases, we used top-1-5 error rates. For CIFAR-10 evaluation (10K images), we measured the top-1 error rate with one- and ten-crops.

Measuring Shift Invariance. For each image in the ImageNet evaluation set, we extracted several patches of size 224, each of which being shifted by 0.5 pixel along a given axis. We then compared their outputs in order to measure the model's robustness to shifts. This was done by computing the Kullback-Leibler (KL) divergence between output vectors—which, under certain hypotheses, can be interpreted as probability distributions (Bishop and Mitchell, 2014, pp. 205-206). This metric is intended for visual representation.

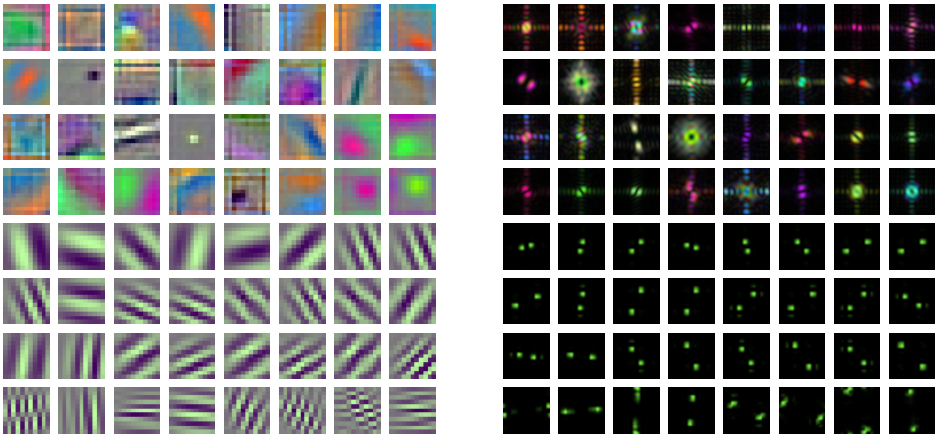
In addition, we measured the mean flip rate (mFR) between predictions (Hendrycks and Dietterich, 2019), as done by R. Zhang (2019) in its blurpooled models. For each direction (vertical, horizontal and diagonal), we measured the mean frequency upon which two shifted input images yield different top-1 predictions, for shift distances varying from 1 to 8 pixels. We then normalized the results with respect to AlexNet's mFR, and averaged over the three directions. This metric is also referred to as *consistency*.

We repeated the procedure for the models trained on CIFAR-10. This time, we extracted patches of size 32×32 from the evaluation set, and computed mFR for shifts varying from 1 to 4 pixels. Besides, normalization was performed with respect to ResNet-18's mFR.

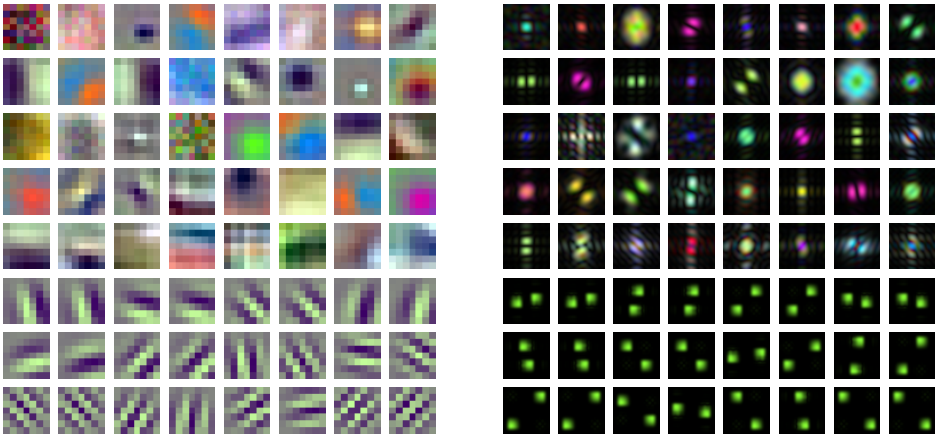
5.6 Results and Discussion

5.6.1 Kernel Visualization and Characteristic Frequencies

Figure 5.7 displays the kernels $\mathbf{V} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)^{L \times K}$, with $K = 3$ and $L = 64$, for WAlexNet and WResNet. The kernels are shown as RGB color images, after training with ImageNet,



(a) WAlexNet (32 Gabor channels)

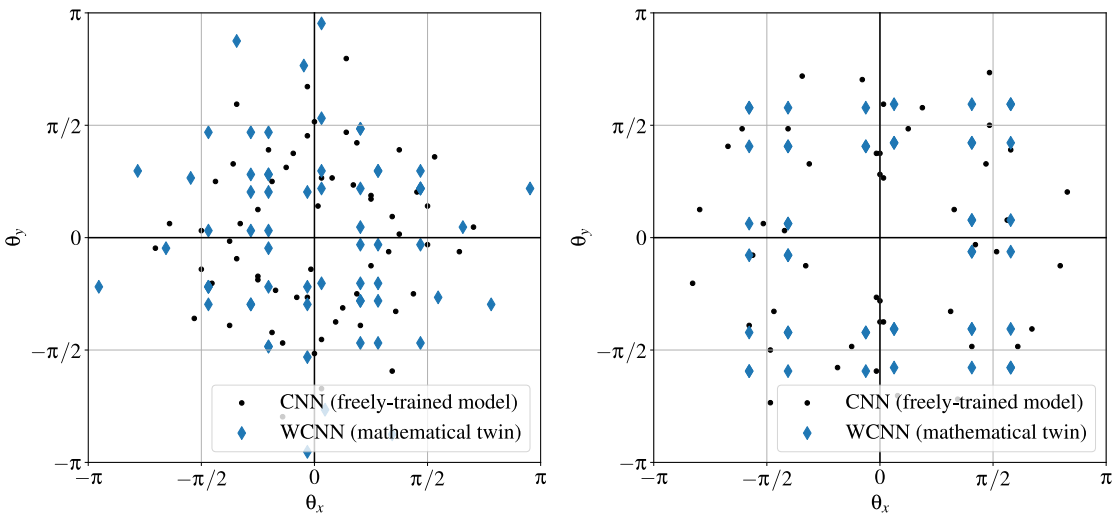


(b) WResNet (24 Gabor channels)

Figure 5.7. Convolution kernels in the first layer of WAlexNet (a) and WResNet-34 (b), after training with ImageNet-1K. For each output channel $l \in \{0\,..\,63\}$, the corresponding convolution kernel $(\mathrm{V}_{lk})_{k\in\{0..2\}}$ is displayed as an RGB image in the spatial domain (left), and its associated magnitude spectrum in the Fourier domain (right). The convolution kernels associated to the Gabor channels are displayed on the 4 and 3 last rows for WAlexNet and WResNet, respectively. For the sake of visual rendering, they have been respectively cropped to (11×11) and (7×7) to match the size of the freely-trained kernels.

for both freely-trained and Gabor channels. This figure is to be compared with Figure 5.2 for the freely-trained models. Furthermore, the characteristic frequencies $(\boldsymbol{\theta}_l)_{l\in\mathcal{G}}$, such as defined in (5.19), are plotted in Figure 5.8, together with the characteristic frequencies of the freely-trained models.

From Figure 5.7, we observe that the Fourier resolution of V_{lk} increases with the subsampling factor m. This property is consistent with what is observed in freely-trained CNNs, as shown in Figure 5.2: in AlexNet, where $m = 4$, the Gabor-like filters are more localized in frequency (and less spatially localized) than in ResNet, where $m = 2$. Furthermore, we notice that, up to a few exceptions, the freely-trained channels (4 and 5 first rows for AlexNet and ResNet, respectively) have been specialized to lower-frequency



(a) AlexNet (b) ResNet-34

Figure 5.8. Two-dimensional characteristic frequencies $\boldsymbol{\theta}_l \in [-\pi, \pi]^2$ (5.19), for any Gabor channel $l \in \mathcal{G}$. For the sake of clarity, the plots have been symmetrized with respect to the origin.

filters (mono- or bi-color blobs). Finally, from Figure 5.8, we observe that the point clouds representing the characteristic frequencies of CNN and WCNN architectures are similarly distributed across the Fourier domain. However, unlike CNNs, the bandwidth in WCNNs is generally well-contained within a square window of size $\kappa \times \kappa$, with $\kappa := \pi/m$ (Hypothesis 4.5). This property is evidenced in Figure 5.14, in contrast with Figure 5.4 for the freely-trained models.

5.6.2 Validation and Test Accuracy

Top-1 accuracy of AlexNet- and ResNet-based models along training with ImageNet are plotted in Figure 5.9. In addition, error rates, computed on the evaluation sets, are provided in Table 5.2 for ImageNet and Table 5.3 for CIFAR-10.

Our $\mathbb{C}$Mod-based approach significantly outperforms the baselines for AlexNet: $\mathbb{C}$WCNN vs WCNN (blue diamonds), and $\mathbb{C}$BlurWCNN vs BlurWCNN (red stars). Remarkably, the exclusive application of $\mathbb{C}$Mod-based antialiasing to the Gabor channels ($\mathbb{C}$WCNN, solid blue line) is sufficient to match the performance of blur-pooling-based antialiasing (BlurWCNN, dashed red line), which, in contrast, is implemented throughout the entire network. On the other hand, the ResNet validation curves (Figure 5.9b) indicate an improvement of $\mathbb{C}$WCNN over WCNN (blue diamonds, solid versus dashed lines), but marginal to no gains of $\mathbb{C}$BlurWCNN over BlurWCNN (red stars), or $\mathbb{C}$ABlurWCNN versus ABlurWCNN (green squares), when trained on ImageNet. Yet, notable gains are observed on the evaluation set for $\mathbb{C}$BlurWCNN versus BlurWCNN, as reported in Table 5.2. This indicates a better generalization capability of our approach, compared to the static blur pooling-based method. In particular, simply replacing the blurpooled Gabor channels in the first layer with our $\mathbb{C}$Mod-based approach (BlurWCNN versus $\mathbb{C}$BlurWCNN) produces improvements on a similar order of magnitude as adaptive blur pooling, which is applied throughout the whole network (ABlurWCNN). However, adap-

Model	One-crop		Ten-crops		Shifts
	top-1	top-5	top-1	top-5	mFR
	AlexNet				
CNN	*45.3*	*22.2*	*41.3*	*19.3*	*100.0*
WCNN	44.9	21.8	40.8	19.0	101.4
ℂWCNN*	**44.3**	**21.3**	**40.2**	**18.5**	**88.0**
BlurCNN†	*44.4*	*21.6*	*40.7*	*18.7*	*63.8*
BlurWCNN†	44.3	21.4	40.5	18.5	**63.1**
ℂBlurWCNN†*	**43.3**	**20.5**	**39.6**	**17.9**	69.4
	ResNet-34				
CNN	*27.6*	*9.2*	*24.8*	*7.7*	*78.1*
WCNN	27.4	9.2	24.7	7.6	77.2
ℂWCNN*	**27.2**	**9.0**	**24.4**	**7.4**	**73.1**
BlurCNN†	*26.7*	*8.6*	*24.0*	*7.2*	*61.2*
BlurWCNN†	26.7	8.6	24.1	7.3	65.2
ℂBlurWCNN†*	**26.5**	**8.4**	**23.7**	**7.0**	**62.5**
ABlurCNN‡	*26.1*	*8.3*	*23.5*	*7.0*	*60.8*
ABlurWCNN‡	**26.0**	**8.2**	**23.6**	**6.9**	**62.1**
ℂABlurWCNN‡*	26.1	**8.2**	23.7	7.0	63.1

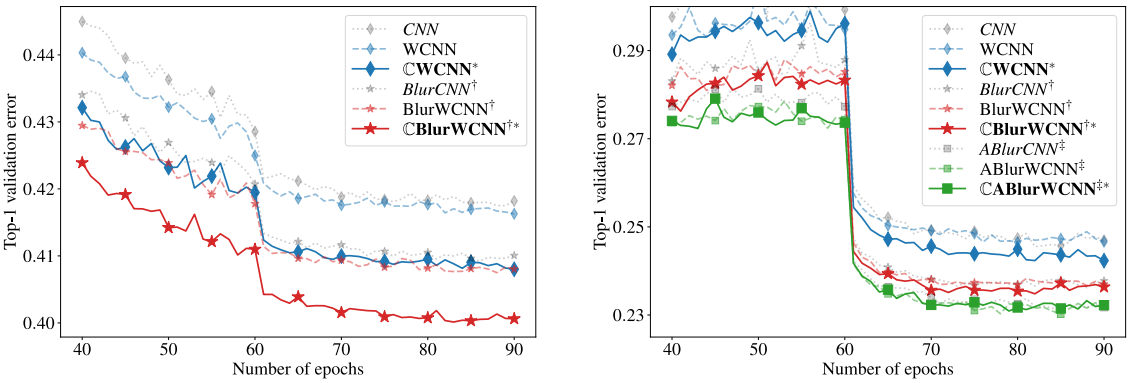
Table 5.2. Evaluation metrics on ImageNet (%): the lower the better. Models: †static and ‡adaptive blur pooling; *ℂMod-based antialiasing (our approach).

Model	ResNet-18			ResNet-34		
	1crp	10crp	shft	1crp	10crps	shft
CNN	*14.9*	*10.8*	*100.0*	*15.2*	*10.9*	*100.3*
WCNN	14.2	10.3	92.4	14.5	10.5	99.2
ℂWCNN*	**13.8**	**9.6**	**88.8**	**12.9**	**9.2**	**93.0**
BlurCNN†	*14.2*	*10.4*	*87.7*	*15.7*	*11.6*	*88.2*
BlurWCNN†	13.1	9.7	**84.6**	13.2	9.9	85.6
ℂBlurWCNN†*	**12.3**	**8.9**	85.7	**12.4**	**9.1**	**83.7**
ABlurCNN‡	*14.6*	*11.0*	*90.9*	*16.3*	*12.8*	*91.9*
ABlurWCNN‡	14.5	11.0	86.5	14.0	10.4	93.3
ℂABlurWCNN‡*	**12.8**	**9.7**	**81.7**	**12.8**	**9.2**	**86.6**

Table 5.3. Evaluation metrics on CIFAR-10 (%): top-1 error rate using one- and ten-crops methods ("1crp" and "10crp"); and mFR measuring consistency ("shft"). Models: †static and ‡adaptive blur pooling; *ℂMod-based antialiasing (our approach).

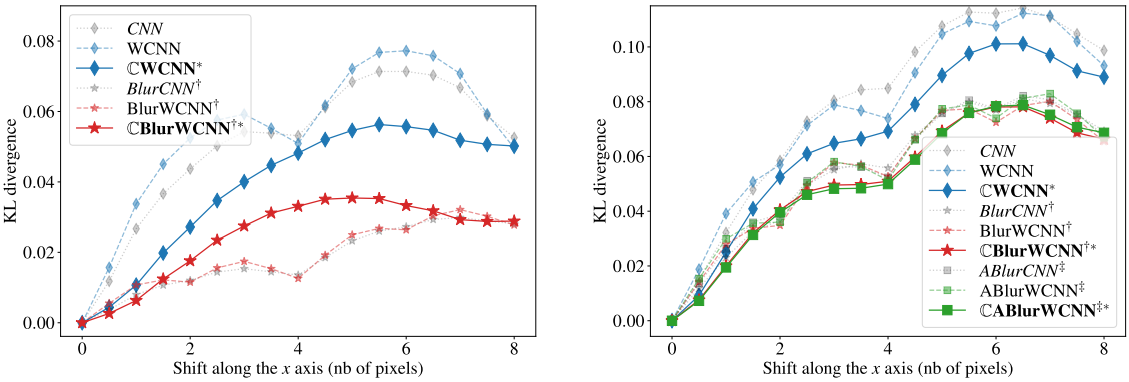
Model	One-crop		Ten-crops		Shifts
	top-1	top-5	top-1	top-5	mFR
WCNN	27.4	9.2	24.7	7.6	77.2
BlurWCNN†	26.7	8.6	24.1	7.3	65.2
ℂBlurWCNN†*	**26.5**	**8.4**	**23.7**	**7.0**	**62.5**
→ BlurWCNN†◇	26.8	8.6	24.3	7.1	69.7
ABlurWCNN‡	**26.0**	**8.2**	**23.6**	**6.9**	**62.1**
ℂABlurWCNN‡*	26.1	**8.2**	23.7	7.0	63.1
→ ABlurWCNN†◇	26.4	8.4	23.9	7.0	70.3

Table 5.4. Ablation study on WResNet-34, described in Section 5.6.6. Evaluation metrics on ImageNet (%): the lower the better. Models: †static and ‡adaptive blur pooling; *ℂMod-based antialiasing on the Gabor channels (our approach); ◇ablated model: no antialiasing on the Gabor channels (neither blur pooling nor ℂMod).



(a) AlexNet (b) ResNet-34

Figure 5.9. Evolution of the top-1 validation error (one-crop) along training with ImageNet. The freely-trained models, upon which the mathematical twins are built, appear in faint gray. Legend: †static and ‡adaptive blur pooling; *ℂMod-based antialiasing (our approach).



(a) AlexNet (b) ResNet-34

Figure 5.10. AlexNet-based models: mean KL divergence between the outputs of a reference image versus shifted images (the smaller the better). Legend: †static and ‡adaptive blur pooling; *ℂMod-based antialiasing (our approach).

tive blur pooling (ABlurWCNN), yields similar or marginally higher accuracy than our approach (ℂABlurWCNN). Nevertheless, our method is computationally more efficient, requires less memory (see Section 5.6.5 for more details), and do not demand additional training, unlike adaptive blur pooling. To better support this claim, we conducted ablation studies which we present in Section 5.6.6. Finally, when trained on CIFAR-10 (see Table 5.3), our ℂMod-based antialiased models built upon ResNet-18 and 34 achieve significant gains in accuracy over non-antialiased models, as well as models antialiased with both blur-pooling-based methods.

Arguably, the higher gains obtained for AlexNet, compared to ResNet, are attributed to the higher impact of the initial layers on the network's predictive power. This is partly due to the larger subsampling factor performing dimensionality reduction of higher order, which may capture more complex dependencies in the structure of the input data. Additionally, ResNet models are deeper than AlexNet, and much of their performance is

derived from non-geometric features captured by the deeper layers (Oyallon et al., 2017). As a result, our antialiasing approach may have a comparatively smaller impact on the network's overall accuracy. Concerning CIFAR classification, the first layers have a high impact on the network's predictive power, due to the small size of input images. Therefore, our antialiasing method have a higher impact on accuracy, comparatively to ImageNet.

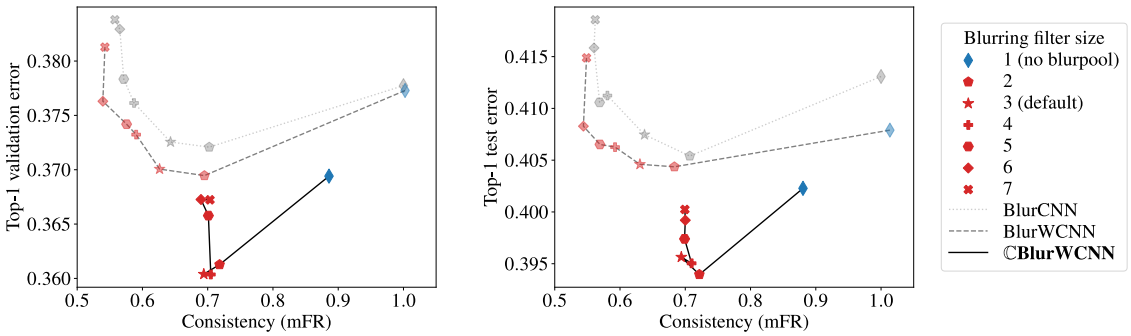
As a side note, the training curves for WCNNs (colored dashed lines) closely follow those of standard CNNs (gray dotted lines). This is an expected result, since the former models are designed to mimic the behavior of the latter.

5.6.3 Shift Invariance (KL Divergence)

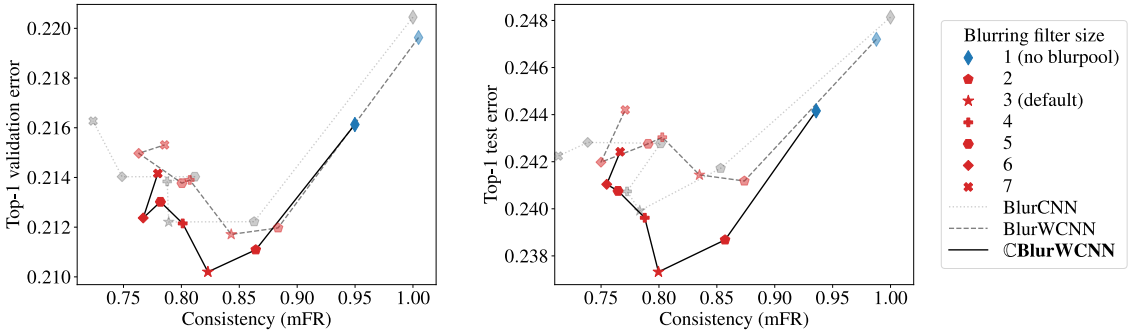
The mean KL divergence between outputs of shifted images are plotted in Figure 5.10 for AlexNet and ResNet-34 trained on ImageNet. Moreover, the mean flip rate for shifted inputs (consistency) is reported in Table 5.2 for ImageNet (AlexNet and ResNet-34) and Table 5.3 for CIFAR-10 (ResNet-18 and 34).

In both AlexNet and WAlexNet (Figure 5.10a, gray and blue diamonds, dashed lines), the initial convolution layer's output undergoes a one-pixel shift for every four-pixel shift in the input image. Consequently, any divergence between the output vectors is due to the instability of subsequent layers to one-pixel shifts. In contrast, instabilities which are accountable to the initial layer are observed for shifts that are *not* multiples of 4. Likewise, for input shifts of eight pixels, the max pooling's output is shifted by exactly one pixel, resulting in even higher stability. In $\mathbb{C}$WAlexNet (blue diamonds, solid line), the same eight-to-one-pixel ratio occurs to the modulus layer's output, which explains why the two curves meet for 8-pixel shifts. However, the $\mathbb{R}$Max-$\mathbb{C}$Mod substitution has greatly reduced first-layer instabilities, resulting in a flattened curve and avoiding the "bumps" observed for non-antialiased models. Similar observations can be drawn for ResNet (Figure 5.10b), with the caveat that input images must only be shifted by 2 and 4 pixels for a one-pixel shift at the output of the convolution and max pooling layers, respectively.

On the other hand, BlurAlexNet and BlurWAlexNet (Figure 5.10a, gray and red stars, dashed lines) exhibit considerably flattened curves, compared to non-antialiased models and $\mathbb{C}$WAlexNet. This demonstrates the effectiveness of Zhang's blur-pooling-based method in enhancing shift invariance. Applying $\mathbb{C}$Mod-based antialiasing instead of blur pooling on the Gabor channels ($\mathbb{C}$BlurWAlexNet, red stars, solid line) actually degrades shift invariance (except for shifts smaller than 1.5 pixels), as evidenced by the bell-shaped curve. Nevertheless, the corresponding classifier is significantly more accurate. This is not surprising, as our approach prioritizes the conservation of high-frequency details, which are important for classification. An extreme reduction of shift variance using a large blur pooling filter would indeed result in a significant loss of accuracy. Therefore, our work achieves a better balance between shift invariance and information preservation. Experimental details on this matter are provided in Section 5.6.4. However, we may notice that, applied to the ResNet architecture (Figure 5.10b), $\mathbb{C}$BlurWCNN (red stars, solid line) is generally more stable than BlurWCNN (dashed lines). The same observation car be drawn for the models based on adaptive blur pooling (green squares, solid vs dashed lines).



(a) AlexNet



(b) ResNet-34

Figure 5.11. Classification accuracy (ten-crops) vs consistency, measuring the stability of predictions to small input shifts, for (a) AlexNet- and (b) ResNet-based models (the lower the better for both axes). Left: validation set (100K images set aside from the training set). Right: evaluation set (50K images provided as a separate dataset). For each of the three architectures, we increased the blurring filter size from 1 (*i.e.*, no blur pooling) to 7. The blue diamonds (no blur pooling) and red stars (blur pooling with filters of size 3) correspond to the models from Figures 5.9 and 5.10. At equivalent consistency levels, our ℂMod-based approach (solid line) yields higher accuracy.

5.6.4 Accuracy vs Consistency

To gain further insights into the tradeoff between shift invariance and information preservation, we conducted experiments by varying the size of the blurring filters in AlexNet- and ResNet-based models. Figure 5.11 shows the relationship between consistency and prediction accuracy on ImageNet, for different filter sizes ranging from 1 (no blur pooling) to 7 (heavy loss of high-frequency information).[3] We found that a near-optimal tradeoff is achieved when the filter size is set to 2 or 3. Furthermore, at equivalent consistency levels, ℂBlurWCNN outperforms BlurWCNN in terms of accuracy. Note however that the optimal version of ℂBlurWCNN is not necessarily more consistent than the optimal version of BlurWCNN, despite achieving higher accuracy.

5.6.5 Computational Resources

Table 5.5 compares the computational resources and memory footprint required for each antialiasing method. To achieve this, we considered models with freely-trained convolu-

[3]Similar plots can be found in Zhang's paper (2019).

Method	Computational cost		Memory footprint	
	AlexNet	ResNet	AlexNet	ResNet
No antialiasing (ref)	*1.0*	*1.0*	*1.0*	*1.0*
BlurPool (R. Zhang, 2019)	4.0	1.0	4.7	1.9
ABlurPool (X. Zou et al., 2023)	–	2.1	–	2.0
ℂMod (ours)	**0.5**	**0.5**	**0.6**	**0.4**

Table 5.5. Computational cost and memory footprint required for each antialiasing method, per Gabor channel. The values are normalized relative to non-antialiased AlexNet or ResNet. Computational cost: FLOPs for computing $\mathrm{Y}_l^{\max}$ (5.21) or $\mathrm{Y}_l^{\mathrm{mod}}$ (5.23). Memory footprint: size of the intermediate and output tensors saved by PyTorch for the backward pass. It should be noted that the amount of computing resources required by the blur pooling layers in other areas of the network has not been considered in this analysis.

tions, because the goal was to evaluate the computational performances of the various antialiasing approaches, excluding implementation tricks based on DT-ℂWPT from the scope of analysis. More details are provided in Sections 5.B and 5.C.

5.6.6 Ablation Study

In Section 5.6.2, we demonstrated that our models outperform the baselines, except in cases where adaptive blur pooling is used as an antialiasing method. Applying our ℂMod-based approach on the Gabor channels slightly degrades accuracy compared to adaptive blur pooling, but it requires fewer computational resources and memory, as shown in Section 5.6.5. Following this idea, we want to determine whether antialiasing in the Gabor channels was necessary in the first place, given the fact that blur pooling is implemented throughout the entire network. To this end, we conducted an ablation study where antialiasing is removed from the Gabor channels, whereas static or adaptive blur pooling remain used in the rest of the network.

We found that, compared to blur-pooling- or ℂMod-based methods, removing antialiasing from the Gabor channels generally leads to decreased accuracy and consistency, as shown in Table 5.4. However, in the case of ABlurWResNet (adaptive blur pooling), this drop in performance is marginal, suggesting that antialiasing in this part of the network may not be essential. However, this approach requires using multiple adaptive blur pooling layers throughout the entire network, at the cost of additional computational resources, memory, and trainable parameters. Thus, although it yields the most significant gains in performance, adaptive blur-pooling-based models may not be the most suitable options in scenarios where computational resources are limited.

5.7 Concluding Remarks

The mathematical twins introduced in this chapter serve a proof of concept for our ℂMod-based antialiasing approach. However, its range of application extends well beyond DT-ℂWPT filters. While we focused on the first convolution layer, it is important to note that such initial layers play a critical role in CNNs by extracting low-level geometric features such as edges, corners or textures. Therefore, a specific attention is required for their design. In contrast, deeper layers are more focused on capturing high-level structures that conventional image processing tools are poorly suited for, as pointed out by Oyallon et al.

(2017) in the context of hybrid scattering networks.

Furthermore, while our approach is tailored for CNN architectures, which were chosen to make a fair comparison to related methods, it has potential for broader applicability. In particular, as covered in Section 6.2.4, it could be used to produce stable token embeddings for vision transformers. In conclusion, designing deep learning architectures to be invariant to certain groups of transformations can eliminate the need to learn such invariance from vast amounts of data. These architectures are therefore particularly suitable in situations where available data is scarce.

5.A Appendix: Technical Complements

5.A.1 Design of WCNNs

In this section, we provide complements to the description of the mathematical twin (WCNN) introduced in Section 5.4.2. Expressions (5.26) and (5.27) imply that, for each Gabor channel $l \in \mathcal{G}$, the average kernel $\widetilde{\mathrm{V}}_l$ is the real part of a DT-$\mathbb{C}$WPT filter:

$$\exists k' \in \{0\,..\,4 \times 4^J - 1\} : \widetilde{\mathrm{V}}_l = \mathrm{Re}(\mathrm{W}_{k'}^{\mathrm{dt}}), \tag{5.32}$$

where $J \in \mathbb{N} \setminus \{0\}$ denotes the decomposition depth, and $\mathrm{W}_{k'}^{\mathrm{dt}}$ is one of the 4×4^J filters spawned by DT-$\mathbb{C}$WPT. We now explain how the filter selection is done; in other words, how k' is chosen among $\{0\,..\,4 \times 4^J - 1\}$. Since input images are real-valued, we restrict to the filters with bandwidth located in the half-plane of positive x-values. For the sake of concision, we denote by $K_{\mathrm{dt}} := 2 \times 4^J$ the number of such filters.

For any RGB image $\mathbf{X} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)^3$, a luminance image $\mathrm{X}^{\mathrm{lum}} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ is computed following (5.24), using a 1×1 convolution layer. Then, DT-$\mathbb{C}$WPT is performed on $\mathrm{X}^{\mathrm{lum}}$. We denote by $\mathbf{D} := (\mathrm{D}_k)_{k \in \{0..K_{\mathrm{dt}}-1\}}$ the tensor containing the real part of the DT-$\mathbb{C}$WPT feature maps:

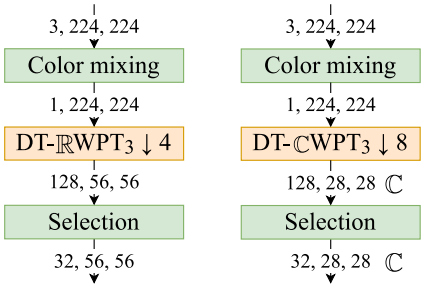
$$\mathrm{D}_k = (\mathrm{X}^{\mathrm{lum}} \star \mathrm{Re}\,\mathrm{W}_k^{(J)}) \downarrow 2^{J-1}. \tag{5.33}$$

For the sake of computational efficiency, DT-$\mathbb{C}$WPT is performed with a succession of subsampled separable convolutions and linear combinations of real-valued wavelet packet feature maps (I. W. Selesnick et al., 2005). To match the subsampling factor $m := 2^{J-1}$ (5.25) of the standard model, the last decomposition stage is performed without subsampling.

Filter Selection. The number of dual-tree feature maps K_{dt} may be greater than the number of Gabor channels L_{gab}. In that case, we therefore want to select filters that contribute the most to the network's predictive power. First, the low-frequency feature maps D_0 and $\mathrm{D}_{(4^J+1)}$ are discarded. Then, a subset of $K'_{\mathrm{dt}} < K_{\mathrm{dt}}$ feature maps is manually selected and permuted in order to form clusters in the Fourier domain. Considering a (truncated) permutation matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{K'_{\mathrm{dt}} \times K_{\mathrm{dt}}}$, the output of this transformation, denoted by $\mathbf{D}' \in l^2_{\mathbb{C}}(\mathbb{Z}^2)^{K'_{\mathrm{dt}}}$, is defined by:

$$\mathbf{D}' := \boldsymbol{\Sigma}\,\mathbf{D}. \tag{5.34}$$

The feature maps $\mathbf{D}'$ are then sliced into Q groups of channels $\mathbf{D}^{(q)} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)^{K_q}$, each of them corresponding to a cluster of band-pass dual-tree filters with neighboring frequencies and orientations. On the other hand, the output of the wavelet block, $\mathbf{Y}^{\mathrm{gab}} :=$



(a) WBlock $\downarrow 4$ (b) $\mathbb{C}$WBlock $\downarrow 8$

Figure 5.12. Detail of a wavelet block with $J = 3$ as in AlexNet, in its $\mathbb{R}$Max (a) and $\mathbb{C}$Mod (b) versions. DT-$\mathbb{R}$WPT corresponds to the real part of DT-$\mathbb{C}$WPT.

$(\mathrm{Y}_l)_{l\in\{L_{\text{free}}+1..L\}} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)^{L_{\text{gab}}}$, where Y_l has been introduced in (5.6), is also sliced into Q groups of channels $\mathbf{Y}^{(q)} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)^{L_q}$. Then, for each group $q \in \{0\,..\,Q-1\}$, an affine mapping between $\mathbf{D}^{(q)}$ and $\mathbf{Y}^{(q)}$ is performed. It is characterized by a trainable matrix $\boldsymbol{A}^{(q)} := (\boldsymbol{\alpha}_1^{(q)}, \cdots, \boldsymbol{\alpha}_{L_q}^{(q)})^\top \in \mathbb{R}^{L_q\times K_q}$ such that, for any $l \in \{0\,..\,L_q-1\}$,

$$\mathrm{Y}_l^{(q)} := \boldsymbol{\alpha}_l^{(q)\top} \cdot \mathbf{D}^{(q)}. \tag{5.35}$$

As in the color mixing stage, this operation is implemented as a 1×1 convolution layer.

A schematic representation of the real- and complex-valued wavelet blocks can be found in Figure 5.12.

Sparse Regularization. For any group $q \in \{0\,..\,Q-1\}$ and output channel $l \in \{0\,..\,L_q-1\}$ in the q-th group, we want the model to select one and only one wavelet packet feature map within the q-th group. In other words, each row vector $\boldsymbol{\alpha}_l^{(q)} := (\alpha_{l,1}^{(q)}, \cdots, \alpha_{l,K_q}^{(q)})^\top$ of $\boldsymbol{A}^{(q)}$ contains no more than one nonzero element, such that (5.35) becomes

$$\mathrm{Y}_l^{(q)} = \alpha_{lk}^{(q)} \mathrm{X}_k^{(q)} \tag{5.36}$$

for some (unknown) value of $k \in \{0\,..\,K_q-1\}$. To enforce this property during training, we add a mixed-norm l^1/l^∞-regularizer (J. Liu and J. Ye, 2010) to the loss function to penalize non-sparse feature map mixing as follows:

$$\mathcal{L} := \mathcal{L}_0 + \sum_{q=0}^{Q-1} \lambda_q \sum_{l=0}^{L_q-1} \left(\frac{\|\boldsymbol{\alpha}_l^{(q)}\|_1}{\|\boldsymbol{\alpha}_l^{(q)}\|_\infty} - 1 \right), \tag{5.37}$$

where $\mathcal{L}_0$ denotes the standard cross-entropy loss and $\boldsymbol{\lambda} \in \mathbb{R}^Q$ denotes a vector of regularization hyperparameters. Note that the unit bias in (5.37) serves for interpretability of the regularized loss ($\mathcal{L} = \mathcal{L}_0$ in the desired configuration) but has no impact on training.

5.A.2 Batch Normalization in ResNet

This is a complement to Section 5.4.5. In many recent architectures including ResNet, the bias (see Figure 5.6) is replaced by an affine batch normalization layer (BN). In this section, we show how to adapt our approach to this context.

A BN layer is parameterized by trainable weight and bias vectors, respectively denoted by $\boldsymbol{a}$ and $\boldsymbol{b} \in \mathbb{R}^L$. In the remaining of the section, we consider input images $\mathbf{X}$ as a stack of discrete stochastic processes. Then, (5.7) is replaced by

$$\mathrm{A}_l := \mathrm{MaxPool}\left\{\mathrm{ReLU}\left(a_l \cdot \frac{\mathrm{Y}_l - \mathbb{E}_m[\mathrm{Y}_l]}{\sqrt{\mathbb{V}_m[\mathrm{Y}_l] + \varepsilon}} + b_l\right)\right\}, \tag{5.38}$$

with Y_l satisfying (5.6) (output of the first convolution layer). In the above expression, we have introduced $\mathbb{E}_m(\mathrm{Y}_l) \in \mathbb{R}$ and $\mathbb{V}_m(\mathrm{Y}_l) \in \mathbb{R}_+$, which respectively denote the mean expected value and variance of $\mathrm{Y}_l[\boldsymbol{n}]$, for indices $\boldsymbol{n}$ contained in the support of Y_l, denoted by $\mathrm{supp}(\mathrm{Y}_l)$. Let us denote by $N \in \mathbb{N} \setminus \{0\}$ the support size of input images. Therefore, if the filter's support size N_f is much smaller that N, then $\mathrm{supp}(\mathrm{Y}_l)$ is roughly of size N/m. We thus define the above quantities as follows:

$$\mathbb{E}_m[\mathrm{Y}_l] := \frac{m^2}{N^2} \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathbb{E}[\mathrm{Y}_l[\boldsymbol{n}]]; \tag{5.39}$$

$$\mathbb{V}_m[\mathrm{Y}_l] := \frac{m^2}{N^2} \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathbb{V}[\mathrm{Y}_l[\boldsymbol{n}]]. \tag{5.40}$$

In practice, estimators are computed over a minibatch of images, hence the layer's denomination. Besides, $\varepsilon > 0$ is a small constant added to the denominator for numerical stability. For the sake of concision, we now assume that $a_l = 1$. Extensions to other multiplicative factors is straightforward.

Let $l \in \mathcal{G}$ denote a Gabor channel. Then, recall that Y_l satisfies (5.28) (output of the WBlock), with

$$\widetilde{\mathrm{V}}_l := \mathrm{Re}\, \widetilde{\mathrm{W}}_l, \tag{5.41}$$

where $\widetilde{\mathrm{W}}_l$ denotes one of the Gabor-like filters spawned by DT-$\mathbb{C}$WPT, as written in (5.27).[4] The following proposition states that, if the kernel's bandwidth is small enough and excludes the origin (band-pass filter), then the output of the convolution layer sums to zero. This result is not straightforward because of the subsampled convolutions.

Proposition 5.1. *We assume that Hypotheses 4.4 and 4.5 are satisfied (monochrome and Gabor-like convolution kernels). If, moreover,*

$$\|\boldsymbol{\theta}_l\|_1 > \frac{\kappa}{2}, \tag{5.42}$$

then

$$\sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathrm{Y}_l[\boldsymbol{n}] = 0, \tag{5.43}$$

where we remind that $\mathrm{Y}_l \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ has been introduced in (5.6).

Proof. Using Hypothesis 4.4, we get

$$\mathrm{Y}_l = \left(\mathrm{X}_l^{\mathrm{lum}} * \mathrm{Re}\, \overline{\widetilde{\mathrm{W}}_l}\right) \downarrow m, \tag{5.44}$$

[4] In practice, (5.27) is not always perfectly satisfied for WAlexNet (see Section 5.A.1). However, we ignore this shortcoming in our experiments.

where $\mathrm{X}_l^{\mathrm{lum}}$ and $\widetilde{\mathrm{W}}_l$ have been introduced in (4.195) and (4.191), respectively. We consider the Shannon interpolations of $\mathrm{X}_l^{\mathrm{lum}}$ and $\widetilde{\mathrm{W}}_l$, respectively denoted by $F_{\mathrm{X}_l^{\mathrm{lum}}} \in L^2_{\mathbb{R}}(\mathbb{R}^2)$ and $\Psi_{\widetilde{\mathrm{W}}_l} \in L^2_{\mathbb{C}}(\mathbb{R}^2)$, satisfying (4.30). We also define, as in (4.49),

$$F_0 : \boldsymbol{x} \mapsto \left(F_{\mathrm{X}_l^{\mathrm{lum}}} * \overline{\Psi}_{\widetilde{\mathrm{W}}_l}\right)(\boldsymbol{x})\, \mathrm{e}^{i\langle \boldsymbol{\theta}_l/s,\, \boldsymbol{x}\rangle}. \tag{5.45}$$

According to (4.31) (Proposition 4.2), Hypothesis 4.5 implies $\Psi_{\widetilde{\mathrm{W}}_l} \in \mathcal{V}(\boldsymbol{\theta}_l/s,\, \kappa/s)$. Therefore, according to Lemma 4.1,

$$\operatorname{supp} \widehat{F_0} \subset B_\infty\left(\frac{\kappa}{2s}\right). \tag{5.46}$$

Moreover, according to Hypothesis 4.5, $\kappa \leq \pi/m$. We actually relax this hypothesis, and consider that $\kappa \leq 2\pi/m$. Thus,

$$B_\infty\left(\frac{\kappa}{2s}\right) \subset B_\infty\left(\frac{\pi}{ms}\right). \tag{5.47}$$

Therefore,

$$F_0 \in \mathcal{V}^{(s')}, \qquad \text{with} \qquad s' := ms. \tag{5.48}$$

Note that the proof of Theorem 4.1 follows a similar reasoning, with $s' := 2ms$ instead of $s' := ms$. We now introduce, similar to (4.59), $\mathrm{X}_0 \in l^2_{\mathbb{R}}(\mathbb{Z}^2)$ such that

$$\mathrm{X}_0[\boldsymbol{n}] := s' F_0(s'\boldsymbol{n}) \tag{5.49}$$

for any $\boldsymbol{n} \in \mathbb{Z}^2$. On the one hand, according to (4.26) (Lemma 4.2) with $G \leftarrow F_0$, $\mathrm{Y} \leftarrow \mathrm{X}_0$ and $s \leftarrow s'$, we get, for any $\boldsymbol{\xi} \in B_\infty(\pi/s')$,

$$\widehat{F_0}(\boldsymbol{\xi}) = s'\, \widehat{\mathrm{X}_0}(s'\boldsymbol{\xi}). \tag{5.50}$$

On the other hand,

$$\mathrm{X}_0[\boldsymbol{n}] = ms\left(F_{\mathrm{X}_l^{\mathrm{lum}}} * \overline{\Psi}_{\widetilde{\mathrm{W}}_l}\right)(ms\boldsymbol{n})\, \mathrm{e}^{i\langle \boldsymbol{\theta}_l/s,\, ms\boldsymbol{n}\rangle} \tag{5.51}$$

$$= ms\left[\left(\mathrm{X}_l^{\mathrm{lum}} * \overline{\widetilde{\mathrm{W}}_l}\right) \downarrow m\right][\boldsymbol{n}]\, \mathrm{e}^{i\langle m\boldsymbol{\theta}_l,\, \boldsymbol{n}\rangle}, \tag{5.52}$$

according to (4.33) in Proposition 4.2, with $\mathrm{X} \leftarrow \mathrm{X}_l^{\mathrm{lum}}$ and $\mathrm{W} \leftarrow \widetilde{\mathrm{W}}_l$. Therefore,

$$\sum_{\boldsymbol{n}\in\mathbb{Z}^2} \mathrm{Y}_l[\boldsymbol{n}] = \frac{1}{s'} \operatorname{Re}\left(\sum_{\boldsymbol{n}\in\mathbb{Z}^2} \mathrm{X}_0[\boldsymbol{n}]\, \mathrm{e}^{-i\langle m\boldsymbol{\theta}_l,\, \boldsymbol{n}\rangle}\right) \tag{5.53}$$

$$= \frac{1}{s'} \operatorname{Re} \widehat{\mathrm{X}_0}(m\boldsymbol{\theta}_l) = \frac{1}{s'^2} \operatorname{Re} \widehat{F_0}(\boldsymbol{\theta}_l/s), \tag{5.54}$$

according to (5.50) with $\boldsymbol{\xi} \leftarrow \boldsymbol{\theta}_l/s$. By hypothesis (5.42), $\|\boldsymbol{\theta}_l/s\|_1 > \kappa/(2s)$. Therefore, according to (5.46), $\boldsymbol{\theta}_l/s$ is outside the support of $\widehat{F_0}$, which concludes the proof. □

In practice, the power spectrum of DT-$\mathbb{C}$WPT filters cannot be exactly zero on regions with nonzero measure, since they are finitely supported. However, we assume that it is concentrated in a region of size $\pi/2^{J-1} = \pi/m$, as evidenced in Table 4.1. Therefore, since we have discarded low-pass filters, the conditions of Proposition 5.1 are approximately met for $\widetilde{\mathrm{W}}_l$. We then assume that (5.43) is satisfied. Moreover, we assume that $\mathbb{E}[\mathrm{Y}_l[\boldsymbol{n}]]$ is constant for any $\boldsymbol{n} \in \mathrm{supp}(\mathrm{Y}_l)$.[5] We then get, for any $\boldsymbol{n} \in \mathbb{Z}^2$, $\mathbb{E}[\mathrm{Y}_l[\boldsymbol{n}]] = 0$. Therefore, interchanging max pooling and ReLU yields the normalized version of (5.20):

$$\mathrm{A}_l^{\max} = \mathrm{ReLU}\left(\frac{\mathrm{Y}_l^{\max}}{\sqrt{\mathbb{E}_m[\mathrm{Y}_l^2] + \varepsilon}} + b_l\right). \tag{5.55}$$

As in Section 5.4.1, we replace $\mathrm{Y}_l^{\max}$ by $\mathrm{Y}_l^{\mathrm{mod}}$ for any Gabor channel $l \in \mathcal{G}$, which yields the normalized version of (5.22):

$$\mathrm{A}_l^{\mathrm{mod}} := \mathrm{ReLU}\left(\frac{\mathrm{Y}_l^{\mathrm{mod}}}{\sqrt{\mathbb{E}_m[\mathrm{Y}_l^2] + \varepsilon}} + b_l\right). \tag{5.56}$$

Implementing (5.56) within a deep learning architecture is cumbersome because Y_l needs to be explicitly computed and kept in memory, in addition to $\mathrm{Y}_l^{\mathrm{mod}}$. Instead, we want to express the second-order moment $\mathbb{E}_m[\mathrm{Y}_l^2]$ (in the denominator) as a function of $\mathrm{Y}_l^{\mathrm{mod}}$. To this end, we state the following proposition.

Proposition 5.2. *Under Hypotheses 4.4 and 4.5, we have*

$$\|\mathrm{Y}_l^{\mathrm{mod}}\|_2^2 = \frac{1}{4}\|\mathrm{Z}_l\|_2^2, \tag{5.57}$$

with

$$\mathrm{Z}_l := \sum_{k=0}^{K-1} (\mathrm{X}_k * \overline{\mathrm{W}}_{lk}) \downarrow m, \tag{5.58}$$

where the complex-valued filter $\mathrm{W}_{lk} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ *has been introduced in* (5.12).

Proof. Under Hypothesis 4.4, (5.58) becomes

$$\mathrm{Z}_l = \left(\mathrm{X}_l^{\mathrm{lum}} * \overline{\widetilde{\mathrm{W}}_l}\right) \downarrow m, \tag{5.59}$$

similar to (5.44). As in the proof of Proposition 5.1, we consider the low-frequency function $F_0 \in L^2_{\mathbb{C}}(\mathbb{R}^2)$ satisfying (5.45), and a uniform sampling $\mathrm{X}_0 \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ of F_0 at interval $s' := ms$, satisfying (5.49). Besides, we consider another uniform sampling of F_0, denoted by X_{00}, satisfying

$$\mathrm{X}_{00}[\boldsymbol{n}] := s'' F_0(s''\boldsymbol{n}), \tag{5.60}$$

with $s'' := 2ms$.

[5] Aside from boundary effects, this is true if $\mathbb{E}[\mathrm{X}^{\mathrm{lum}}[\boldsymbol{n}]]$ is constant for any $\boldsymbol{n} \in \mathrm{supp}(\mathrm{X}^{\mathrm{lum}})$. This property is a consequence of the stationary hypothesis formulated in Proposition 4.10. It is a rough description of images of natural scenes or man-made objects. This hypothesis has to be considered with caution, as explained in Remark 4.16.

On the one hand, using (5.52) and (5.59), we get

$$\|\mathrm{X}_0\|_2^2 = m^2 s^2 \, \|\mathrm{Z}_l\|_2^2 \,. \tag{5.61}$$

On the other hand, $\mathrm{Y}_l^{\mathrm{mod}}$, defined in (5.23), can be written

$$\mathrm{Y}_l^{\mathrm{mod}} = \left| \left(\mathrm{X}_l^{\mathrm{lum}} * \widetilde{\overline{\mathrm{W}_l}} \right) \downarrow (2m) \right| . \tag{5.62}$$

Moreover, $\mathrm{X}_{00}[\boldsymbol{n}] = 2\mathrm{X}_0[2\boldsymbol{n}]$ for any $\boldsymbol{n} \in \mathbb{Z}^2$. Therefore, using again (5.52), we get

$$\|\mathrm{X}_{00}\|_2^2 = 4m^2 s^2 \, \|\mathrm{Y}_l^{\mathrm{mod}}\|_2^2 . \tag{5.63}$$

Finally, we show that

$$\|\mathrm{X}_{00}\|_2^2 = \|\mathrm{X}_0\|_2^2 \,. \tag{5.64}$$

To do so, we consider Hypothesis 4.5, and apply a similar reasoning as (5.46)–(5.48) in the proof of Proposition 5.1. Considering $\kappa \leq 2\pi/m$ on the one hand and $\kappa \leq \pi/m$ on the other hand, we get

$$\operatorname{supp} \widehat{F_0} \subset B_\infty \left(\frac{\pi}{ms} \right) \qquad \text{and} \qquad \operatorname{supp} \widehat{F_0} \subset B_\infty \left(\frac{\pi}{2ms} \right) , \tag{5.65}$$

which yields

$$F_0 \in \mathcal{V}^{(s')} \qquad \text{and} \qquad F_0 \in \mathcal{V}^{(s'')} . \tag{5.66}$$

Therefore, according to (4.26) (Lemma 4.2) with $G \leftarrow F_0$, $\mathrm{Y} \leftarrow \mathrm{X}_0$ (respectively, $\mathrm{Y} \leftarrow \mathrm{X}_{00}$), and $s \leftarrow s'$ (respectively, $s \leftarrow s''$), we get

$$\|F_0\|_{L^2} = \|\mathrm{X}_0\|_2 \qquad \text{and} \qquad \|F_0\|_{L^2} = \|\mathrm{X}_{00}\|_2 \,, \tag{5.67}$$

which yields (5.64). Finally, combining (5.61) and (5.63) concludes the proof. □

In the denominator of (5.56), we have

$$\mathbb{E}_m[\mathrm{Y}_l^2] = \frac{m^2}{N^2} \, \mathbb{E}\left[\|\mathrm{Y}_l\|_2^2 \right] . \tag{5.68}$$

Moreover, $\mathrm{Y}_l = \operatorname{Re} \mathrm{Z}_l$. However, there is no deterministic relationship between $\|\mathrm{Y}_l\|_2^2$ and $\|\mathrm{Z}_l\|_2^2$, due to aliasing effects: the energy of Z_l may be unbalanced between its real and imaginary parts. Nevertheless, we can reasonably assume that

$$\mathbb{E}\left[\|\mathrm{Y}_l\|_2^2 \right] = \frac{1}{2} \mathbb{E}\left[\|\mathrm{Z}_l\|_2^2 \right] . \tag{5.69}$$

Therefore, Proposition 5.2 implies that

$$\mathbb{E}\left[\|\mathrm{Y}_l^{\mathrm{mod}}\|_2^2 \right] = \frac{1}{2} \mathbb{E}\left[\|\mathrm{Y}_l\|_2^2 \right] . \tag{5.70}$$

Furthermore, we compute, similar to (5.68),

$$\mathbb{E}_{2m}\left[\mathrm{Y}_l^{\mathrm{mod}\,2} \right] = \frac{4m^2}{N^2} \, \mathbb{E}\left[\left\| \mathrm{Y}_l^{\mathrm{mod}} \right\|_2^2 \right] \tag{5.71}$$

$$= 2 \, \mathbb{E}_m[\mathrm{Y}_l^2] , \tag{5.72}$$

according to (5.70) and (5.68). In the above expression, the second-order moment of $\mathrm{Y}_l^{\mathrm{mod}}$ is computed on feature maps which are twice smaller than Y_l in both directions, hence the index "$2m$", which is the subsampling factor for the $\mathbb{C}$Mod operator. Finally, (5.56) becomes

$$\mathrm{A}_l^{\mathrm{mod}} := \mathrm{ReLU}\left(\frac{\mathrm{Y}_l^{\mathrm{mod}}}{\sqrt{\frac{1}{2}\mathbb{E}_{2m}\left[{\mathrm{Y}_l^{\mathrm{mod}}}^2\right] + \varepsilon}} + b_l\right). \tag{5.73}$$

In the case of ResNet, the bias layer (Bias) is therefore preceded by a batch normalization layer without mean centering satisfying (5.73), which we call BN0.

5.A.3 Experimental Settings

We provide further information that complements the experimental details presented in Section 5.5.1 and Table 5.1.

As explained in Section 5.4.2, the decomposition depth J is chosen such that $m = 2^{J-1}$, where m denotes the subsampling factor. Since $m = 4$ in AlexNet and 2 in ResNet, we get $J = 3$ and 2, respectively. Therefore, the number of dual-tree filters $K_{\mathrm{dt}} := 2 \times 4^J$ is equal to 128 and 32, respectively. For any $k \in \{0\,..\,K_{\mathrm{dt}} - 1\}$, we denote by $\boldsymbol{\theta}_{k'}^{\mathrm{dt}}$ the characteristic frequency of $\mathrm{W}_{k'}^{\mathrm{dt}}$, such that

$$\mathrm{W}_{k'}^{\mathrm{dt}} \in \mathcal{J}(\boldsymbol{\theta}_{k'}^{\mathrm{dt}}, \kappa), \tag{5.74}$$

where $\kappa := \pi/2^{J-1}$ denotes the bandwidth of the DT-$\mathbb{C}$WPT filters such as introduced in (4.214). Recall that the Gabor-like property (5.74) has been established in Proposition 4.9. We then manually selected $K'_{\mathrm{dt}} < K_{\mathrm{dt}}$ filters. In particular, we removed the two low-pass filters, which are outside the scope of our theoretical study, satisfying

$$\boldsymbol{\theta}_{k'}^{\mathrm{dt}} \in \left\{\boldsymbol{\theta} \in [0,\, \pi] \times [-\pi,\, \pi] \;\middle|\; \|\boldsymbol{\theta}\|_\infty \leq \frac{\pi}{2^J}\right\}. \tag{5.75}$$

Besides, for computational reasons, in WAlexNet we removed the 32 filters satisfying

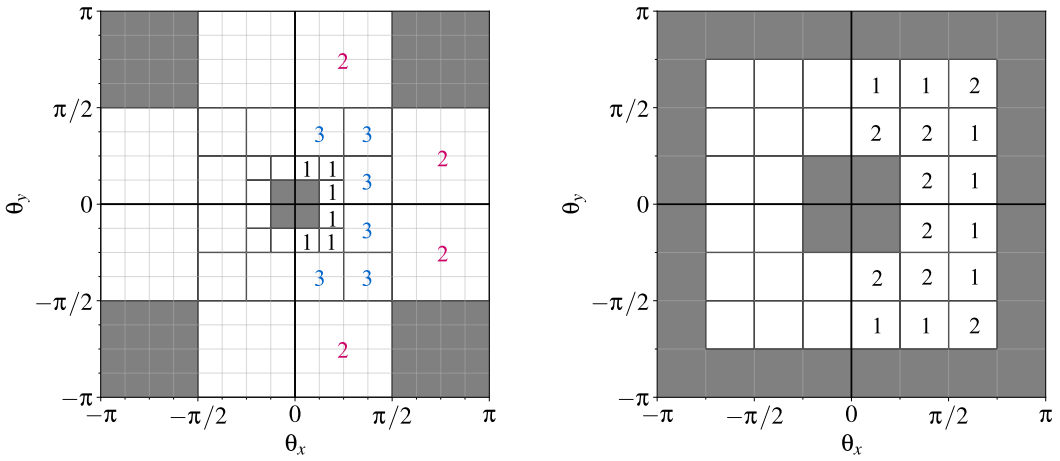
$$\boldsymbol{\theta}_{k'}^{\mathrm{dt}} \in \left\{\boldsymbol{\theta} := (\theta_x,\, \theta_y)^\top \in [0,\, \pi] \times [-\pi,\, \pi] \;\middle|\; \min\left(|\theta_x|,\, |\theta_y|\right) > \frac{\pi}{2}\right\}, \tag{5.76}$$

corresponding to the corner regions of the Fourier domain, as depicted in Figure 5.13a. These characteristic frequencies are indeed absent from the freely-trained model, as evidenced in Figure 5.8a. Finally, in WResNet we removed the 14 filters whose bandwidths outreach the boundaries of the Fourier domain $[-\pi,\, \pi]^2$. Their characteristic frequencies satisfy

$$\boldsymbol{\theta}_{k'}^{\mathrm{dt}} \in \left\{\boldsymbol{\theta} \in [0,\, \pi] \times [-\pi,\, \pi] \;\middle|\; \|\boldsymbol{\theta}\|_\infty > \frac{3\pi}{4}\right\}, \tag{5.77}$$

as depicted in Figure 5.13b. These filters indeed have a poorly-defined orientation, since a small fraction of their energy is located at the far end of the Fourier domain (Bayram and I. W. Selesnick, 2008, see Fig. 1, "Proposed DT-$\mathbb{C}$WPT"). Therefore, they somewhat exhibit a checkerboard pattern.[6]

[6] Note that the same procedure could have been applied to WAlexNet, but it was deemed unnecessary because the boundary filters were in general spontaneously discarded during training.



(a) WAlexNet ($J = 3$) (b) WResNet ($J = 2$)

Figure 5.13. Mapping scheme from DT-$\mathbb{C}$WPT feature maps $\mathbf{D} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)^{K_{\mathrm{dt}}}$ to the wavelet block's output $\mathbf{Y}^{\mathrm{gab}} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)^{L_{\mathrm{gab}}}$. Each wavelet feature map is symbolized by a small square in the Fourier domain, where its energy is mainly located. The gray areas show the feature maps which have been manually removed, according to (5.75), (5.77) and (5.77). Elsewhere, each group of feature maps $\mathbf{D}^{(q)} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)^{K_q}$ is symbolized by a dark frame (K_q is always equal to 1 for WResNet). For each group $q \in \{0\,..\,Q-1\}$, a number indicates how many output channels L_q are assigned to it. The colored numbers in (a) refer to groups on which we have applied l^∞/l^1-regularization. Note that, since inputs are real-valued, only the half-plane of positive x-values is considered.

As explained in Section 5.A.1, once the DT-$\mathbb{C}$WPT feature maps have been manually selected, the output $\mathbf{D}' \in l^2_{\mathbb{C}}(\mathbb{Z}^2)^{K'_{\mathrm{dt}}}$ is sliced into Q groups of channels $\mathbf{D}^{(q)} \in l^2_{\mathbb{C}}(\mathbb{Z}^2)^{K_q}$. For each group q, a depthwise linear mapping from $\mathbf{D}^{(q)}$ to a bunch of output channels $\mathbf{Y}^{(q)} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)^{L_q}$ is performed. Finally, the wavelet block's output feature maps $\mathbf{Y}^{\mathrm{gab}} \in l^2_{\mathbb{R}}(\mathbb{Z}^2)^{L_{\mathrm{gab}}}$ are obtained by concatenating the outputs $\mathbf{Y}^{(q)}$ depthwise, for any $q \in \{0\,..\,Q-1\}$. Figure 5.13 shows how the above grouping is made, and how many output channels L_q each group q is assigned to. These values were estimated from the freely-trained models, by analyzing the empirical distribution of the characteristic frequencies $(\boldsymbol{\theta}_l)_{l \in \mathcal{G}}$ such as introduced in (5.19).

During training, the above process aims at selecting one single DT-$\mathbb{C}$WPT feature map among each group. This is achieved through mixed-norm l^∞/l^1 regularization, as introduced in (5.37). The regularization hyperparameters $\boldsymbol{\lambda}^{(q)}$ have been chosen empirically. If they are too small, then regularization will not be effective. On the contrary, if they are too large, then the regularization term will become predominant, forcing the trainable parameter vector $\boldsymbol{\alpha}_l^{(q)}$ to randomly collapse to 0 except for one element. The chosen values of $\boldsymbol{\lambda}^{(q)}$ are displayed in Table 5.6.

As for the freely-trained models (Figure 5.4), the statistical distribution of $(\widetilde{\rho}_l)_{k \in \mathcal{G}}$, where $\widetilde{\rho}_l \in [0,\, 1]$ (5.18) denotes the maximum percentage of energy of $\widetilde{\mathrm{W}}_l \in l^2_{\mathbb{C}}(\mathbb{Z}^2)$ within a Fourier window of size $\kappa \times \kappa$, is plotted in Figure 5.14. As before, the window size κ has been set to its largest admissible value, *i.e.*, $\kappa := \pi/m = \pi/2^{J-1}$. If (5.36) is satisfied (perfect sparse regularization, *i.e.*, exactly one DT-$\mathbb{C}$WPT filter selected during the training process), then we should have $\widetilde{\rho}_l \approx 0.95$ for WAlexNet and 0.98 for WResNet, according to Table 4.1. This is true for WResNet, since each Gabor channel $l \in \mathcal{G}$ has been

Model	Filt. frequency	Reg. param.
WAlexNet	$[\pi/8, \pi/4[$	–
	$[\pi/4, \pi/2[$	$4.1 \cdot 10^{-3}$
	$[\pi/2, \pi[$	$3.2 \cdot 10^{-4}$
WResNet	any	–

Table 5.6. Regularization hyperparameters $\boldsymbol{\lambda}^{(q)}$ for each group q of DT-$\mathbb{C}$WPT feature maps. The groups with only one feature map do not need any regularization since this feature map is automatically selected. The second and third rows of WAlexNet correspond to the blue and magenta groups in Figure 5.13a, respectively.

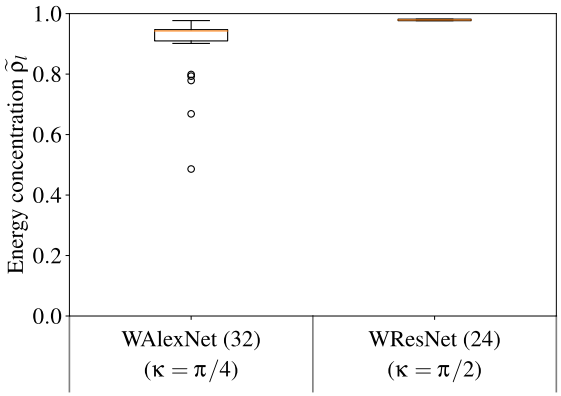


Figure 5.14. Mathematical twins: energy concentration of the Gabor-like kernels $\left(\widetilde{\mathrm{W}}_l\right)_{l \in \mathcal{G}}$ (5.27) within a Fourier window of size $\kappa \times \kappa$, with $\kappa := \pi/m$. This plot is to be compared with Figure 5.4 for the freely-trained models. According to Table 4.1 (energy concentration of DT-$\mathbb{C}$WPT filters with depth $J = 3$, *i.e.*, stride $m = 4$), we could expect $\widetilde{\rho}_l \approx 0.95$ for any $l \in \mathcal{G}$ in WAlexNet. However, this is generally not true, because the wavelet block may fail to properly select a single DT-$\mathbb{C}$WPT filter (see Figure 5.13).

assigned to one DT-$\mathbb{C}$WPT filter exactly—see Figure 5.13. However, in WAlexNet, the l^1/l^∞-regularizer (5.37) guides training towards sparse filter selection, without completely enforcing it. Therefore, (5.36) is not fully guaranteed, which explains the distribution plotted in Figure 5.14. We ignored this shortcoming in our experiments, and treated all Gabor channels as if (5.36) was properly achieved.

5.B Appendix: Computational Cost

This section provides technical details about our estimation of the computational cost (FLOPs), such as reported in Table 5.5, for *one input image* and *one Gabor channel*. This metric was estimated in the case of standard 2D convolutions.

Average Computation Time per Operation

The following values have been determined experimentally using PyTorch (CPU computations). They have been normalized with respect to the computation time of an addition.

$$
\begin{aligned}
t_{\mathrm{s}} &= 1.0 && \text{(addition);}\\
t_{\mathrm{p}} &= 1.0 && \text{(multiplication);}\\
t_{\mathrm{e}} &= 0.75 && \text{(exponential);}\\
t_{\mathrm{mod}} &= 3.5 && \text{(modulus);}\\
t_{\mathrm{relu}} &= 0.75 && \text{(ReLU);}\\
t_{\mathrm{max}} &= 12.0 && \text{(max pooling).}
\end{aligned}
$$

Computational Cost per Layer

In the following paragraphs, $L \in \mathbb{N} \setminus \{0\}$ denotes the number of output channels (depth) and $N' \in \mathbb{N} \setminus \{0\}$ denotes the size of output feature maps (height and width). However, note that N' is not necessary the same for all layers. For instance, in standard ResNet, the output of the first convolution layer is of size $N' = 112$, whereas the output of the subsequent max pooling layer is of size $N' = 56$. For each type of layer, we calculate the number of FLOPs required to produce a single output channel $l \in \{0\,..\,L-1\}$. Moreover, we assume, without loss of generality, that the model processes one input image at a time.

Convolution Layers. Inputs of size $(K \times N \times N)$ (input channels, height and width); outputs of size $(L \times N' \times N')$. For each output unit, a convolution layer with kernels of size $(N_{\mathrm{f}} \times N_{\mathrm{f}})$ requires KN_{f}^2 multiplications and $KN_{\mathrm{f}}^2 - 1$ additions. Therefore, the computational cost per output channel is equal to

$$T_{\mathrm{conv}} = N'^2 \left((KN_{\mathrm{f}}^2 - 1) \cdot t_{\mathrm{s}} + KN_{\mathrm{f}}^2 \cdot t_{\mathrm{p}} \right). \tag{5.78}$$

Complex Convolution Layers. Inputs of size $(K \times N \times N)$; complex-valued outputs of size $(L \times N' \times N')$. For each output unit, a complex-valued convolution layer requires $2 \times KN_{\mathrm{f}}^2$ multiplications and $2 \times (KN_{\mathrm{f}}^2 - 1)$ additions. Computational cost per output channel:

$$T_{\mathbb{C}\,\mathrm{conv}} = 2N'^2 \left((KN_{\mathrm{f}}^2 - 1) \cdot t_{\mathrm{s}} + KN_{\mathrm{f}}^2 \cdot t_{\mathrm{p}} \right). \tag{5.79}$$

Note that, in our implementations, the complex-valued convolution layers are less expensive than the real-valued ones, because the output size N' is twice smaller, due to the larger subsampling factor.

Bias and ReLU. Inputs and outputs of size $(L \times N' \times N')$. One evaluation for each output unit:

$$T_{\mathrm{bias}} = N'^2\, t_{\mathrm{s}} \qquad \text{and} \qquad T_{\mathrm{relu}} = N'^2\, t_{\mathrm{relu}}. \tag{5.80}$$

Max Pooling. Outputs of size $(L \times N' \times N')$, with N' depending on whether subsampling is performed at this stage (no subsampling when followed by a blur pooling layer). One evaluation for each output unit:

$$T_{\mathrm{max}} = N'^2\, t_{\mathrm{max}}. \tag{5.81}$$

Modulus Pooling. Complex-valued inputs and real-valued outputs of size $(L \times N' \times N')$. One evaluation for each output unit:

$$T_{\text{mod}} = N'^2 \, t_{\text{mod}}. \tag{5.82}$$

Batch Normalization. Inputs and outputs of size $(L \times N' \times N')$. A batch normalization (BN) layer, described in (5.38), can be split into several stages.

(1) Mean: N'^2 additions.

(2) Standard deviation: N'^2 multiplications, N'^2 additions (second moment), N'^2 additions (subtract squared mean).

(3) Final value: N'^2 additions (subtract mean), $2N'^2$ multiplications (divide by standard deviation and multiplicative coefficient).

Overall, the computational cost per image and output channel of a BN layer is equal to

$$T_{\text{bn}} = N'^2 \, (4\, t_{\text{s}} + 3\, t_{\text{p}}) \, . \tag{5.83}$$

Static Blur Pooling. Inputs of size $(L \times 2N' \times 2N')$; outputs of size $(L \times N' \times N')$. For each output unit, a static blur pooling layer (R. Zhang, 2019) with filters of size $(N_{\text{b}} \times N_{\text{b}})$ requires N_{b}^2 multiplications and $N_{\text{b}}^2 - 1$ additions. The computational cost per output channel is therfore equal to

$$T_{\text{blur}} = N'^2 \left((N_{\text{b}}^2 - 1) \cdot t_{\text{s}} + N_{\text{b}}^2 \cdot t_{\text{p}} \right) . \tag{5.84}$$

Adaptive Blur Pooling. Inputs of size $(L \times 2N' \times 2N')$; outputs of size $(L \times N' \times N')$. An adaptive blur pooling layer (X. Zou et al., 2023) with filters of size $(N_{\text{b}} \times N_{\text{b}})$ splits the L output channels into $Q := L/L_{\text{g}}$ groups of L_{g} channels that share the same blurring filters. The adaptive blur pooling layer can be decomposed into the following stages.

(1) Generation of blurring filters using a convolution layer with trainable kernels of size $(N_{\text{b}} \times N_{\text{b}})$: inputs of size $(L \times 2N' \times 2N')$, outputs of size $(QN_{\text{b}}^2 \times N' \times N')$. For each output unit, this stage requires LN_{b}^2 multiplications and $LN_{\text{b}}^2 - 1$ additions. The computational cost divided by the number L of channels is therefore equal to

$$T_{\text{conv ablur}} = N'^2 \, \frac{N_{\text{b}}^2}{L_{\text{g}}} \left((LN_{\text{b}}^2 - 1) \cdot t_{\text{s}} + LN_{\text{b}}^2 \cdot t_{\text{p}} \right) . \tag{5.85}$$

Note that, despite being expressed on a per-channel basis, the above computational cost depends on the number L of output channels. This is due to the asymptotic complexity of this stage in $O(L^2)$.

(2) Batch normalization, inputs and outputs of size $(QN_{\text{b}}^2 \times N' \times N')$:

$$T_{\text{bn ablur}} = N'^2 \, \frac{N_{\text{b}}^2}{L_{\text{g}}} \, (4\, t_{\text{s}} + 3\, t_{\text{p}}) \, . \tag{5.86}$$

(3) Softmax along the depthwise dimension:

$$T_{\text{sftmx ablur}} = N'^2 \frac{N_\text{b}^2}{L_\text{g}}(t_\text{e} + t_\text{s} + t_\text{p}). \tag{5.87}$$

(4) Blur pooling of input feature maps, using the filter generated at stages (1)–(3): inputs of size $(L \times 2N' \times 2N')$, outputs of size $(L \times N' \times N')$. The computational cost per output channel is identical to the static blur pooling layer, even though the weights may vary across channels and spatial locations:

$$T_\text{blur} = N'^2 \left((N_\text{b}^2 - 1) \cdot t_\text{s} + N_\text{b}^2 \cdot t_\text{p}\right). \tag{5.88}$$

Overall, the computational cost of an adaptive blur pooling layer per input image and output channel is equal to

$$T_\text{ablur} = N'^2 \frac{N_\text{b}^2}{L_\text{g}} \left[\left((L+1)N_\text{b}^2 + 3\right) \cdot t_\text{s} + \left((L+1)N_\text{b}^2 + 4\right) \cdot t_\text{p} + t_\text{e}\right]. \tag{5.89}$$

We notice that an adaptive blur pooling layer has an asymptotic complexity in $O(N_\text{b}^4)$, versus $O(N_\text{b}^2)$ for static blur pooling.

Application to AlexNet- and ResNet-based Models

Since they are normalized by the computational cost of standard models, the FLOPs reported in Table 5.5 only depend on the size of the convolution kernels and blur pooling filters, respectively denoted by N_f and $N_\text{b} \in \mathbb{N} \setminus \{0\}$. In addition, the computational cost of the adaptive blur pooling layer depend on the number of output channels L as well as the number of output channels per group L_g.

In practice, N_f is respectively equal to 11 and 7 for AlexNet- and ResNet-based models. Moreover, $N_\text{b} = 3$, $L = 64$ and $L_\text{g} = 8$. Actually, the computational cost is largely determined by the convolution layers, including step (1) of adaptive blur pooling.

5.C Appendix: Memory Footprint

This section provides technical details about our estimation of the memory footprint for *one input image* and *one output channel*, such as reported in Table 5.5. This metric is generally difficult to estimate, and is very implementation-dependent. Hereafter, we consider the size of the output tensors, as well as intermediate tensors saved by `torch.autograd` for the backward pass. However, we didn't take into account the tensors containing the trainable parameters. To get the size of intermediate tensors, we used the Python package PyTorchViz.[7] These tensors are saved according to the following rules.

- Convolution (Conv), batch normalization (BN), Bias, max pooling (MaxPool or Max), blur pooling (BlurPool), and Modulus: the input tensors are saved, not the output. When Bias follows Conv or BN, no intermediate tensor is saved.
- ReLU, Softmax: the output tensors are saved, not the input.

[7] https://github.com/szagoruyko/pytorchviz

- If an intermediate tensor is saved at both the output of a layer and the input of the next layer, its memory is not duplicated. An exception is Modulus, which stores the input feature maps as complex numbers.
- MaxPool or Max: a tensor of indices is kept in memory, indicating the position of the maximum values. The tensors are stored as 64-bit integers, so they weight twice as much as conventional float-32 tensors.
- BN: four 1D tensors of length L are kept in memory: computed mean and variance, and running mean and variance. For BN0 (Section 5.A.2), where variance is not computed, only two tensors are kept in memory.

In the following paragraphs, we denote by L the number of output channels, N the size of input images (height and width), m the subsampling factor of the baseline models (4 for AlexNet, 2 for ResNet), N_b the blurring filter size (set to 3 in practice). For each model, a table contains the size of all saved intermediate or output tensors. For example, the values associated to "Layer1 → Layer2" correspond to the depth (number of channel), height and width of the intermediate tensor between Layer1 and Layer2.

AlexNet-based Models

No Antialiasing. Conv → Bias → ReLU → MaxPool.

ReLU → MaxPool	L	$\frac{N}{m}$	$\frac{N}{m}$
MaxPool → *output*	L	$\frac{N}{2m}$	$\frac{N}{2m}$
MaxPool indices (×2)	L	$\frac{N}{2m}$	$\frac{N}{2m}$

Then, the memory footprint for each output channel is equal to

$$\implies S_\text{std} = \frac{7}{4}\frac{N^2}{m^2}.$$

Static Blur Pooling. Conv → Bias → ReLU → BlurPool → Max → BlurPool.

ReLU → BlurPool	L	$\frac{2N}{m}$	$\frac{2N}{m}$
BlurPool → Max	L	$\frac{N}{m}$	$\frac{N}{m}$
Max → BlurPool	L	$\frac{N}{m}$	$\frac{N}{m}$
Max indices (×2)	L	$\frac{N}{m}$	$\frac{N}{m}$
BlurPool → *output*	L	$\frac{N}{2m}$	$\frac{N}{2m}$

$$\implies S_\text{blur} = \frac{33}{4}\frac{N^2}{m^2}.$$

ℂMod-based Approach. ℂConv → Modulus → Bias → ReLU.

ℂConv → Modulus	$2L$	$\frac{N}{2m}$	$\frac{N}{2m}$
Modulus → Bias	L	$\frac{N}{2m}$	$\frac{N}{2m}$
ReLU → *output*	L	$\frac{N}{2m}$	$\frac{N}{2m}$

$$\implies S_\text{mod} = \frac{N^2}{m^2}.$$

ResNet-based Models

No Antialiasing. Conv → BN → Bias → ReLU → MaxPool.

Conv → BN	L	$\frac{N}{m}$	$\frac{N}{m}$
BN metrics	$4L$	–	–
ReLU → MaxPool	L	$\frac{N}{m}$	$\frac{N}{m}$
MaxPool → *output*	L	$\frac{N}{2m}$	$\frac{N}{2m}$
MaxPool indices (×2)	L	$\frac{N}{2m}$	$\frac{N}{2m}$

$$\implies S_{\text{std}} = \frac{11}{4}\frac{N^2}{m^2} + 4 \approx \frac{11}{4}\frac{N^2}{m^2}.$$

Static Blur Pooling. Conv → BN → Bias → ReLU → Max → BlurPool.

Conv → BN	L	$\frac{N}{m}$	$\frac{N}{m}$
BN metrics	$4L$	–	–
ReLU → Max	L	$\frac{N}{m}$	$\frac{N}{m}$
Max → BlurPool	L	$\frac{N}{m}$	$\frac{N}{m}$
Max indices (×2)	L	$\frac{N}{m}$	$\frac{N}{m}$
BlurPool → *output*	L	$\frac{N}{2m}$	$\frac{N}{2m}$

$$\implies S_{\text{blur}} = \frac{21}{4}\frac{N^2}{m^2} + 4 \approx \frac{21}{4}\frac{N^2}{m^2}.$$

Adaptive Blur Pooling. Conv → BN → Bias → ReLU → Max → ABlurPool.

Conv → BN	L	$\frac{N}{m}$	$\frac{N}{m}$
BN metrics	$4L$	–	–
ReLU → Max	L	$\frac{N}{m}$	$\frac{N}{m}$
Max → ABlurPool	L	$\frac{N}{m}$	$\frac{N}{m}$
Max indices (×2)	L	$\frac{N}{m}$	$\frac{N}{m}$
ABlurPool → *output*	L	$\frac{N}{2m}$	$\frac{N}{2m}$
Generate adaptive blurring filter			
Conv → BN → Bias → Softmax			
Conv → BN	$\frac{LN_{\text{b}}^2}{L_{\text{g}}}$	$\frac{N}{2m}$	$\frac{N}{2m}$
BN metrics	$4\frac{LN_{\text{b}}^2}{L_{\text{g}}}$	–	–
Softmax → *output*	$\frac{LN_{\text{b}}^2}{L_{\text{g}}}$	$\frac{N}{2m}$	$\frac{N}{2m}$

$$\begin{aligned}\implies S_{\text{ablur}} &= \frac{21}{4}\frac{N^2}{m^2} + 4 + \frac{N_{\text{b}}^2}{L_{\text{g}}}\left(\frac{N^2}{2m^2} + 4\right) \\ &\approx \frac{21}{4}\frac{N^2}{m^2} + \frac{N_{\text{b}}^2}{L_{\text{g}}}\frac{N^2}{2m^2}.\end{aligned}$$

$\mathbb{C}$Mod-based Approach. $\mathbb{C}$Conv $\to$ Modulus $\to$ BN0 $\to$ Bias $\to$ ReLU.

$\mathbb{C}$Conv $\to$ Modulus	$2L$	$\frac{N}{2m}$	$\frac{N}{2m}$
Modulus $\to$ BN0	L	$\frac{N}{2m}$	$\frac{N}{2m}$
BN0 metrics	$2L$	–	–
ReLU $\to$ *output*	L	$\frac{N}{2m}$	$\frac{N}{2m}$

$$\implies S_{\text{mod}} = \frac{N^2}{m^2} + 2 \approx \frac{N^2}{m^2}.$$

Chapter 6

Conclusion and Perspectives

This thesis makes theoretical and experimental contributions to the understanding of shift invariance in CNNs. To this end, we applied theoretical tools from the field of digital image processing, thus reinforcing the link between the two disciplines. Our main achievements are summarized in Section 6.1. To wrap up, we explore in Section 6.2 several perspectives for future research.

6.1 Summary of Contributions

6.1.1 Theoretical Study

In Chapter 4, we presented a theoretical study on the relationship between two operators. The first one, $\mathbb{R}$Max, implements the max pooling of real-valued convolutions as implemented in CNNs, whereas the second one, $\mathbb{C}$Mod, implements the modulus of complex-valued convolutions. In this chapter, we showed that (1) $\mathbb{C}$Mod is nearly invariant to translations, if the convolution kernel is band-pass and oriented (Gabor-like filter); (2) $\mathbb{R}$Max and $\mathbb{C}$Mod produce comparable outputs, except for some filter frequencies regularly scattered across the Fourier domain. We then combined these two properties to establish a probabilistic metric of stability for $\mathbb{R}$Max outputs, which depends on the convolution kernel's frequency vector. This chapter tackled the three issues raised in Section 1.1.1 as follows.

- This work sought to bridge the gap in research regarding the role of the max pooling operator on shift invariance in CNNs. Inspired by Waldspurger (2015), we showed that max pooling tends to extract the local maximum amplitude of a high-frequency 2D signal. When the signal exhibits slow variations of its envelope, this process approximates the modulus of a complex signal, thereby establishing stability properties to small input shifts, and a connection with complex analysis.

- While many studies have investigated invariance properties in the continuous framework, the discrete case requires careful consideration, because aliasing effects can appear when sampling high-frequency signals. Our work addressed this issue by using Shannon's sampling theorem to demonstrate a link between analog and digital image processing. Additionally, due to potential degeneracies that can arise when using max pooling (which operates on a discrete grid) with certain filter frequencies,

we adopted a probabilistic approach, unraveling a regular pattern of unstable filters in the Fourier domain—see Figure 4.3. This is what motivated the design of our antialiasing approach, which we have described and tested in Chapter 5.

- A link was missing between real- and complex-valued convolutions in CNNs. By comparing the outputs of $\mathbb{C}$Mod and $\mathbb{R}$Max operators, we established a connection between these two worlds, creating opportunities for extensions of the results obtained for complex wavelet transforms and wavelet scattering networks. To paraphrase Tygert et al. (2016), the correspondence between standard real-valued CNNs (using max pooling) and complex wavelets is no longer "just a vague analogy."

This work was essentially theoretical, with only essential experiments conducted on a deterministic model based on the dual-tree complex wavelet packet transform (DT-$\mathbb{C}$WPT).

6.1.2 Experimental Study

Building upon our theoretical work, Chapter 5 introduced an antialiasing approach for CNNs, based on convolutions with complex-valued Gabor-like kernels. Specifically, we considered the $\mathbb{C}$Mod operator as a proxy for $\mathbb{R}$Max, extracting comparable, yet more stable features. In compliance with the theory established in Chapter 4, the $\mathbb{R}$Max-$\mathbb{C}$Mod substitution was only applied to the output channels associated with Gabor-like kernels, which are known to arise spontaneously in the first layer of CNNs trained on image datasets. Prior to antialiasing, we thus forced the kernels to adopt a Gabor-like structure by designing a DT-$\mathbb{C}$WPT-based mathematical twin reproducing the behavior of freely-trained architectures, which we called WCNN. This chapter synthesized the ideas introduced in Sections 1.1.2 and 1.1.3. On the one hand, the WCNN design achieved the following goals:

- Conventional Gabor transforms require manual tuning of a set of parameters. In contrast, DT-$\mathbb{C}$WPT only requires choosing the subsampling factor, which is imposed by the network architecture. It shall be noted that a set of QMFs also needs to be selected—see Section 3.3.3. However, in practice, a commonly-used filter with a reasonable length, such as Q-shift with length 10, sufficed to produce convincing results. A deeper discussion on this topic is provided in Section 6.2.3.

- We evaluated the similarity between WCNNs and freely-trained CNNs in terms of kernel frequency, orientation, and bandwidth. Our findings showed that both models shared similar kernel characteristics, which supported our decision to use DT-$\mathbb{C}$WPT as a discrete Gabor-like transform.

- Controlling the behavior of the model allowed for easier implementation of our antialiasing method. In particular, the real and imaginary components of the DT-$\mathbb{C}$WPT filters form an approximate Hilbert transform pair, which naturally provides the complex-valued representation needed for this purpose.

On the other hand, our antialiasing approach, which we have tested on the WCNN architecture, offers several advantages over concurrent methods for improving shift invariance.

- Unlike blur-pooling-based antialiasing (R. Zhang, 2019; X. Zou et al., 2023), our method extracts high-frequency features that are stable to translations, thereby

achieving a better balance between stability and information preserving. This resulted in increased accuracy for most models.

- In contrast to adaptive blur pooling (X. Zou et al., 2023), our approach is more computationally and memory efficient, and does not require additional trainable parameters.
- Our proposed approach is designed for stability with respect to fractional-pixel translations, similar to blur pooling but unlike APS (Chaman and Dokmanic, 2021), which does not perform antialiasing. Nonetheless, combining it with APS could lead to even more robust models. This topic is left for future work.

In the following sections, we discuss several research directions that build upon the contributions of this thesis. Specifically, we explore the following topics: (1) stability of max pooling outputs with respect to other types of deformations; (2) applicability of our theoretical and experimental contributions to small convolution kernels; (3) learning optimal QMFs in the WCNN twin architecture introduced in Chapter 5; (4) using complex-valued convolutions layers as a stable tokenizer for vision transformers.

6.2 Future Research Directions

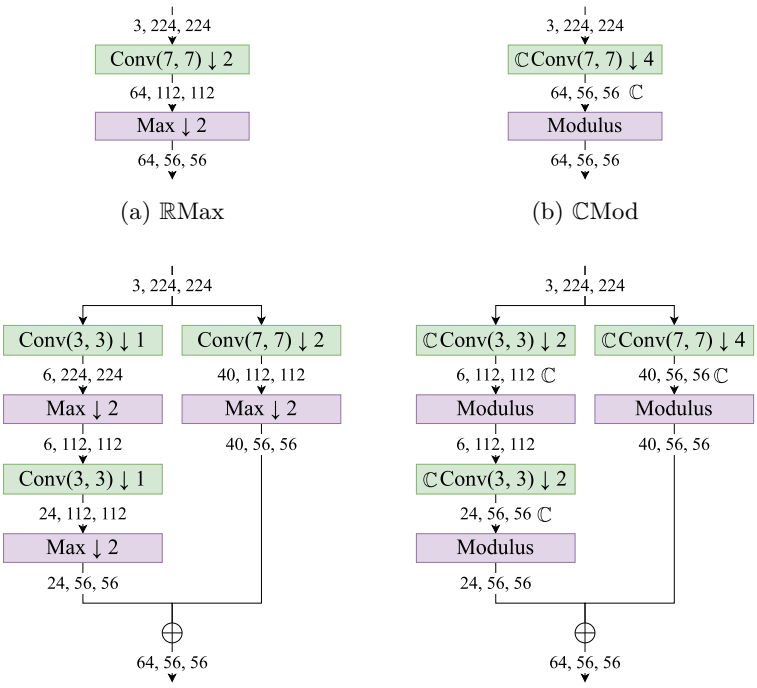
6.2.1 Beyond Translation Invariance

Chapter 4 tackled the subject of translation invariance of $\mathbb{C}$Mod and $\mathbb{R}$Max outputs, when the convolution kernel is band-pass and oriented. As stated in Theorems 4.1 and 4.3, stability increases when the Fourier window size κ decreases. The asymptotical case is the infinite filter of complex exponential $\mathrm{W}[\boldsymbol{n}] := \mathrm{e}^{i\langle \boldsymbol{n}, \boldsymbol{\theta}\rangle}$, for a given $\boldsymbol{\theta} \in [-\pi, \pi]^2$, which satisfies $\mathrm{W} \in \mathcal{J}(\boldsymbol{\theta}, \kappa)$ (4.12) with $\kappa = 0$. The convolution with $\overline{\mathrm{W}}$, which computes the discrete-time Fourier transform up to a phase shift, has a magnitude which is perfectly translation invariant:

$$\forall \boldsymbol{u} \in \mathbb{R}^2, \ |\mathcal{T}_{\boldsymbol{u}}\mathrm{X} * \overline{\mathrm{W}}| = |\mathrm{X} * \overline{\mathrm{W}}|. \tag{6.1}$$

In CNNs, the bandwidth of the Gabor-like convolution kernels (*i.e.*, κ) tends to decrease with larger stride m (subsampling factor) and wider kernels. For instance, AlexNet, which implements an initial convolution layer with kernels of size 11×11 and stride $m = 4$, yields higher Fourier resolution than ResNet, with kernels of size 7×7 and $m = 2$ (see Figure 5.2). An analogy can be done with the dual-tree complex wavelet packet transform (DT-$\mathbb{C}$WPT), for which the subsampling factor $m := 2^J$ is inversely proportional to the bandwidth $\kappa := \pi/2^{J-1}$. Therefore, in order to improve shift invariance, it could be tempting to jointly widen the convolution kernel and increase the stride m in the first convolution layer, thus allowing to increase the Fourier resolution. However, we face another problem at high frequencies: when the bandwidth of a convolution filter decreases, it becomes more susceptible to small distortions in the input signal (Mallat, 2012). When choosing hyperparameters for a convolution layer, a tradeoff must therefore be found to preserve both shift invariance and stability to deformations.

In wavelet scattering networks (ScatterNets) (Bruna and Mallat, 2013), the tradeoff is chosen as follows: the bandwidth is larger—and the stride is smaller—at higher frequencies, following a standard multiresolution structure. This ensures stability to deformations. Then, near shift-invariance is obtained by low-pass filtering and downsampling



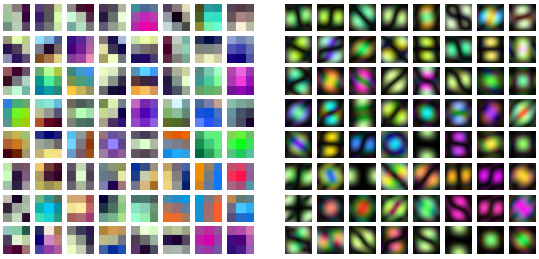
(a) ℝMax (b) ℂMod

(c) ScatterNet-like ℝMax (d) ScatterNet-like ℂMod

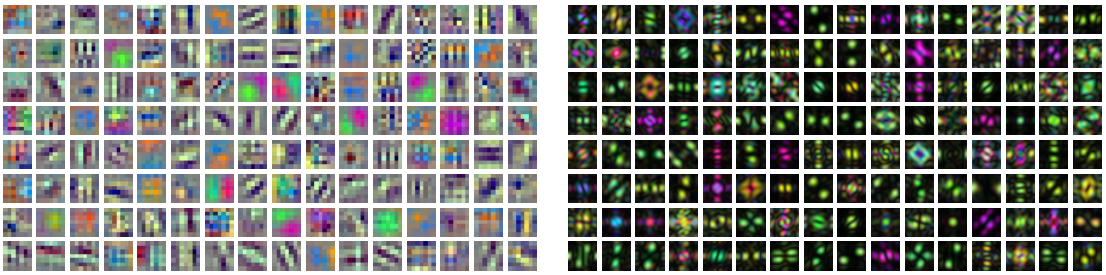
Figure 6.1. **Top**: typical CNN configuration where all output channels share the same stride and kernel size (batch normalization, bias and ReLU are ignored for the sake of simplicity). (a) ResNet's initial convolution layer followed by max pooling. (b) Shift-invariant model: ℝMax-ℂMod substitution introduced in Chapter 5 (simplified version). **Bottom**: ScatterNet-inspired multiresolution architecture, with two layers of scattering transform. The smaller kernels are intended for higher frequencies (and larger bandwidth). (c) ℝMax version, similar to standard CNNs. (d) ℂMod version (antialiased model), similar to standard ScatterNets with trainable parameters. The scattering-like structure allows combining near-shift invariance and stability to deformations. The number of output channels have been arbitrarily chosen and may be adjusted to optimize performances.

the feature maps after computing the modulus—not unlike the blur pooling strategy introduced by R. Zhang (2019). Finally, high-frequency information is preserved by further decomposing the magnitude of the wavelet feature maps, therefore adopting a multilayer convolutional architecture. In contrast however, standard convolution layers in CNNs generally share the same kernel size and stride across all output channels, which prevents the network from learning a multiresolution-like feature extractor. Figure 6.1 provides a schematic representation of a standard convolution layer followed by max pooling, as well as a ScatterNet-inspired architecture, which contains parallel convolution layers with different kernel sizes and strides. The ℂMod version of this architecture (Figure 6.1d) is therefore expected to be stable to both translations and other types of deformations.

Further investigation would be needed to analyze the Lipschitz continuity to deformations for ℂMod and ℝMax outputs in the discrete framework, and to improve the antialiasing approach implemented in Chapter 5 to incorporate these additional constraints.



(a) One layer, 64 output channels. Kernel size $= 3 \times 3$, stride $= 2$



(b) Two layers, 128 output channels. Kernel size $= 7 \times 7$, stride $= 4$

Figure 6.2. Resulting kernels obtained by stacking the first convolution layers of VGG-11 trained on ImageNet-1K. Left: spatial domain; right: Fourier domain (magnitude spectra). The two-layer kernels (b) are visually quite similar to ResNet's first-layer kernels (see Figure 5.2b).

6.2.2 What About Small Convolution Kernels?

This thesis was focused on convolution kernels that are wide enough to exhibit Gabor-like patterns. However, a certain number of CNN architectures, including VGG (Simonyan et al., 2014) or MobileNet (Howard et al., 2017; Sandler et al., 2018), rely on filters with small 3×3 receptive fields, even in the first layer. By stacking several such convolution layers, it is possible to get larger receptive fields, as in previous models. This architecture has two main advantages: reducing the number of trainable parameters, and improving feature discrimination by adding a nonlinear activation function and/or pooling operator between any two convolution layers. According to Simonyan et al. (2014), this can be seen as imposing a regularization on a wider convolution kernel.

Can our theoretical results and antialiasing method be adapted to this situation? We can indeed reasonably assume that the basic feature extraction principles observed in previous CNNs, such as corner and edge detection, remain valid on those models. A first thought was to remove the nonlinear operators between convolution layers and visualize the resulting non-regularized convolution kernels. Figure 6.2a displays the first-layer 3×3 kernels of VGG-11 trained on ImageNet-1K, whereas Figure 6.2b shows the resulting kernels obtained by stacking the first two convolution layers. To be consistent with the original network's subsampling factor, we increased the stride of the first convolution layer to compensate the absence of max pooling layer. The resulting subsampling factor is therefore equal to 4, with receptive field of size 7×7. This preliminary result suggests that Gabor-like structures seem to emerge from the combination of several convolution

layers. To our knowledge, there is no theoretical work studying the relationship between small-kernel architectures and Gabor-like transforms. Delving deeper into this topic may therefore lead to a generalization of our work to a broader family of CNNs.

Additionally, the ScatterNet-inspired model presented in Figure 6.1 suggests that CNNs could benefit from using a stack of small convolution kernels followed by nonlinear activation and pooling operators, as implemented in VGG, in parallel to wider kernels with larger stride, as implemented in ResNet. This would lead to a scattering-like multiresolution architecture, with potential for producing stable feature representations (with respect to both translations and deformations), while preserving high-frequency information.

6.2.3 Learning Optimal Filters for DT-$\mathbb{C}$WPT

In the design of the mathematical twin in Chapter 5, DT-$\mathbb{C}$WPT was implemented with two pairs of QMFs $(h^{[0]}, g^{[0]})$ and $(h^{[1]}, g^{[1]})$ of type Q-shift (Kingsbury, 2003), which approximately satisfy the half-sample delay condition (3.91) required to get analytic complex wavelets. However, these filters were fixed during training. As explained in Section 3.2.3, the ability of wavelets to produce sparse image representations depend on the number of vanishing moments and support size. Additionally, other aspects such as filter symmetry (Daubechies, 1993), or relationship between the dual-tree low-pass filters $h^{[0]}$ and $h^{[1]}$ (I. W. Selesnick et al., 2005), may be considered to produce optimal feature vectors. However, some of these requirements are mutually exclusive. For instance, there exists a tradeoff between the number of vanishing moments and the support size (Daubechies, 1988). Besides, the half-sample delay condition cannot be exactly satisfied if the filters are finitely-supported. Therefore, choosing the best set of filters is not straightforward and may depend on the task at hand.

A possible room for improvement could be to consider the QMFs as trainable parameters. To this end, we conducted initial experiments on the WCNN models introduced in Section 5.4. The filters were initialized as before, but their weights were updated during training. The experiment however was not conclusive: the $\mathbb{R}$Max-$\mathbb{C}$Mod substitution resulted in a loss of accuracy. This could be improved by adding a regularization term to the model's cross-entropy loss, to ensure that conditions (3.17), (3.18) and (3.91) remain satisfied after training.

6.2.4 From CNNs to Vision Transformers

Throughout this thesis, we focused on the first convolution layer followed by nonlinear activation and max pooling operators. While this may seem like a limited scope, it is important to note that initial layers play a critical role in CNNs by extracting low-level geometric features such as edges, corners or textures. Therefore, a specific attention is required for their design. In contrast, deeper layers are more focused on capturing high-level structures that conventional image processing tools are poorly suited for, as pointed out by Oyallon et al. (2017) in the context of hybrid scattering networks.

In fact, there is a growing interest in using self-attention mechanisms in computer vision to capture complex, long-range dependencies among image representations. As mentioned in Section 2.3.4, Dosovitskiy et al. (2021) were the first to adapt the transformer architecture from the field of NLP to computer vision. The network, referred to as *vision transformer*, or ViT in short, does not contain any convolution layer. Instead, input

images $\mathbf{X} \in \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^K$ are split into N fixed-size patches, each of which being flattened and linearly embedded into a token space of dimension D, using a fully-connected layer. The resulting matrix, denoted by $\boldsymbol{X} \in \mathbb{R}^{N \times D}$, serves as input for the first self-attention module as described in Section 2.3.4.[1] This simple procedure lacks inductive biases such as found in CNNs. Specifically, prior knowledge about the intrinsic structure of images is missing, such as local 2D connectivity and weight sharing, which led to the very design of CNNs by LeCun et al. (1989). As a consequence, while ViT outperforms CNNs when pretrained on large image datasets such as ImageNet-21K (containing 21 classes and 13 millions of images), it fails to do so when trained from scratch on medium-sized or datasets such as ImageNet-1K, CIFAR or MNIST. In the former case, the lack of inductive bias was compensated by the very large amount of data from which the network could learn the basic structure of images.

Recent work on vision transformers has proposed reintroducing inductive bias to the architecture, by using the first layers of a CNN as a "convolutional tokenizer" (Wu et al., 2021; Yuan et al., 2021; Hassani et al., 2022), instead of the naive patch extraction described above. In a nutshell, consider $\mathbf{Y} \in \left(l^2_{\mathbb{R}}(\mathbb{Z}^2)\right)^D$ as the output of a multilayer convolutional architecture. Then, flattening the 2D output feature maps yields a set of tokens $\boldsymbol{Y} \in \mathbb{R}^{N \times D}$, used as input for a self-attention module. Note that, unlike the original ViT, a single pixel in the input image is used for several tokens (overlapping patches). These hybrid CNN-ViT architectures succeeded in outperform standalone CNNs on medium or even small image datasets, while avoiding cumbersome pretraining on millions of images.

However, an important property has been overlooked: stability with respect to translations and deformations. Shift invariance in CNNs is often taken for granted (R. Zhang, 2019), but this is generally wrong, as discussed in this thesis. From the perspective of ViTs, we can reasonably assume that stable token embeddings could contribute to improving the network's predictive power. By applying our $\mathbb{C}$Mod-based antialiasing method to the convolutional tokenizer, we could provide self-attention modules with nearly shift-invariant inputs, which could be highly beneficial for ViTs, especially when the amount of available data is limited.

6.3 Epilogue

By establishing connections between real-valued CNNs and complex-valued image processing tools, this thesis contributes to the body of work focused on advancing our understanding of deep learning for computer vision. Increasing control over the behavior of computer vision algorithms is critical in domains where theoretical guarantees are essential, such as medicine, or when data are scarce.

[1] ViT actually implements positional embedding and multi-head attention, but this is left outside the scope of this discussion.

Bibliography

Achour, E. M., F. Malgouyres, and F. Mamalet (2022). "Existence, Stability and Scalability of Orthogonal Convolutional Neural Networks". *The Journal of Machine Learning Research* 23.1, 347:15743–347:15798.

Aharon, M., M. Elad, and A. Bruckstein (2006). "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation". *IEEE Transactions on Signal Processing* 54.11, pp. 4311–4322.

Aizerman, A. (1964). "Theoretical Foundations of the Potential Function Method in Pattern Recognition Learning". *Automation and Remote Control* 25, pp. 821–837.

Alekseev, A. and A. Bobe (2019). "GaborNet: Gabor Filters with Learnable Parameters in Deep Convolutional Neural Network". In: *2019 International Conference on Engineering and Telecommunication (EnT)*.

Andén, J. and S. Mallat (2014). "Deep Scattering Spectrum". *IEEE Transactions on Signal Processing* 62.16, pp. 4114–4128.

Athreya, K. B. and S. N. Lahiri (2006). *Measure Theory and Probability Theory*. Vol. 19. Springer.

Azulay, A. and Y. Weiss (2019). "Why Do Deep Convolutional Networks Generalize so Poorly to Small Image Transformations?" *Journal of Machine Learning Research* 20.184, pp. 1–25.

Bahdanau, D., K. Cho, and Y. Bengio (2016). "Neural Machine Translation by Jointly Learning to Align and Translate". In: *International Conference on Learning Representations (ICLR)*.

Balan, R., M. Singh, and D. Zou (2018). "Lipschitz Properties for Deep Convolutional Networks". *Contemporary Mathematics* 706, pp. 129–151.

Bansal, N., X. Chen, and Z. Wang (2018). "Can We Gain More from Orthogonality Regularizations in Training Deep Networks?" In: *Advances in Neural Information Processing Systems*.

Bayram, I. and I. W. Selesnick (2008). "On the Dual-Tree Complex Wavelet Packet and M-Band Transforms". *IEEE Transactions on Signal Processing* 56.6, pp. 2298–2310.

Bell, A. J. and T. J. Sejnowski (1997). "The "Independent Components" of Natural Scenes Are Edge Filters". *Vision Research* 37.23, pp. 3327–3338.

Berkson, J. (1944). "Application of the Logistic Function to Bio-Assay". *Journal of the American Statistical Association* 39.227, pp. 357–365.

Bietti, A. (2022). "Approximation and Learning with Deep Convolutional Models: A Kernel Perspective". In: *International Conference on Learning Representations (ICLR)*.

Bietti, A. and J. Mairal (2017). "Invariance and Stability of Deep Convolutional Representations". In: *Advances in Neural Information Processing Systems*.

— (2019a). "Group Invariance, Stability to Deformations, and Complexity of Deep Convolutional Representations". *The Journal of Machine Learning Research* 20.1, pp. 876–924.

— (2019b). "On the Inductive Bias of Neural Tangent Kernels". In: *Advances in Neural Information Processing Systems*.

Bigun, J., G. Granlund, and J. Wiklund (1991). "Multidimensional Orientation Estimation with Applications to Texture Analysis and Optical Flow". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13.8, pp. 775–790.

Bishop, C. M. and T. M. Mitchell (2014). *Pattern Recognition and Machine Learning*. Springer.

Boser, B. E., I. M. Guyon, and V. N. Vapnik (1992). "A Training Algorithm for Optimal Margin Classifiers". In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*.

Brochard, A., S. Zhang, and S. Mallat (2022). "Generalized Rectifier Wavelet Covariance Models For Texture Synthesis". In: *International Conference on Learning Representations (ICLR)*.

Brown, T., B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, and A. Askell (2020). "Language Models Are Few-Shot Learners". In: *Advances in Neural Information Processing Systems*.

Bruna, J. and S. Mallat (2013). "Invariant Scattering Convolution Networks". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.8, pp. 1872–1886.

Candès, E. J. and D. L. Donoho (1999). "Ridgelets: A Key to Higher-Dimensional Intermittency?" *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 357.1760, pp. 2495–2509.

— (2000). "Curvelets: A Surprisingly Effective Nonadaptive Representation for Objects with Edges". In: *International Conference on Curves and Surfaces*.

Carion, N., F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko (2020). "End-to-End Object Detection with Transformers". In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Chabiron, O., F. Malgouyres, J.-Y. Tourneret, and N. Dobigeon (2015). "Toward Fast Transform Learning". *International Journal of Computer Vision* 114.2, pp. 195–216.

Chaman, A. and I. Dokmanic (2021). "Truly Shift-Invariant Convolutional Neural Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Chang, S.-Y. and N. Morgan (2014). "Robust CNN-based Speech Recognition with Gabor Filter Kernels". In: *Fifteenth Annual Conference of the International Speech Communication Association*.

Chollet, F. (2017). "Xception: Deep Learning With Depthwise Separable Convolutions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Coifman, R. and M. Wickerhauser (1992). "Entropy-Based Algorithms for Best Basis Selection". *IEEE Transactions on Information Theory* 38.2, pp. 713–718.

Cortes, C. and V. Vapnik (1995). "Support-Vector Networks". *Machine Learning* 20.3, pp. 273–297.

Cotter, F. and N. Kingsbury (2019). "A Learnable Scatternet: Locally Invariant Convolutional Layers". In: *2019 IEEE International Conference on Image Processing (ICIP)*.

Cramer, J. S. (2002). "The Origins of Logistic Regression". *Tinbergen Institute Working Paper No. 2002-119/4*.

Czaja, W. and W. Li (2019). "Analysis of Time-Frequency Scattering Transforms". *Applied and Computational Harmonic Analysis* 47.1, pp. 149–171.

— (2020). "Rotationally Invariant Time–Frequency Scattering Transforms". *Journal of Fourier Analysis and Applications* 26.1, p. 4.

Daubechies, I. (1988). "Orthonormal Bases of Compactly Supported Wavelets". *Communications on pure and applied mathematics* 41.7, pp. 909–996.

— (1993). "Orthonormal Bases of Compactly Supported Wavelets II. Variations on a Theme". *SIAM Journal on Mathematical Analysis* 24.2, pp. 499–519.

Daugman, J. G. (1980). "Two-Dimensional Spectral Analysis of Cortical Receptive Field Profiles". *Vision Research* 20.10, pp. 847–856.

Dedmari, M. A., S. Conjeti, S. Estrada, P. Ehses, T. Stöcker, and M. Reuter (2018). "Complex Fully Convolutional Neural Networks for MR Image Reconstruction". In: *Machine Learning for Medical Image Reconstruction*.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". arXiv: 1810.04805.

Do, M. and M. Vetterli (2000). "Orthonormal Finite Ridgelet Transform for Image Compression". In: *Proceedings 2000 International Conference on Image Processing (ICIP)*.

— (2005). "The Contourlet Transform: An Efficient Directional Multiresolution Image Representation". *IEEE Transactions on Image Processing* 14.12, pp. 2091–2106.

Donoho, D. L. and M. Elad (2003). "Optimally Sparse Representation in General (Nonorthogonal) Dictionaries via L^1 Minimization". *Proceedings of the National Academy of Sciences* 100.5, pp. 2197–2202.

Donoho, D. L. and I. M. Johnstone (1995). "Adapting to Unknown Smoothness via Wavelet Shrinkage". *Journal of the American Statistical Association* 90.432, pp. 1200–1224.

Dosovitskiy, A., L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby (2021). "An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *International Conference on Learning Representations (ICLR)*.

Dua, S., U. R. Acharya, P. Chowriappa, and S. V. Sree (2012). "Wavelet-Based Energy Features for Glaucomatous Image Classification". *IEEE Transactions on Information Technology in Biomedicine* 16.1, pp. 80–87.

Duffin, R. J. and A. C. Schaeffer (1952). "A Class of Nonharmonic Fourier Series". *Transactions of the American Mathematical Society* 72.2, p. 341.

Elad, M., M. A. T. Figueiredo, and Y. Ma (2010). "On the Role of Sparse and Redundant Representations in Image Processing". *Proceedings of the IEEE* 98.6, pp. 972–982.

Fujieda, S., K. Takayama, and T. Hachisuka (2017). "Wavelet Convolutional Neural Networks for Texture Classification". arXiv: 1707.07394.

Fukushima, K. and S. Miyake (1982). "Neocognitron: A New Algorithm for Pattern Recognition Tolerant of Deformations and Shifts in Position". *Pattern Recognition* 15.6, pp. 455–469.

Gabor, D. (1946). "Theory of Communication". *Journal of the Institution of Electrical Engineers - Part III: Radio and Communication Engineering* 93.26, pp. 429–457.

Gama, F., A. Ribeiro, and J. Bruna (2019). "Stability of Graph Scattering Transforms". In: *Advances in Neural Information Processing Systems*.

Gauthier, S., B. Thérien, L. Alsène-Racicot, M. Chaudhary, I. Rish, E. Belilovsky, M. Eickenberg, and G. Wolf (2022). "Parametric Scattering Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Georgiou, G. and C. Koutsougeras (1992). "Complex Domain Backpropagation". *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* 39.5, pp. 330–334.

Girshick, R. (2015). "Fast R-CNN". In: *Proceedings of the IEEE International Conference on Computer Vision*.

Glorot, X., A. Bordes, and Y. Bengio (2011). "Deep Sparse Rectifier Neural Networks". In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*.

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio (2014). "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems*.

Grossmann, A. and J. Morlet (1984). "Decomposition of Hardy Functions into Square Integrable Wavelets of Constant Shape". *SIAM Journal on Mathematical Analysis* 15.4, pp. 723–736.

Haar, A. (1911). “Zur Theorie der orthogonalen Funktionensysteme”. *Mathematische Annalen* 71.1, pp. 38–53.

Halmos, P. R. (2013). *Measure Theory*. Springer.

Harris, C. and M. Stephens (1988). “A Combined Corner and Edge Detector”. In: *Alvey Vision Conference*.

Hassani, A., S. Walton, N. Shah, A. Abuduweili, J. Li, and H. Shi (2022). “Escaping the Big Data Paradigm with Compact Transformers”. arXiv: 2104.05704.

Havlicek, J., J. Havlicek, and A. Bovik (1997). “The Analytic Image”. In: *Proceedings of International Conference on Image Processing*.

He, K., X. Zhang, S. Ren, and J. Sun (2016). “Deep Residual Learning for Image Recognition”. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Hendrycks, D. and T. Dietterich (2019). “Benchmarking Neural Network Robustness to Common Corruptions and Perturbations”. In: *International Conference on Learning Representations (ICLR)*.

Hinton, G. E., S. Osindero, and Y.-W. Teh (2006). “A Fast Learning Algorithm for Deep Belief Nets”. *Neural Computation* 18.7, pp. 1527–1554.

Hochreiter, S. and J. Schmidhuber (1997). “Long Short-Term Memory”. *Neural Computation* 9.8, pp. 1735–1780.

Howard, A. G., M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam (2017). “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications”. arXiv: 1704.04861.

Hu, J., L. Shen, and G. Sun (2018). “Squeeze-and-Excitation Networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Huang, G., Z. Liu, L. Van Der Maaten, and K. Q. Weinberger (2017). “Densely Connected Convolutional Networks”. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Hubel, D. H. and T. N. Wiesel (1959). “Receptive Fields of Single Neurones in the Cat’s Striate Cortex”. *The Journal of Physiology* 148.3, pp. 574–591.

— (1962). “Receptive Fields, Binocular Interaction and Functional Architecture in the Cat’s Visual Cortex”. *The Journal of Physiology* 160.1, pp. 106–154.2.

— (1968). “Receptive Fields and Functional Architecture of Monkey Striate Cortex”. *The Journal of Physiology* 195.1, pp. 215–243.

Iandola, F. N., S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer (2016). “SqueezeNet: AlexNet-level Accuracy with 50x Fewer Parameters and <0.5MB Model Size”. arXiv: 1602.07360.

Ioffe, S. and C. Szegedy (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *Proceedings of the 32nd International Conference on Machine Learning*.

Islam, M. A., S. Jia, and N. D. B. Bruce (2020). "How Much Position Information Do Convolutional Neural Networks Encode?" In: *International Conference on Learning Representations (ICLR)*.

Jones, J. P. and L. A. Palmer (1987). "An Evaluation of the Two-Dimensional Gabor Filter Model of Simple Receptive Fields in Cat Striate Cortex". *Journal of Neurophysiology* 58.6, pp. 1233–1258.

Kataoka, M., M. Kinouchi, and M. Hagiwara (1998). "Music Information Retrieval System Using Complex-Valued Recurrent Neural Networks". In: *SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.98CH36218)*.

Kayhan, O. S. and J. C. van Gemert (2020). "On Translation Invariance in CNNs: Convolutional Layers Can Exploit Absolute Spatial Location". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Ke, Y. and R. Sukthankar (2004). "PCA-SIFT: A More Distinctive Representation for Local Image Descriptors". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Khushaba, R. N., S. Kodagoda, S. Lal, and G. Dissanayake (2011). "Driver Drowsiness Classification Using Fuzzy Wavelet-Packet-Based Feature-Extraction Algorithm". *IEEE Transactions on Biomedical Engineering* 58.1, pp. 121–131.

Kim, M. and C. Guest (1990). "Modification of Backpropagation Networks for Complex-Valued Signal Processing in Frequency Domain". In: *1990 IJCNN International Joint Conference on Neural Networks*.

Kingsbury, N. (1999). "Image Processing with Complex Wavelets". *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 357.1760, pp. 2543–2560.

— (2000). "A Dual-Tree Complex Wavelet Transform with Improved Orthogonality and Symmetry Properties". In: *Proceedings 2000 International Conference on Image Processing (Cat. No.00CH37101)*.

— (2003). "Design of Q-shift Complex Wavelets for Image Processing Using Frequency Domain Energy Minimization". In: *Proceedings International Conference on Image Processing*.

Kingsbury, N. and J. Magarey (1998). "Wavelet Transforms in Image Processing". In: *Signal Analysis and Prediction*. Applied and Numerical Harmonic Analysis, pp. 27–46.

Krizhevsky, A. and G. Hinton (2009). *Learning Multiple Layers of Features from Tiny Images*. University of Toronto.

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2017). "ImageNet Classification with Deep Convolutional Neural Networks". *Communications of the ACM* 60.6, pp. 84–90.

Kuo, C. .-. J. (2016). "Understanding Convolutional Neural Networks with a Mathematical Model". *Journal of Visual Communication and Image Representation* 41, pp. 406–413.

Laine, A. and J. Fan (1993). "Texture Classification by Wavelet Packet Signatures". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15.11, pp. 1186–1191.

Langlotz, C. P., B. Allen, B. J. Erickson, J. Kalpathy-Cramer, K. Bigelow, T. S. Cook, A. E. Flanders, M. P. Lungren, D. S. Mendelson, J. D. Rudie, G. Wang, and K. Kandarpa (2019). "A Roadmap for Foundational Research on Artificial Intelligence in Medical Imaging: From the 2018 NIH/RSNA/ACR/The Academy Workshop". *Radiology* 291.3, pp. 781–791.

Lawrence, S., C. Giles, A. C. Tsoi, and A. Back (1997). "Face Recognition: A Convolutional Neural-Network Approach". *IEEE Transactions on Neural Networks* 8.1, pp. 98–113.

Le Pennec, E. and S. Mallat (2005). "Bandelet Image Approximation and Compression". *Multiscale Modeling & Simulation* 4.3, pp. 992–1039.

Learned, R. E. and A. S. Willsky (1995). "A Wavelet Packet Approach to Transient Signal Classification". *Applied and Computational Harmonic Analysis* 2.3, pp. 265–278.

LeCun, Y., Y. Bengio, and G. Hinton (2015). "Deep Learning". *Nature* 521.7553 (7553), pp. 436–444.

LeCun, Y., B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel (1989). "Handwritten Digit Recognition with a Back-Propagation Network". In: *Advances in Neural Information Processing Systems*.

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998). "Gradient-Based Learning Applied to Document Recognition". *Proceedings of the IEEE* 86.11, pp. 2278–2323.

Lee, C., H. Hasegawa, and S. Gao (2022). "Complex-Valued Neural Networks: A Comprehensive Survey". *IEEE/CAA Journal of Automatica Sinica* 9.8, pp. 1406–1426.

Lee, H., R. Grosse, R. Ranganath, and A. Y. Ng (2009). "Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations". In: *Proceedings of the 26th Annual International Conference on Machine Learning*.

Lee, J., J. Yang, and Z. Wang (2020). "What Does CNN Shift Invariance Look Like? A Visualization Study". arXiv: 2011.04127.

Leterme, H. (2023). *WCNN, a Python Library for Shift-Invariant Twin Models Based on Complex Wavelets*. URL: https://github.com/hubert-leterme/wcnn.

Leterme, H., K. Polisano, V. Perrier, and K. Alahari (2021). "Modélisation Parcimonieuse de CNNs Avec Des Paquets d'Ondelettes Dual-Tree". In: *ORASIS*.

— (2022). "On the Shift Invariance of Max Pooling Feature Maps in Convolutional Neural Networks". arXiv: 2209.11740.

— (2023). "From CNNs to Shift-Invariant Twin Models Based on Complex Wavelets". arXiv: 2212.00394.

Leung, H. and S. Haykin (1991). "The Complex Backpropagation Algorithm". *IEEE Transactions on Signal Processing* 39.9, pp. 2101–2104.

Li, T. and M. Zhou (2016). "ECG Classification Using Wavelet Packet Entropy and Random Forests". *Entropy* 18.8 (8), p. 285.

Liao, B. and F. Peng (2010). "Rotation-Invariant Texture Features Extraction Using Dual-Tree Complex Wavelet Transform". In: *2010 International Conference on Information, Networking and Automation (ICINA)*.

Lin, M., Q. Chen, and S. Yan (2014). "Network In Network". arXiv: 1312.4400.

Lin, Y., F. Lv, S. Zhu, M. Yang, T. Cour, K. Yu, L. Cao, and T. Huang (2011). "Large-Scale Image Classification: Fast Feature Extraction and SVM Training". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Liu, C.-C. and D.-Q. Dai (2009). "Face Recognition Using Dual-Tree Complex Wavelet Features". *IEEE Transactions on Image Processing* 18.11, pp. 2593–2599.

Liu, J. and J. Ye (2010). "Efficient L1/Lq Norm Regularization". arXiv: 1009.4766.

Liu, L., J. Liu, S. Yuan, G. Slabaugh, A. Leonardis, W. Zhou, and Q. Tian (2020). "Wavelet-Based Dual-Branch Network for Image Demoiréing". In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Liu, M., L. Jiao, X. Liu, L. Li, F. Liu, and S. Yang (2021). "C-CNN: Contourlet Convolutional Neural Networks". *IEEE Transactions on Neural Networks and Learning Systems* 32.6, pp. 2636–2649.

Liu, P., H. Zhang, W. Lian, and W. Zuo (2019). "Multi-Level Wavelet Convolutional Neural Networks". *IEEE Access* 7, pp. 74973–74985.

Lowe, D. G. (2004). "Distinctive Image Features from Scale-Invariant Keypoints". *International Journal of Computer Vision* 60.2, pp. 91–110.

Lu, H., H. Wang, Q. Zhang, D. Won, and S. W. Yoon (2018). "A Dual-Tree Complex Wavelet Transform Based Convolutional Neural Network for Human Thyroid Medical Image Segmentation". In: *IEEE International Conference on Healthcare Informatics (ICHI)*.

Luan, S., C. Chen, B. Zhang, J. Han, and J. Liu (2018). "Gabor Convolutional Networks". *IEEE Transactions on Image Processing* 27.9, pp. 4357–4366.

Mairal, J., F. Bach, J. Ponce, and G. Sapiro (2009). "Online Dictionary Learning for Sparse Coding". In: *Proceedings of the 26th Annual International Conference on Machine Learning*.

Mairal, J., M. Elad, and G. Sapiro (2008a). "Sparse Representation for Color Image Restoration". *IEEE Transactions on Image Processing* 17.1, pp. 53–69.

Mairal, J., P. Koniusz, Z. Harchaoui, and C. Schmid (2014). "Convolutional Kernel Networks". In: *Advances in Neural Information Processing Systems*.

Mairal, J., J. Ponce, G. Sapiro, A. Zisserman, and F. Bach (2008b). "Supervised Dictionary Learning". In: *Advances in Neural Information Processing Systems*.

Malfait, M. and D. Roose (1997). "Wavelet-Based Image Denoising Using a Markov Random Field a Priori Model". *IEEE Transactions on Image Processing* 6.4, pp. 549–565.

Mallat, S. and W. Hwang (1992). "Singularity Detection and Processing with Wavelets". *IEEE Transactions on Information Theory* 38.2, pp. 617–643.

Mallat, S. (1989). "Multiresolution Approximations and Wavelet Orthonormal Bases of $L^2(\mathbb{R})$". *Transactions of the American Mathematical Society* 315.1, pp. 69–87.

— (2009). *A Wavelet Tour of Signal Processing : The Sparse Way*. Academic Press.

— (2012). "Group Invariant Scattering". *Communications on Pure and Applied Mathematics* 65.10, pp. 1331–1398.

— (2016). "Understanding Deep Convolutional Networks". *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374.2065, p. 20150203.

Mallat, S. and I. Waldspurger (2015). "Deep Learning by Scattering". arXiv: 1306.5532.

Mallat, S., S. Zhang, and G. Rochette (2020). "Phase Harmonic Correlations and Convolutional Neural Networks". *Information and Inference: A Journal of the IMA* 9.3, pp. 721–747.

Mallat, S. and Z. Zhang (1993). "Matching Pursuits with Time-Frequency Dictionaries". *IEEE Transactions on Signal Processing* 41.12, pp. 3397–3415.

Marĉelja, S. (1980). "Mathematical Description of the Responses of Simple Cortical Cells*". *JOSA* 70.11, pp. 1297–1300.

Marcellin, M., M. Gormish, A. Bilgin, and M. Boliek (2000). "An Overview of JPEG-2000". In: *Proceedings DCC 2000. Data Compression Conference*.

McCulloch, W. S. and W. Pitts (1943). "A Logical Calculus of the Ideas Immanent in Nervous Activity". *The bulletin of mathematical biophysics* 5.4, pp. 115–133.

McFadden, D. (1973). "Conditional Logit Analysis of Qualitative Choice Behavior". In: *Frontiers in Econometrics*. Academic Press, pp. 105–142.

McKelvey, R. D. and W. Zavoina (1975). "A Statistical Model for the Analysis of Ordinal Level Dependent Variables". *The Journal of Mathematical Sociology* 4.1, pp. 103–120.

Meyer, Y. (1985). "Principe d'incertitude, Bases Hilbertiennes et Algèbres d'opérateurs". In: *Séminaire Bourbaki*.

Mikolajczyk, K. and C. Schmid (2005). "A Performance Evaluation of Local Descriptors". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.10, pp. 1615–1630.

Minsky, M. L. and S. A. Papert (1988). *Perceptrons: Expanded Edition*. MIT press.

Nason, G. P. and B. W. Silverman (1995). "The Stationary Wavelet Transform and Some Statistical Applications". In: *Wavelets and Statistics*. Ed. by A. Antoniadis and G. Oppenheim. Lecture Notes in Statistics. Springer, pp. 281–299.

Nelder, J. A. and R. W. M. Wedderburn (1972). "Generalized Linear Models". *Journal of the Royal Statistical Society: Series A (General)* 135.3, pp. 370–384.

Ning, F., D. Delhomme, Y. LeCun, F. Piano, L. Bottou, and P. Barbano (2005). "Toward Automatic Phenotyping of Developing Embryos from Videos". *IEEE Transactions on Image Processing* 14.9, pp. 1360–1371.

Olshausen, B. A. and D. J. Field (1996). "Emergence of Simple-Cell Receptive Field Properties by Learning a Sparse Code for Natural Images". *Nature* 381.6583 (6583), pp. 607–609.

— (1997). "Sparse Coding with an Overcomplete Basis Set: A Strategy Employed by V1?" *Vision Research* 37.23, pp. 3311–3325.

Osuna, E., R. Freund, and F. Girosit (1997). "Training Support Vector Machines: An Application to Face Detection". In: *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Oyallon, E., E. Belilovsky, and S. Zagoruyko (2017). "Scaling the Scattering Transform: Deep Hybrid Networks". In: *Proceedings of the IEEE International Conference on Computer Vision*.

Oyallon, E., E. Belilovsky, S. Zagoruyko, and M. Valko (2018). "Compressing the Input for CNNs with the First-Order Scattering Transform". In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Park, K. I. and M. Park (2018). *Fundamentals of Probability and Stochastic Processes with Applications to Communications*. Springer.

Paszke, A., S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer (2017). "Automatic Differentiation in PyTorch". In: *Advances in Neural Information Processing Systems*.

Pearl, R. and L. J. Reed (1920). "On the Rate of Growth of the Population of the United States since 1790 and Its Mathematical Representation1". *Proceedings of the National Academy of Sciences* 6.6, pp. 275–288.

Pérez, J. C., M. Alfarra, G. Jeanneret, A. Bibi, A. Thabet, B. Ghanem, and P. Arbeláez (2020). "Gabor Layers Enhance Network Robustness". In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Perlmutter, M., F. Gao, G. Wolf, and M. Hirn (2020). "Geometric Wavelet Scattering Networks on Compact Riemannian Manifolds". In: *Proceedings of The First Mathematical and Scientific Machine Learning Conference*.

Perronnin, F., J. Sánchez, and T. Mensink (2010). "Improving the Fisher Kernel for Large-Scale Image Classification". In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Peyré, G. and S. Mallat (2008). "Orthogonal Bandelet Bases for Geometric Images Approximation". *Communications on Pure and Applied Mathematics* 61.9, pp. 1173–1212.

Pittner, S. and S. V. Kamarthi (1999). "Feature Extraction from Wavelet Coefficients for Pattern Recognition Tasks". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21.1, pp. 83–88.

Pustelnik, N., A. Benazza-Benhayia, Y. Zheng, and J.-C. Pesquet (2016). "Wavelet-Based Image Deconvolution and Reconstruction". In: *Wiley Encyclopedia of Electrical and Electronics Engineering*. John Wiley & Sons, Ltd, pp. 1–34.

Rai, M. and P. Rivas (2020). "A Review of Convolutional Neural Networks and Gabor Filters in Object Recognition". In: *2020 International Conference on Computational Science and Computational Intelligence (CSCI)*.

Ramachandran, P., N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens (2019). "Stand-Alone Self-Attention in Vision Models". In: *Advances in Neural Information Processing Systems*.

Ranzato, M. aurelio, C. Poultney, S. Chopra, and Y. LeCun (2006). "Efficient Learning of Sparse Representations with an Energy-Based Model". In: *Advances in Neural Information Processing Systems*.

Ren, S., K. He, R. Girshick, and J. Sun (2015). "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *Advances in Neural Information Processing Systems*.

Riesenhuber, M. and T. Poggio (1999). "Hierarchical Models of Object Recognition in Cortex". *Nature Neuroscience* 2.11 (11), pp. 1019–1025.

Rodríguez, P., J. Gonzàlez, G. Cucurull, J. M. Gonfaus, and X. Roca (2017). "Regularizing CNNs with Locally Constrained Decorrelations". In: *International Conference on Learning Representations (ICLR)*.

Rosenblatt, F. (1958). "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain". *Psychological Review* 65.6, pp. 386–408.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). "Learning Representations by Back-Propagating Errors". *Nature* 323.6088 (6088), pp. 533–536.

Russakovsky, O., J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei (2015). "ImageNet Large Scale Visual Recognition Challenge". *International Journal of Computer Vision* 115.3, pp. 211–252.

Sanchez, J. and F. Perronnin (2011). "High-Dimensional Signature Compression for Large-Scale Image Classification". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Sander, M., P. Ablin, and G. Peyré (2022). "Do Residual Neural Networks Discretize Neural Ordinary Differential Equations?" In: *Advances in Neural Information Processing Systems*.

Sandler, M., A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen (2018). "MobileNetV2: Inverted Residuals and Linear Bottlenecks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Sarwar, S. S., P. Panda, and K. Roy (2017). "Gabor Filter Assisted Energy Efficient Fast Learning Convolutional Neural Networks". In: *2017 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*.

Scetbon, M. and Z. Harchaoui (2020). "Harmonic Decompositions of Convolutional Networks". In: *International Conference on Machine Learning*.

Scherer, D., A. Müller, and S. Behnke (2010). "Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition". In: *Artificial Neural Networks – ICANN 2010*.

Selesnick, I. (2001). "Hilbert Transform Pairs of Wavelet Bases". *IEEE Signal Processing Letters* 8.6, pp. 170–173.

Selesnick, I. W., R. Baraniuk, and N. Kingsbury (2005). "The Dual-Tree Complex Wavelet Transform". *IEEE Signal Processing Magazine* 22.6, pp. 123–151.

Sengur, A., I. Turkoglu, and M. C. Ince (2007). "Wavelet Packet Neural Networks for Texture Classification". *Expert Systems with Applications* 32.2, pp. 527–533.

Shannon, C. (1949). "Communication in the Presence of Noise". *Proceedings of the IRE* 37.1, pp. 10–21.

Sifre, L. and S. Mallat (2013). "Rotation, Scaling and Deformation Invariant Scattering for Texture Discrimination". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Simonyan, K., A. Vedaldi, and A. Zisserman (2014). "Deep inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps". In: *2nd International Conference on Learning Representations, ICLR 2014 - Workshop Track Proceedings*.

Simonyan, K. and A. Zisserman (2015). "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *International Conference on Learning Representations (ICLR)*.

Singh, A. and N. Kingsbury (2017). "Dual-Tree Wavelet Scattering Network with Parametric Log Transformation for Object Classification". In: *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*.

Smola, A. J. and B. Schölkopf (2004). "A Tutorial on Support Vector Regression". *Statistics and Computing* 14.3, pp. 199–222.

Starck, J.-L. and A. Bijaoui (1994). "Filtering and Deconvolution by the Wavelet Transform". *Signal Processing* 35.3, pp. 195–211.

Subasi, A. (2007). "EEG Signal Classification Using Wavelet Feature Extraction and a Mixture of Expert Model". *Expert Systems with Applications* 32.4, pp. 1084–1093.

Szegedy, C., S. Ioffe, V. Vanhoucke, and A. A. Alemi (2017). "Inception-v4, Inception-Resnet and the Impact of Residual Connections on Learning". In: *Thirty-First AAAI Conference on Artificial Intelligence*.

Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich (2015). "Going Deeper with Convolutions". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Szegedy, C., W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus (2014). "Intriguing Properties of Neural Networks". arXiv: 1312.6199.

Tan, M. and Q. Le (2019). "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: *International Conference on Machine Learning*.

Tibshirani, R. (1996). "Regression Shrinkage and Selection Via the Lasso". *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1, pp. 267–288.

Tompson, J., R. Goroshin, A. Jain, Y. LeCun, and C. Bregler (2015). "Efficient Object Localization Using Convolutional Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Torralba, A. and A. Oliva (2003). "Statistics of Natural Image Categories". *Network: Computation in Neural Systems* 14.3, pp. 391–412.

Toshev, A. and C. Szegedy (2014). "DeepPose: Human Pose Estimation via Deep Neural Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Trabelsi, C., O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal (2018). "Deep Complex Networks". In: *International Conference on Learning Representations (ICLR)*.

Tygert, M., J. Bruna, S. Chintala, Y. LeCun, S. Piantino, and A. Szlam (2016). "A Mathematical Motivation for Complex-Valued Convolutional Networks". *Neural Computation* 28.5, pp. 815–825.

Ulicny, M., V. A. Krylov, and R. Dahyot (2019). "Harmonic Networks for Image Classification". In: *British Machine Vision Conference*.

Vaillant, R., C. Monrocq, and Y. LeCun (1994). "Original Approach for the Localisation of Objects in Images". *IEE Proceedings - Vision, Image and Signal Processing* 141.4, pp. 245–250.

Vasconcelos, C., H. Larochelle, V. Dumoulin, N. L. Roux, and R. Goroshin (2020). "An Effective Anti-Aliasing Approach for Residual Networks". arXiv: 2011.10675.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin (2017). "Attention Is All You Need". In: *Advances in Neural Information Processing Systems*.

Vetterli, M. (1986). "Filter Banks Allowing Perfect Reconstruction". *Signal Processing* 10.3, pp. 219–244.

Vetterli, M. (2001). "Wavelets, Approximation, and Compression". *IEEE Signal Processing Magazine* 18.5, pp. 59–73.

Virmaux, A. and K. Scaman (2018). "Lipschitz Regularity of Deep Neural Networks: Analysis and Efficient Estimation". In: *Advances in Neural Information Processing Systems*.

Waibel, A., T. Hanazawa, G. Hinton, K. Shikano, and K. Lang (1989). "Phoneme Recognition Using Time-Delay Neural Networks". *IEEE Transactions on Acoustics, Speech, and Signal Processing* 37.3, pp. 328–339.

Waldspurger, I. (2015). "Wavelet Transform Modulus : Phase Retrieval and Scattering". Doctoral thesis. Ecole normale supérieure, Paris.

— (2016). "Exponential Decay of Scattering Coefficients". In: *2017 International Conference on Sampling Theory and Applications (SampTA)*.

Wang, J., Y. Chen, R. Chakraborty, and S. X. Yu (2020). "Orthogonal Convolutional Neural Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Wang, X., R. Girshick, A. Gupta, and K. He (2018). "Non-Local Neural Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Weiler, M. and G. Cesa (2021). "General E(2)-Equivariant Steerable CNNs". arXiv: 1911.08251.

Wiatowski, T. and H. Bölcskei (2018). "A Mathematical Theory of Deep Convolutional Neural Networks for Feature Extraction". *IEEE Transactions on Information Theory* 64.3, pp. 1845–1866.

Williams, T. and R. Li (2016). "Advanced Image Classification Using Wavelets and Convolutional Neural Networks". In: *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*.

— (2018). "Wavelet Pooling for Convolutional Neural Networks". In: *International Conference on Learning Representations (ICLR)*.

Wu, H., B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, and L. Zhang (2021). "CvT: Introducing Convolutions to Vision Transformers". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

Xie, S., R. Girshick, P. Dollar, Z. Tu, and K. He (2017). "Aggregated Residual Transformations for Deep Neural Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Xie, Z.-M., E.-F. Wang, G.-H. Zhang, G.-C. Zhao, and X.-G. Chen (2004). "Seismic Signal Analysis Based on the Dual-Tree Complex Wavelet Packet Transform". *Acta Seismologica Sinica* 17.1, pp. 117–122.

Xu, J., H. Kim, T. Rainforth, and Y. Teh (2021). "Group Equivariant Subsampling". In: *Advances in Neural Information Processing Systems*.

Xu, Y., J. Weaver, D. Healy, and J. Lu (1994). "Wavelet Transform Domain Filters: A Spatially Selective Noise Filtration Technique". *IEEE Transactions on Image Processing* 3.6, pp. 747–758.

Yamaguchi, K., K. Sakamoto, T. Akabane, and Y. Fujimoto (1990). "A Neural Network for Speaker-Independent Isolated Word Recognition." In: *International Conference on Spoken Language Processing (ICSLP)*. International Conference on Spoken Language Processing (ICSLP).

Ye, J. C., Y. Han, and E. Cha (2018). "Deep Convolutional Framelets: A General Deep Learning Framework for Inverse Problems". *SIAM Journal on Imaging Sciences* 11.2, pp. 991–1048.

Yen, G. G. (2000). "Wavelet Packet Feature Extraction for Vibration Monitoring". *IEEE Transactions on Industrial Electronics* 47.3, pp. 650–667.

Yosinski, J., J. Clune, Y. Bengio, and H. Lipson (2014). "How Transferable Are Features in Deep Neural Networks?" In: *Advances in Neural Information Processing Systems*.

Yu, R. and H. Ozkaramanli (2005). "Hilbert Transform Pairs of Orthogonal Wavelet Bases: Necessary and Sufficient Conditions". *IEEE Transactions on Signal Processing* 53.12, pp. 4723–4725.

Yuan, K., S. Guo, Z. Liu, A. Zhou, F. Yu, and W. Wu (2021). "Incorporating Convolution Designs Into Visual Transformers". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

Zagoruyko, S. and N. Komodakis (2017). "Wide Residual Networks". arXiv: 1605.07146.

Zandi, A., J. Allen, E. Schwartz, and M. Boliek (1995). "CREW: Compression with Reversible Embedded Wavelets". In: *Proceedings DCC '95 Data Compression Conference*.

Zarka, J., F. Guth, and S. Mallat (2021). "Separation and Concentration in Deep Networks". In: *International Conference on Learning Representations (ICLR)*.

Zarka, J., L. Thiry, T. Angles, and S. Mallat (2020). "Deep Network Classification by Scattering and Homotopy Dictionary Learning". In: *International Conference on Learning Representations (ICLR)*.

Zeiler, M. D. and R. Fergus (2014). "Visualizing and Understanding Convolutional Networks". In: *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*.

Zhang, J. and J. Zhang (2018). "An Analysis of CNN Feature Extractor Based on KL Divergence". *International Journal of Image and Graphics* 18.03, p. 1850017.

Zhang, R. (2019). "Making Convolutional Networks Shift-Invariant Again". In: *International Conference on Machine Learning*.

Zhang, S. and S. Mallat (2021). "Maximum Entropy Models from Phase Harmonic Covariances". *Applied and Computational Harmonic Analysis* 53, pp. 199–230.

Zhang, Z., H. Wang, F. Xu, and Y.-Q. Jin (2017). "Complex-Valued Convolutional Neural Network and Its Application in Polarimetric SAR Image Classification". *IEEE Transactions on Geoscience and Remote Sensing* 55.12, pp. 7177–7188.

Zou, D., R. Balan, and M. Singh (2020). "On Lipschitz Bounds of General Convolutional Neural Networks". *IEEE Transactions on Information Theory* 66.3, pp. 1738–1759.

Zou, D. and G. Lerman (2020). "Graph Convolutional Neural Networks via Scattering". *Applied and Computational Harmonic Analysis* 49.3, pp. 1046–1074.

Zou, X., F. Xiao, Z. Yu, Y. Li, and Y. J. Lee (2023). "Delving Deeper into Anti-Aliasing in ConvNets". *International Journal of Computer Vision* 131.1, pp. 67–81.