Equivariant convolutional networks

Cohen, T.S.

Publication date
2021
Document Version
Final published version

Equivariant Convolutional Networks

Taco Cohen

Equivariant Convolutional Networks

Taco Cohen

Equivariant Convolutional Networks

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor

aan de Universiteit van Amsterdam

op gezag van de Rector Magnificus

prof. dr. ir. K.I.J. Maex

ten overstaan van een door het College voor Promoties ingestelde

commissie, in het openbaar te verdedigen in de Agnietenkapel

op woensdag 9 juni 2021, te 16.00 uur

door Taco Sebastiaan Cohen

geboren te Leeuwarden

Promotiecommissie

Promotor:	
Prof. dr. M. Welling	Universiteit van Amsterdam
Overige leden:	
Prof. dr. G.E. Hinton	University of Toronto
Prof. dr. E.P. Verlinde	Universiteit van Amsterdam
Prof. dr. M. Bronstein	Imperial College London
dr. R. Kondor	University of Chicago
dr. E. Bekkers	Universiteit van Amsterdam
dr. L. Dorst	Universiteit van Amsterdam
Prof. dr. J.M. Mooij	Universiteit van Amsterdam

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

Funding for this research was provided by the Netherlands Organisation for Scientific Research (NWO) grant number NAI.14.108 and a 2017 Google European PhD Fellowship in Machine Learning.

Summary

Deep neural networks can solve many kinds of learning problems, but only if a lot of data is available. For many problems (e.g. in medical imaging), it is expensive to acquire a large amount of labelled data, so it would be highly desirable to improve the statistical efficiency of deep learning methods. In this thesis we explore ways to leverage symmetries to improve the ability of convolutional neural networks to generalize from relatively small samples.

We argue and show empirically that in the context of deep learning it is better to learn equivariant rather than invariant representations, because invariant ones lose information too early on in the network. We present a sequence of increasingly general group equivariant convolutional neural networks (G-CNNs), adapted to the particular symmetries of various spaces. Specifically, we present roto-translation equivariant networks for planar images and volumetric signals, rotation equivariant spherical CNNs for analyzing spherical signals such as global weather patterns and omnidirectional images, and gauge equivariant CNNs for the analysis of signals on general manifolds.

We have evaluated these networks on problems such as image classification and segmentation in vision and medical imaging, 3D model classification, detection of extreme weather events, quantum chemistry, and protein structure classification. We show that across the board, G-CNNs outperform conventional translation equivariant CNNs on problems that exhibit symmetries.

In Part II we present a general mathematical theory of G-CNNs. The theory describes convolutional feature spaces as spaces of fields over a manifold, i.e. spaces of sections of an associated vector bundle. Symmetries are described as groups acting on a principal bundle by automorphisms, and layers of the network are described as linear and non-linear equivariant maps between spaces of fields. Through the use of a common mathematical language, an analogy to theoretical physics (especially gauge theory) is established. We show that in general, convolution-like maps arise from symmetry principles, and specifically that each one of the generalized convolutions used in Part I is recovered from symmetry principles as the most general class of linear maps that is equivariant to a certain group of symmetries.

Acknowledgements

First and foremost, I would like to thank Max Welling for his exceptional guidance during my PhD studies. I have learned a lot from Max, and not just about technical matters. For one thing, I have learned the value of enthusiasm and optimism in research, both of which Max has in abundance. Max has provided me with plenty of freedom to pursue my research interests, and provided encouragement when needed, without which the research presented herein would not have materialized. Besides these personal and mentoring qualities, Max also posses deep knowledge across a wide spectrum of machine learning topics and beyond, and has an unparalleled ability to quickly understand a research area and provide insightful feedback on new ideas. I have seen repeatedly how Max is able to get the best out of people, so needless to say I feel extremely fortunate to have had him as my advisor.

I also feel fortunate to have met and worked with many amazing people over the last few years. My co-founders and colleagues at Scyfer, who stuck together through thick and thin. My lab mate Durk Kingma, with whom I shared many interesting conversations and some very fun and memorable trips around the world, from Toronto to Detroit, Irvine, Beijing, and Shanghai. Geoff Hinton, one of the liveliest minds I have come across, with whom I had the great fortune of spending several months working together intensively during my internship at DeepMind. The many talented students I've worked with, including Mario Geiger, Luisa Zintgraf, Pim de Haan, Gabriele Cesa, Marysia Winkels, Jonas Köhler, Jorn Peters, Emiel Hoogeboom, Tim Davidson, Carla Groenland, Adeel Pervez, Maurice Weiler, Berkay Kicanaoglu, Mergahney Mohammed, Bas Veeling, Jim Winkens, Jasper Linmans, Phillip Lippe, Liam Schoneveld, and Marcel Boersma. My colleagues at Qualcomm, and the many wonderful people at the AMLAB, Deepmind, and OpenAI.

During the final stretch of thesis writing, Tobias Diez generously answered a long list of technical questions I had related to the mathematical theory presented in Part II, and provided several pointers to relevant mathematical literature. Similarly, Juan Camilo Orduz kindly provided technical feedback. Of course, any mistakes that remain are entirely my own.

I would like to express my gratitude to my parents for nurturing my curiousity and independence from a young age, and for providing love and support throughout my life so far. Finally, thanks to my friends and family for continuing to drag me away from the computer for much needed fun and relaxation!

CONTENTS

1 Introduction

Human beings and other intelligent animals have the remarkable ability to learn new skills and concepts very quickly. A newborn horse learns to walk in just a few minutes after being born, and children learn the names of visual object categories from just a few examples. In contrast, although the best current machine learning systems can acquire basic motor skills and perceptual capabilities, they require a very large amount of training data to do so.

Because of this limited ability to generalize, the long tail of machine learning problems remains inaccessible. In medical imaging and computer aided diagnosis, for instance, it can be very expensive to collect a large expert-annotated dataset for every imaging device, configuration, resolution, patient population, health condition, etc., rendering many niche applications uneconomical.

In this thesis we aim to improve the statistical efficiency of neural networks by enabling them to exploit the symmetries of learning problems. The emphasis is on problems involving signals on various spaces (e.g. visual images on the plane), for which Convolutional Neural Networks (CNNs) are well suited. We show how the classical CNN (LeCun et al., 1998) can be generalized in a natural way to exploit not just translational symmetries, but many other kinds of symmetries of many other spaces as well. This class of methods, which we refer to generically as G-CNNs, can for instance exploit rotation and reflection symmetries of Euclidean space (Chapter 2, 3), the sphere (Chapter 4), local gauge symmetries like changes of frame of the tangent space or color space symmetries (Chapter 5). Even DNA sequences, which have reverse-complement symmetry, can be analyzed by an appropriate G-CNN (Lunter and Brown, 2018).

In the process of developing these increasingly general methods, an interesting mathematical picture has emerged. Groups are the central object in the mathematical study of symmetry, so it is only natural to consider replacing convolution by group convolution in CNNs, as we do in Chapter 2. Group convolution can be defined for any group or homogeneous space that we can sum or integrate over, and so extensions like Spherical CNNs (Chapter 4) follow naturally. The feature maps of such *Regular* G-CNNs can be interpreted as scalar functions on a group or homogeneous space (a certain kind of manifold), and such functions transform according to the so-called *regular representation* of the group under consideration. From there it is rather natural to ask if we can learn feature maps that represent more general non-scalar *fields* (e.g. vector fields) on a homogeneous space, which leads to the *induced representations* and *Steerable G-CNNs* (Chapter 3). The neatest way to describe such fields and their transformations is using *homogeneous principal bundles* (Covered in Part II, Chapter 9), which in turn leads naturally to considering more general principal bundles such as the frame bundle of a manifold, which is how we came to the idea of *Gauge CNNs* on general manifolds, treated in Chapter 5. Although the relations and commonalities of these ideas should become somewhat clear from reading Part I (Chapters 1 – 6), in Part II (Chapters 7 – 10) we will show explicitly how all of these methods are just special cases of a general mathematical framework that can be presented in an elegant way using the language of fiber bundles.

At this point it is not necessary to know what the terms group, representation, fiber bundle or field mean precisely. It may be interesting to know however that this mathematical language which so neatly describes G-CNNs is also the language in which most modern physical theories are written (Indeed none of the mathematics in this thesis is fundamentally new; Our contribution is entirely in the application to neural networks). Physicists have long recognized that knowledge of symmetries can serve as a powerful inductive bias, greatly constraining the space of admissible theories or models. The same is true in machine learning, and so it is only natural that the same mathematical toolbox can be of service in this field. By the same token, one may expect these tools to be of use in describing the structure and functon of the brain, and indeed this appears to be the case (Petitot, 2017).

In the next section we will provide an accesible introduction to the central concepts of symmetries, representations, and equivariance. This should be sufficient to read Part I, but for those who want to go deeper into the mathematics we provide pointers in Section 1.3. In the last section of this chapter we list our contributions to each chapter, as required by the doctoral regulations.

1.1 Symmetries, Representations & Equivariance

There are three general concepts that play a role throughout the work presented in this thesis: symmetry, representations, and equivariance. Because these terms are only now gradually beginning to enter the lexicon of machine learning research, we will in this section introduce each one and discuss their relation to learning.

1.1.1 Symmetries

A symmetry of an object is a transformation of that object that leaves it unchanged. For instance, we can rotate a perfect square by 90 degrees about its center, without leading to any visible changes. If we consider the set of *all* symmetries of a given object, the following facts are immediately apparent:

1. The identity transformation is always a symmetry (*identity*).
2. Given two symmetry transformations, we can do one after the other (i.e. compose them in an associative manner) and the result is again a symmetry transformation (*closure under composition*).
3. The inverse of a symmetry transformation is also a symmetry (*closure under inverses*).

These are essentially the axioms of a *group*, so this is the key mathematical concept we need to understand symmetry.

In mathematics, the concept of symmetry applies not just to geometrical objects like squares, but to more general abstract mathematical objects. In applications to machine learning, the object of interest could be a label function $L : \mathcal{X} \to \mathcal{C}$ that assigns class-labels $c \in \mathcal{C}$ to each point in the input space $\mathcal{X}$ (e.g. a space of images), or it could be a probability density function $p : \mathcal{X} \to \mathbb{R}^+$. A symmetry of a label function (resp. density) on $\mathcal{X}$ may be defined as a transformation $g : \mathcal{X} \to \mathcal{X}$ that leaves the class label (resp. density) of each point invariant. This can be expressed via the equation $L \circ g = L$, which nicely captures that in accordance with our definition of symmetry, g is a transformation that leaves an object (L in this case) unchanged. The same is expressed diagramatically as follows:

$$\begin{array}{ccc} \mathcal{X} & \xrightarrow{\quad g \quad} & \mathcal{X} \\ & {\scriptstyle L}\searrow \quad \swarrow{\scriptstyle L} & \\ & \mathcal{C} & \end{array} \tag{1.1}$$

All this says is that following g and then L is the same as following L.

If we know about a symmetry group G a priori, or can somehow infer it from data, the learning problem becomes simpler, sometimes dramatically so. For instance, if we observe an image x with label y, we immediately learn about the label of all images in the *orbit* of x, i.e. the set images that we can obtain by applying transformations $g \in G$ to x. Hence, knowledge of a symmetry acts as a multiplier on the size of our dataset (see Figure 1.1). This is not just a figure of speech – in experiments we regularly find that networks that exploit symmetries perform about the same as networks that don't but are trained on a dataset that is C times larger, with the multiplier C being relatively constant across dataset sizes (e.g. Winkels and Cohen (2018a)).

A useful way to think about symmetries $g \in G$ is via the relations they induce on $\mathcal{X}$. We will say that two points $x, y \in \mathcal{X}$ are g-related, and write $x \sim_g y$ if and only if $gx = y$. For instance, if g is a planar rotation, we say that an image y is g-related to an image x if y is the rotation of x by g.

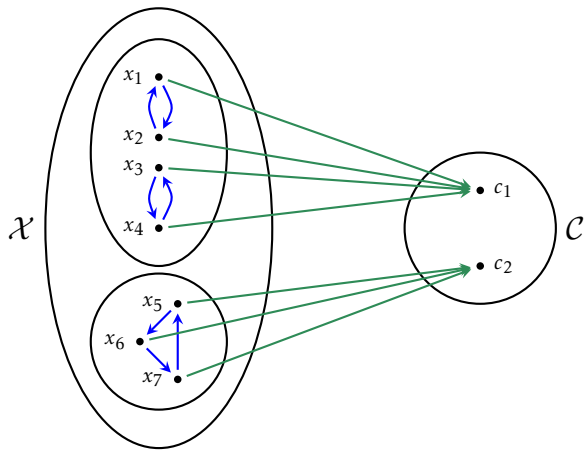


Figure 1.1: *Example of a label function $L : \mathcal{X} \to \mathcal{C}$ (in green) from input space $\mathcal{X}$ to label space $\mathcal{C}$. A symmetry transformation $g : \mathcal{X} \to \mathcal{X}$ of L is drawn in blue. The classes correspond to the preimages $L^{-1}(c_i)$, drawn as ellipses. Since g is a symmetry of L, each class is a union of orbits of g (connected components of the blue graph), and more generally orbits of G. That is, $L^{-1}(c_1) = \{x_1, x_2\} \cup \{x_3, x_4\}$ and $L^{-1}(c_2) = \{x_5, x_6, x_7\}$. Knowing that g is a symmetry of L reduces the minimum number of labels required from 7 to 3 in the noiseless setting.*

If we have a whole group G and consider the relations induced by all of the elements together, we may notice that:

1. Since the identity transformation e is always in G (identity axiom of a group), we have that $x \sim_e x$ for every $x \in \mathcal{X}$.

2. Since $gh \in G$ whenever both $g, h \in G$ (closure under composition), we have that if $x \sim_g y$ and $y \sim_h z$, then $x \sim_{hg} z$.

3. Since $g^{-1} \in G$ whenever $g \in G$ (closure under inverses), we have that if $x \sim_g y$ then $y \sim_{g^{-1}} x$.

This motivates the definition of another relation, this one being an *equivalence* relation. We will say that two points $x, y \in \mathcal{X}$ are G-equivalent, and write $x \sim y$ if and only if there exists $g \in G$ such that $gx = y$, i.e. if x and y are g-related by any $g \in G$. Notice that the axioms of an equivalence relation are satisfied, because: 1) by the identity axiom of a group, we have $x \sim x$ for every $x \in \mathcal{X}$ (the relation is reflexive), 2) by the closure under composition axiom of a group, we have that if $x \sim y$ and $y \sim z$ then $x \sim z$ (the relation is transitive), and 3) by the closure under inverses axiom of a group, the relation is symmetric: $x \sim y$ if and only if $y \sim x$.

We can understand this as follows. Since G is a symmetry group, transformations $g \in G$ all leave something unchanged. A symmetry of a function, such as L, will leave unchanged the *properties* captured by the function, such as for example "x has class c". It stands to reason that if we consider all of the properties left unchanged by G, we obtain an equivalence relation on $\mathcal{X}$: "x and y have all the same G-invariant properties". This is the meaing of G-equivalence.

In theory, one could consider the set of *all* symmetries of a label function, in which case the equivalence classes / orbits induced by the group are the same as the classes. In this case we just need one label per class. So it will not be surprising that in general we won't know the whole symmetry group of L, for if we did, we would not need learning. Nevertheless, it is common to have knowledge of *a* group of symmetries, and in this case we can still gain a lot by exploiting this knowledge. As shown in Figure 1.1, the classes defined by the label function are always unions of the equivalence classes induced by the symmetry group. Hence, in this case we would need only as many labels as there are orbits / equivalence classes.

We would like to mention that this is but one example of how groups and symmetry are relevant to machine learning and artificial intelligence. For instance, it is possible (even common) that the group of symmetries acts not on

the input space but only on some latent space, such as the 3D representation of a scene observed only as a 2D image (there is no 1-to-1 mapping on image space that performs a 3D rotation of the scene (Cohen and Welling, 2015a; Rezende et al., 2016)). Groups also play a role in optimization and inference, via the concept of weight-space symmetries. However, in this thesis we focus on symmetries that act on the input space and feature spaces only.

Finally, we would like to note that the simplicity of the group axioms belie the staggering depth and complexity that is to be found in group theory, a field that touches on many areas of mathematics. Nevertheless, in this thesis we will only make use of elementary group theory. In Section 1.3 we provide references for further study.

1.1.2 Representations

One of the basic ideas behind deep learning is composition of maps. We try to approximate the label function L by composing a number of linear and non-linear layers $\Phi_l : \mathcal{X}_l \to \mathcal{X}_{l+1}$ between *feature spaces* $\mathcal{X}_l$ as $L \approx \Phi_l \circ \cdots \circ \Phi_1$.

In this context it is beneficial to distinguish two notions we have so far conflated: the abstract group of symmetries G and its action on the input space $\mathcal{X} = \mathcal{X}_1$ or a feature space $\mathcal{X}_l$. The abstract group is obtained by taking the symmetries $g : \mathcal{X} \to \mathcal{X}$ and forgetting that they are transformations of $\mathcal{X}$, while remembering how they compose. That is, if $g_1, g_2 : \mathcal{X} \to \mathcal{X}$ compose to form $g_3 = g_1 \circ g_2$, then viewed as abstract group elements we have $g_3 = g_1 g_2$ (with juxtaposition denoting composition in the group).

To recover the connection between G and $\mathcal{X}$, one introduces a *group action*, denoted $\nu : G \times \mathcal{X} \to \mathcal{X}$. To be called a group action, this map has to satisfy $\nu(e, x) = x$ for all $x \in \mathcal{X}$ and $e \in G$ the identity transformation, as well as $\nu(g_1 g_2, x) = \nu(g_1, \nu(g_2, x))$. Then if we set $\nu(g, x) = g(x)$ (remembering that g is a transformation of $\mathcal{X}$), we satisfy the group action axioms and lose nothing. The benefit of making the distinction between abstract group and group action is that now we can speak of the group G acting not just on the input space but also on each of the feature spaces $\mathcal{X}_l$ via a separate action ν_l.

In Figure 1.1 we pictured $\mathcal{X}$ as a finite set and L as a function between finite sets, but in practice it is more common in machine learning to consider feature spaces $\mathcal{X}_l$ that are *vector spaces*. In this case, the appropriate notion of group action is called *group representation*, which is just a linear group action. That is, each group element $g \in G$ is represented by a linear map (matrix) $\rho(g)$ that acts on $\mathcal{X}$. Once again, for such an assignment to be called a representation, it has to satisfy a constraint, namely that for all $g, h \in G$, $\rho(gh) = \rho(g)\rho(h)$, where

gh denotes composition in G and $\rho(g)\rho(h)$ denotes matrix multiplication.

There are many kinds of representations. In our work, we model feature maps of CNNs as spaces of functions or more generally, fields on a manifold. These spaces form (typically infinite dimensional) vector spaces (we can add two functions and multiply them by a scalar). Hence, we focus on representations that act in these functions spaces - the regular and induced representations described in Chapter 2, 3, and Part II of this thesis.

In this thesis we focus exclusively on linear group actions. This may seem limiting, and at odds with the non-linear nature of deep neural networks. However, in many practical applications, the symmetries of interest act linearly on the input space, and there is no reason to non-linearize them in the feature spaces. The reason is that many applications involve *signals* (i.e. functions on some kind of space), and these tend to transform linearly even when the group acts non-linearly on the space over which the signals are defined. For instance, diffeomorphisms act non-linearly on the plane, but linearly (though rather irregularly) on the space of functions on the plane (i.e. images). The same is true in the more general case of *fields* defined on a space, which we use to model feature maps in Steerable G-CNNs.

A further limitation of our work, which we do hope will be addressed at some point in the future, is that we fix the group representations ρ_l associated with each layer in advance. Learning the representations and even the group itself is an interesting and challenging problem whose solution may well fill up several PhD theses.

Group representation theory is not used much in computer science, and even in the rapidly growing literature on equivariant networks it is sometimes avoided. Representation theory may seem rather abstract at first, but in fact it provides us with many practical computational tools. Simply put, representation theory reduces abstract group theory to concrete linear algebra, making it highly appropriate for use in machine learning. As such, we hope more deep learning researchers will take the time to learn and apply it.

1.1.3 Equivariance

In the previous section we stated that the symmetry group G can act on each feature space $\mathcal{X}_l$ in a different manner, via a representation ρ_l. However, if we choose group actions / representations haphazardly, not much will be gained. Adjacent feature spaces $\mathcal{X}_l$ and $\mathcal{X}_{l+1}$ are related by a map (layer) Φ_l, and so the group representations at these layers should be related as well.

To make this compatibility requirement precise, we consider again the relations defined in Section 1.1.1. Specifically, we require that points $x, y \in \mathcal{X}_l$ that are g-related are mapped to g-related points in the next feature space $\mathcal{X}_{l+1}$ (where the g-relatedness relation is defined with respect to ρ_l and ρ_{l+1}, respectively). So it should be the case that $\rho_{l+1}(g)\Phi x = \Phi y$ whenever $\rho_l(g)x = y$. But if $y = \rho_l(g)x$ then $\Phi y = \Phi\rho_l(g)x$, and so the constraint becomes $\rho_{l+1}(g)\Phi x = \Phi\rho_l(g)x$. This must hold for all $x \in \mathcal{X}_l$ and $g \in G$, so we can also write:

$$\rho_{l+1}(g)\,\Phi = \Phi\,\rho_l(g). \tag{1.2}$$

When this constraint is satisfied, we say that Φ is G-equivariant with respect to ρ_l and ρ_{l+1}.

Since all g-relatedness relations are preserved by Φ, so too are G-equivalence relations. If $x \sim y$ then $\Phi x \sim \Phi y$. An equivariant map Φ can in general map two different ($x \neq y$) but G-equivalent ($x \sim y$) inputs to the same output $\Phi x = \Phi y$ without violating the constraint that G-equivalence must be preserved, because by reflexivity $\Phi x \sim \Phi y$ whenever $\Phi x = \Phi y$. In many cases Φ is also free to map points $x \neq y$ in different orbits (i.e. non-G-equivalent points) to the same output, but in that case it should do the same thing for all the points in the orbits of x and y. For instance, if Φ maps an image of a square (x) and an image of a triangle (y) to the same output ($\Phi x = \Phi y$), and G is the rotation group, then Φ should also map g-rotated version of x and y to the same output: $\Phi\rho_l(g)x = \Phi\rho_l(g)y$. This follows automatically from equivariance.

Equivariance is a transitive property: if $\Phi_l : \mathcal{X}_l \to \mathcal{X}_{l+1}$ is equivariant wrt ρ_l and ρ_{l+1}, and $\Phi_{l+1} : \mathcal{X}_{l+1} \to \mathcal{X}_{l+2}$ is equivariant with respect to ρ_{l+1} and ρ_{l+2}, then their composition $\Phi_{l+1} \circ \Phi_l : \mathcal{X}_l \to \mathcal{X}_{l+2}$ is equivariant wrt ρ_l and ρ_{l+2}. Hence, if each layer of our network is equivariant, then our whole network is equivariant, and the relational structure in the input space $\mathcal{X}$ induced by G is lifted into every feature space of the network.

In conventional deep learning, one can compose any two maps $\Phi_l : \mathcal{X}_l \to \mathcal{X}_{l+1}$ and $\Phi_{l+1} : \mathcal{X}_{l+1} \to \mathcal{X}_{l+2}$ as long as the output space of one matches the input space of the other (i.e. they are vector spaces of the same dimensionality). In equivariant deep learning, it is not enough for the codomain of Φ_l to equal the domain of Φ_{l+1} as a vector space; they must be equal as group representations. That is, if Φ_l is equivariant wrt ρ^l_{in} and ρ^l_{out}, and Φ_{l+1} is equivariant wrt ρ^{l+1}_{in} and ρ^{l+1}_{out}, then we can compose them only if $\rho^l_{\text{out}} = \rho^{l+1}_{\text{in}}$. This idea should be familiar to any programmer: we can only compose functions with matching types. Indeed, we can think of a representation $(\mathcal{X}_l, \rho_l)$ as a "geometric type".

The equivariance constriant (Eq. 1.2) limits the admissible maps Φ, and thus acts as an inductive bias. Indeed, if Φ is a learnable linear layer, Eq. 1.2

limits Φ to a linear subspace of the space of linear maps / matrices, because for each g we have a linear constraint $\phi_{l+1}(g)\Phi = \Phi\rho_l(g)$ on Φ. This subspace has a lower dimensionality than the full space of matrices, and since we need a parameter for each dimension, an equivariant map will generally have fewer parameters than a general linear map.

Understanding the space of equivariant linear maps for given input and output representations ρ_l and ρ_{l+1} is a classical problem of representation theory, and an essentially complete understanding can be obtained via Schur's lemma. Moreover, when $\mathcal{X}_l$ are spaces of functions or fields (i.e. feature maps) over some base space (e.g. the plane or the sphere), then it turns out that the equivariance constraint forces us to use a generalized *convolution* layer. This relation between equivariance and convolution is a central theme of the thesis and is investigated in full generality in Part II.

1.2 A Brief History of Symmetry in ML

It has long been recognized that symmetries are important in pattern recognition, computer vision, and machine learning. We will not be able to do justice to the large body of work on this topic, but here we hope to give a sketch of the history of the subject, focusing in particular on highly influential or sadly ignored work from decades past. More recent and closely related work will be discussed in the related work sections of the various chapters in Part I.

Symmetry, in the form of *exchangeability*, is of fundamental importance in Bayesian statistics. The celebrated theorem by De Finetti (1929) shows that an infinite sequence of random variables is exchangeable (i.e. has a distribution invariant under permutation) if and only if the random variables are conditionally i.i.d. given a latent parameter θ, thus motivating the widespread use of parametric models from a symmetry principle. Symmetry is also deeply related to the notion of sufficiency (Diaconis and Freedman, 1984; Bloem-Reddy and Teh, 2019), and motivates equivariant estimation (Eaton, 1989).

Early work on designing equivariant feature detectors for pattern recognition was done by Amari (1978), Kanatani (1990), and Lenz (1990). In the neural networks literature, the famous Group Invariance Theorem for Perceptrons by Minsky and Papert (1987) puts fundamental limitations on the capabilities of (one-layer) perceptrons to learn invariants. This was one of the primary motivations for studying multi-layer architectures (Sejnowski et al., 1986; Shawe-Taylor, 1989; Shawe-Taylor, 1993), which ultimately led to deep learning.

The first works to take representation theoretical view on invariant and equivariant neural networks was (Wood and Shawe-Taylor, 1996). Although we were initially unaware of this work, the setup used in this thesis is similar to that used by Wood & Shawe-Taylor (i.e. feature spaces are group representations and layers are equivariant maps). The main difference is that we consider feature spaces of signals over a space, which leads to a particular kind of representation (regular and induced representations) and a particular kind of linear maps (convolutions). Additionally, we consider not just finite but also infinite discrete and continuous groups. Convolutional networks are equivariant to translations, and were first explored by (Fukushima and Miyake, 1982; LeCun et al., 1990).

1.3 Recommended mathematical reading

In this thesis we make use of various mathematical concepts such as manifolds, groups, and group representations, which are not part of the standard machine learning curriculum. Throughout Part I we have tried to write as accessibly as possible, and have included in each paper / chapter a brief review of the relevant concepts. Nevertheless, this material is unlikely to be sufficient for learning all the mathematics from scratch. For this reason we had initially planned to include a chapter on mathematical preliminaries, but decided to abandon this effort due to time constraints. As a substitute, we have opted to include a list of reading recommendations for those motivated to dig deeper. It is hard to find a single source that covers everything we need and not much more, while working at a reasonable level of mathematical sophistication, but we hope to include something for all readers. Perhaps the closest to a comprehensive yet accessible source is the recent paper by Esteves (2020), which is written specifically for those wanting to learn the mathematics of group equivariant neural networks.

Carter (2009) gives a very elementary visual introduction to group theory. It is perhaps unnecessary to mention, but the wikipedia pages on group, group action, subgroup, normal subgroup, coset, and quotient group provide a compact and fairly accessible introduction to some of the key concepts as well. Goodman and Wallach (2009) have written a very comprehensive introduction to groups, representations and invariants.

The PhD thesis of Kondor (2008) is aimed at a machine learning audience and contains a very readable introduction to many of the concepts we use in our work, including finite groups, topological groups, Lie groups, represen-

tation theory, and non-commutative harmonic analysis, as well as a range of applications to machine learning. The books by Kanatani (1990) and Lenz (1989) cover applications of group representation theory to vision and image processing. Diaconis (1988) covers applications to probability and statistics.

For representations of finite groups, we recommend chapters 1-8 of the classic text by Serre (1977). A more modern and general treatment of representation theory can be found in (Etingof et al., 2011).

For representations of topological groups and non-commutative harmonic analysis, we recommend (Folland, 1995; Sugiura, 1990; Gurarie, 1992). Induced representations of locally compact groups are studied in detail in (Kaniuth and Taylor, 2013).

In Part I of this thesis we do not make much use of differential geometry or fiber bundles, but these do play a role in Part II. To learn about these and related topics in mathematical physics, we recommend (Rudolph and Schmidt, 2012; Rudolph and Schmidt, 2017; Hamilton, 2017).

1.4 Outline & Contibutions

This thesis consists of two parts. In Part I, we present a number of equivariant convolutional networks: Regular G-CNNs, Steerable G-CNNs, Spherical CNNs, Gauge CNNs, and a special case of Gauge CNNs, the Icosahedral CNN. Each method is presented in a chapter based on a conference paper. Part II presents a general theory of equivariant CNNs, that covers all of the methods presented in Part I and derives the various convolutions as the unique linear maps that respect a certain symmetry. This part has not been published before, though some of the results on homogeneous G-CNNs can also be found in (Cohen et al., 2018b; Cohen et al., 2018a).

In concordance with the Doctorate Regulations of the University of Amsterdam, we list below for each chapter what publications it is based on, and what my personal contributions to each project have been.

Chapter 1 Introduction. Not published before, and written from scratch for this thesis by me.

Chapter 2 is based on:

T.S. Cohen & M. Welling, *Group Equivariant Convolutional Networks.* Proceedings of the International Conference on Machine Learning (ICML), 2016.

Personal contributions: original idea to use group convolution, mathematical theory, implementation & experiments, writing.

Chapter 3 is based on:

T.S. Cohen & M. Welling, *Steerable CNNs.* International Conference on Learning Representations (ICLR), 2017.

Personal contributions: original idea to generalize from regular to induced representation, mathematical theory, implementation & experiments, writing.

Chapter 4 is based on:

T.S. Cohen, M. Geiger, J. Koehler, M. Welling, *Spherical CNNs.* ICLR 2018

T.S. Cohen, M. Geiger, J. Koehler, M. Welling, *Convolutional Networks for Spherical Signals.* In Principled Approaches to Deep Learning Workshop ICML 2017.

Personal contributions: original idea to generalize regular G-CNNs to the sphere, mathematical theory, first implementation of G-FFT and convolution in numpy, majority of the writing. Optimized PyTorch implementation of G-FFT and convolution as well as all experiments were performed by my co-authors Mario Geiger & Jonas Koehler.

Chapter 5 & 6 are based on

T.S. Cohen, M. Weiler, B. Kicanaoglu, M. Welling, *Gauge Equivariant Convolutional Networks and the Icosahedral CNN*, Proceedings of the International Conference on Machine Learning (ICML), 2019.

Personal contributions: the initial idea that replacing the homogeneous principal bundle by the frame bundle makes it possible to build G-CNNs on general manifolds. Development of mathematical theory presented in this paper together with Maurice Weiler. Idea to implement gauge CNNs for the icosahedron using conv2d. Majority of the writing. The implementation and optimization of Icosahedral CNNs was done by Berkay, Maurice, and myself. Experiments were performed by my co-authors Berkay and Maurice.

Part II This part I wrote from scratch for the PhD thesis. Some of the results on homogeneous G-CNNs can be found in:

T.S. Cohen, M. Geiger, M. Weiler, *Intertwiners between Induced Representations (with applications to the theory of equivariant neural networks)*, ArXiv preprint 1803.10743, 2018.

T.S. Cohen, M. Geiger, M. Weiler, *A General Theory of Equivariant CNNs on Homogeneous Spaces*, NeurIPS 2019.

The main results on intertwiners between induced representations are

known in the mathematics literature under the name "Mackey Theory" (Mackey, 1952).

Personal contribution to these papers: these papers are mostly my own work, with some help from my co-authors. My contributions include the idea to use fiber bundles to describe homogeneous G-CNNs, and nearly all of the mathematical results and writing for both papers. Mario Geiger first suggested to use a matrix-valued kernel and derived and solved the kernel constraint for the case of SE(3). Both Geiger and Weiler contributed further via discussions, proof reading & editing of the papers. Maurice Weiler performed a literature review for the second paper (table 1, appendix D), and wrote section 4 on implementation.

Outtakes

In order to keep the length of this thesis reasonable and to stick to a single coherent topic, I decided not to include a number of projects and publications completed during the PhD. Here we give a brief overview of these.

Extensions and Applications of G-CNNs

Together with students I worked on a number of extensions and applications of G-CNNs. In these papers we have showed that G-CNNs are very effective for medical imaging problems such as lung nodule detection in CT scans and segmentation of histopathology slides, achieving 2x - 10x better data efficiency (i.e. a G-CNN trained on N data points achieves roughly the same performance as a well tuned CNN trained with data augmentation on $k \times N$ data points).

1. E. Hoogeboom, J.W.T. Peters, T.S. Cohen, M. Welling, *HexaConv*, ICLR 2018.

2. B.S. Veeling, J. Linmans, J. Winkens, T.S. Cohen, M. Welling, *Rotation Equivariant CNNs for Digital Pathology*. International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI), 2018.

3. J. Winkens, J. Linmans, B.S. Veeling, T.S. Cohen, M. Welling, *Improved Semantic Segmentation for Histopathology Using Rotation Equivariant Convolutional Networks*. International Conference on Medical Imaging with Deep Learning (MIDL workshop), 2018.

4. J. Linmans, J. Winkens, B.S. Veeling, T.S. Cohen, M. Welling, *Sample Efficient Semantic Segmentation Using Rotation Equivariant Convolutional Networks*, FAIM/ICML

Workshop on Towards learning with limited labels: Equivariance, Invariance, and Beyond, 2018.

5. M. Winkels, T.S. Cohen, *3D Group-Equivariant Neural Networks for Octahedral and Square Prism Symmetry Groups*, FAIM/ICML Workshop on Towards learning with limited labels: Equivariance, Invariance, and Beyond, 2018.

6. M. Winkels, T.S. Cohen, *3D G-CNNs for Pulmonary Nodule Detection*. International Conference on Medical Imaging with Deep Learning (MIDL), 2018.

7. M. Winkels, T.S. Cohen, *Pulmonary Nodule Detection in CT Scans with Equivariant CNNs*, Medical Image Analysis, 2018.

8. M. Weiler, W. Boomsma, M. Geiger, M. Welling, T.S. Cohen, *3D Steerable CNNs: Learning Rotationally Equivariant Features in Volumetric Data*, Advances in Neural Information Processing Systems (NeurIPS), 2018

Personal contributions: In each of these cases my role was primarily that of an advisor. In most cases I contributed to the writing as well. Credit for these works goes primarily to my co-authors.

Disentangling by Irreducible Representations

My first two publications were about formalizing the notion of disentangling. In "Learning the Irreducible Representations of Commutative Lie Groups" (Cohen and Welling, 2014), I made an attempt to give a mathematical definition of (one sort of) "disentangling" (Bengio et al., 2013) as the reduction of a representation into irreducible representations (an idea borrowed from Physics, and applied to computer vision by Kanatani (1990)).

In this paper we defined a single layer linear model (not a deep network), and this layer was not convolutional but fully connected. Hence it does not fit the *equivariant convolutional* theme of this thesis. It also has limited applicability in the era of deep learning, but nevertheless contains a number of ingredients used also in several later papers. In particular, in Steerable CNNs (Cohen and Welling, 2017) (Chapter 3), we define a deep convolutional network where each fiber (feature vector attached to a spatial position) transforms according to a (typically reduced) representation of a group H. There we also relate this notion of disentangling to Hinton's idea of capsules (Hinton et al., 2018).

In "Transformation Properties of Learned Visual Representations" (Cohen and Welling, 2015a), published at ICLR 2015, we extended the idea of disentangling via irreducible representations to the partially observable case. The

basic idea is that by requiring the learned representation of a video stream to transform as a representation of the Euclidean group under Euclidean motions of the camera or object, one can learn to infer $3D$ from $2D$ without further inductive biasses. In this paper we also relate disentangling in the representation-theoretic sense to disentangling as learning a factorized probability distribution. Although the experimental results obtained in this paper are unimpressive by today's standards, the basic idea has been implemented successfully using modern amortized inference techniques and GANs (Rezende et al., 2016; Nguyen-Phuoc et al., 2019). The idea of defining disentangling in terms of group-theoretical notions has been further pursued as well (Higgins et al., 2018).

Harmonic Exponential Families

In the paper "Harmonic Exponential Families on Manifolds" (Cohen and Welling, 2015b), published in ICML 2015, we defined a class of exponential family distributions on homogeneous spaces G/H whose sufficient statistics are given by matrix elements of irreducible representations of the associated group G. Inference and learning in these models can be performed relatively efficiently using a generalized Fourier transform.

Miscellaneous collaborative projects

Together with students and collaborators, I also worked on:

1. Visualizing neural network decisions (Zintgraf et al., 2017; Zintgraf et al., 2016).
2. Explanation of classifier decision for interpretation of microbiota-based diagnostics (Eck et al., 2017)
3. 3D scattering transforms for disease classification in neurimaging with applications to Alzheimer's disease and HIV (Adel et al., 2017)
4. Teacher-student curriculum learning (Matiisen et al., 2019)
5. L. Falorsi, P. de Haan, T. R. Davidson, N. De Cao, M. Weiler, P. Forré and T. S. Cohen, Explorations in Homeomorphic Variational Auto-Encoding, ICML Workshop on Theoretical Foundations and Applications of Generative Models, 2018

In all of these projects, I contributed some ideas and provided guidance, but the large majority of the credit goes to my collaborators.

Part I

Equivariant Convolutional Networks

2

Regular G-CNNs

Based on: (Cohen and Welling, 2016).

The subject of this chapter is our first and most natural generalization of the classical convolutional network: the group equivariant convolutional neural network (G-CNN). From a mathematical perspective, the key observation underlying this work is that convolution can be defined for any group G, not just the group of translations. Once this is understood, it is natural to suppose that we can develop a convolutional network for any group G, and indeed this turns out to be a very fruitful idea.

We develop the idea for general discrete groups G, and illustrate it using the group of rotations, reflections, and translations of the pixel grid. This moderately abstract approach has already paid off: Since the publication of the article that forms the basis of this chapter (Cohen and Welling, 2016), we have developed 3D G-CNNs (Winkels and Cohen, 2018a) (where the symmetries are already so complicated that an algebraic understanding is indispensible), and others have for instance developed G-CNNs equivariant to the reverse-substitution symmetries of DNA (e.g. ACCTTT $\mapsto$ AAAGGT) (Lunter and Brown, 2018). Spherical CNNs (Cohen et al., 2018c) (Chapter 4) are just a continuous variant of G-CNNs for the rotation group SO(3). Indeed, the mathematics describing spherical CNNs and in even regular G-CNNs on general homogeneous spaces is just the continuous analogue of what is described in this chapter (replacing summation by Haar integration (Kondor and Trivedi, 2018; Cohen et al., 2018a)).

Empirically, we have found regular discrete G-CNNs to consistently and significantly improve results compared to conventional CNNs, even when data

augmentation is used. The results on CIFAR and rotated MNIST presented here are no longer state of the art, but they show that simply replacing convolutions by group convolutions improves the accuracy of the model. Similarly, G-CNNs achieved a significant improvement in statistical efficiency on problems involving histopathology images (Veeling et al., 2018), CT scans (Winkels and Cohen, 2018a), and aerial photographs (Hoogeboom et al., 2018).

This chapter is organized as follows. After discussing related work in section 2.1, we recall a number of mathematical concepts in section 2.2 that allow us to define and analyze the group convolution at a general level. In section 2.3, we analyze the equivariance properties of standard CNNs, and show that they are equivariant to translations but may fail to equivary with more general transformations. Then, in section 2.4 we define G-CNNs by analogy to standard CNNs (the latter being the G-CNN for the translation group). We show that G-convolutions, as well as various kinds of layers used in modern CNNs, such as pooling, arbitrary pointwise nonlinearities, batch normalization and residual blocks are all equivariant, and thus compatible with G-CNNs. In section 2.5 we provide concrete implementation details for group convolutions. In section 2.6 we report experimental results on MNIST-rot and CIFAR10. We show that replacing planar convolutions with G-convolutions consistently improves results without additional tuning. In section 2.7 we provide a discussion of these results and consider several extensions of the method, before concluding in section 2.8.

2.1 Related Work

There is a large body of work on invariant representations. Invariance can be achieved by pose normalization using an equivariant detector (Lowe, 2004; Jaderberg et al., 2015) or by averaging a possibly nonlinear function over a group (Reisert, 2008; Skibbe, 2013; Manay et al., 2006; Kondor, 2007).

Scattering convolution networks use wavelet convolutions, nonlinearities and group averaging to produce stable invariants (Bruna and Mallat, 2013). Scattering networks have been extended to use convolutions on the group of translations, rotations and scalings, and have been applied to object and texture recognition (Sifre and Mallat, 2013; Oyallon and Mallat, 2015).

A number of recent works have addressed the problem of learning or constructing equivariant representations. This includes work on transforming autoencoders (Hinton et al., 2011), equivariant Boltzmann machines (Kivinen and Williams, 2011; Sohn and Lee, 2012), equivariant descriptors (Schmidt

and Roth, 2012), and equivariant filtering (Skibbe, 2013).

Lenc and Vedaldi (2015) show that the AlexNet CNN (Krizhevsky et al., 2012) trained on imagenet spontaneously learns representations that are equivariant to flips, scaling and rotation. This supports the idea that equivariance is a good inductive bias for deep convolutional networks. Agrawal et al. (2015) show that useful representations can be learned in an unsupervised manner by training a convolutional network to be equivariant to ego-motion.

Anselmi et al. (2014) and Anselmi et al. (2015) use the theory of locally compact topological groups to develop a theory of statistically efficient learning in sensory cortex. This theory was implemented for the commutative group consisting of time- and vocal tract length shifts for an application to speech recognition by Zhang et al. (2015).

Gens and Domingos (2014) proposed an approximately equivariant convolutional architecture that uses sparse, high-dimensional feature maps to deal with high-dimensional groups of transformations. Dieleman et al. (2015) showed that rotation symmetry can be exploited in convolutional networks for the problem of galaxy morphology prediction by rotating feature maps, effectively learning an equivariant representation. This work was later extended (Dieleman et al., 2016) and evaluated on various computer vision problems that have cyclic symmetry.

2.2 Preliminaries

In this section we will define the groups $p4$ (rotation + translation) and $p4m$ (rotation + reflection + translation) that we will use as examples throughout the article. Then, we will explain how we think of images and feature maps: as functions on the plane or on a group. For a more detailed introduction to groups and other relevant concepts, see the recommended reading in Sec. 1.3.

2.2.1 The group $p4$

The group $p4$ consists of all compositions of translations and rotations by 90 degrees about any center of rotation in a square grid. A convenient parameterization of this group in terms of three integers r, u, v is

$$g(r,u,v) = \begin{bmatrix} \cos(r\pi/2) & -\sin(r\pi/2) & u \\ \sin(r\pi/2) & \cos(r\pi/2) & v \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.1}$$

where $0 \leq r < 4$ and $(u, v) \in \mathbb{Z}^2$. The group operation is given by matrix multiplication.

The composition and inversion operations could also be represented directly in terms of integers (r, u, v), but the equations are cumbersome. Hence, our preferred method of composing two group elements represented by integer tuples is to convert them to matrices, multiply these matrices, and then convert the resulting matrix back to a tuple of integers (using the atan2 function to obtain r).

The group $p4$ acts on points in $\mathbb{Z}^2$ (pixel coordinates) by multiplying the matrix $g(r, u, v)$ by the homogeneous coordinate vector $x(u', v')$ of a point (u', v'):

$$gx \simeq \begin{bmatrix} \cos(r\pi/2) & -\sin(r\pi/2) & u \\ \sin(r\pi/2) & \cos(r\pi/2) & v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} \tag{2.2}$$

2.2.2 The group $p4m$

The group $p4m$ consists of all compositions of translations, mirror reflections, and rotations by 90 degrees about any center of rotation in the grid. Like $p4$, we can parameterize this group by integers:

$$g(m, r, u, v) = \begin{bmatrix} (-1)^m \cos(\frac{r\pi}{2}) & -(-1)^m \sin(\frac{r\pi}{2}) & u \\ \sin(\frac{r\pi}{2}) & \cos(\frac{r\pi}{2}) & v \\ 0 & 0 & 1 \end{bmatrix},$$

where $m \in \{0, 1\}$, $0 \leq r < 4$ and $(u, v) \in \mathbb{Z}^2$. The reader may verify that this is indeed a group.

Again, composition is most easily performed using the matrix representation. Computing r, u, v from a given matrix g can be done using the same method we use for $p4$, and for m we have $m = \frac{1}{2}(1 - \det(g))$.

2.2.3 Feature maps

We model images and stacks of feature maps in a CNN as functions $f : X \to \mathbb{R}^K$ supported on a bounded domain. Here, X is some kind of discrete or continuous space like the pixel coordinates of an image (Z^2) or voxel grid (Z^3), the plane $\mathbb{R}^2$ or sphere S^2. At each pixel coordinate $(p, q) \in \mathbb{Z}^2$, the stack of feature maps returns a K-dimensional vector $f(p, q)$, where K denotes the number of channels.

Although the feature maps must always be stored in finite arrays, modeling them as functions that extend to infinity (while being non-zero on a finite region only) simplifies the mathematical analysis of CNNs.

We will be concerned with transformations of the feature maps, so we introduce the following notation for a transformation g acting on a set of feature maps:

$$[L_g f](x) = [f \circ g^{-1}](x) = f(g^{-1}x) \tag{2.3}$$

Computationally, this says that to get the value of the g-transformed feature map $L_g f$ at the point x, we need to do a lookup in the original feature map f at the point $g^{-1}x$, which is the unique point that gets mapped to x by g. This operator L_g defines a group representation, because it is linear and satisfies

$$L_g L_h = L_{gh}. \tag{2.4}$$

If g represents a pure translation $t = (u, v) \in \mathbb{Z}^2$ then $g^{-1}x$ simply means $x - t$. The inverse on g in equation 2.3 ensures that the function is shifted in the positive direction when using a positive translation, and that L_g satisfies the criterion for being a homomorphism (eq. 2.4) even for transformations g and h that do not commute (i.e. $gh \neq hg$).

As will be explained in section 2.4.1, feature maps in a G-CNN are functions on the group G, instead of functions on the group $\mathbb{Z}^2$. For functions on G, the definition of L_g is still valid if we simply replace x (an element of $\mathbb{Z}^2$) by h (an element of G), and interpret $g^{-1}h$ as composition.

It is easy to mentally visualize a planar feature map $f : \mathbb{Z}^2 \to \mathbb{R}$ undergoing a transformation, but we are not used to visualizing functions on groups. To visualize a feature map or filter on $p4$, we plot the four patches associated with the four pure rotations on a circle, as shown in figure 2.1 (left). Each pixel in this figure has a rotation coordinate $r = 0, 1, 2, 3$ (the patch in which the pixel appears), and two translation coordinates $(u, v) \in \mathbb{Z}^2$ (the pixel position within the patch).

When we apply the 90 degree rotation r to a function on $p4$ using L_r, each planar patch follows its red r-arrow (thus incrementing the rotation coordinate by 1 (mod 4)), and simultaneously undergoes a 90-degree rotation. The result of this operation is shown on the right of figure 2.1. As we will see in section 2.4, a $p4$ feature map in a $p4$-CNN undergoes exactly this motion under rotation of the input image.

For $p4m$, we can make a similar plot, shown in figure 2.2. A $p4m$ function has 8 planar patches, each one associated with a mirroring m and rotation r.

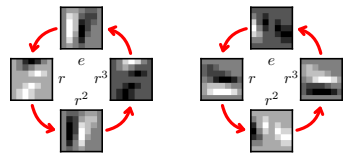


Figure 2.1: *A p4 feature map and its rotation by r.*

Besides red rotation arrows, the figure now includes small blue reflection lines (which are undirected, since reflections are self-inverse).

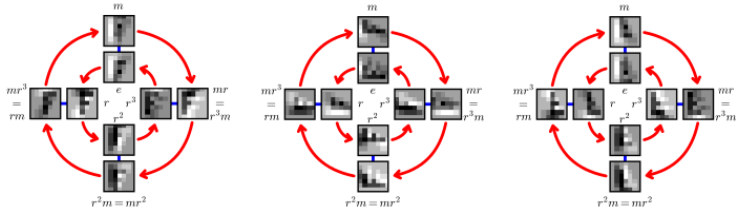


Figure 2.2: *A p4m feature map (left), the same feature map rotated by r (center), and flipped by f (right).*

Upon rotation of a $p4m$ function, each patch again follows its red r-arrows and undergoes a 90 degree rotation. Under a mirroring, the patches connected by a blue line will change places and undergo the mirroring transformation.

This rich transformation structure arises from the group operation of $p4$ or $p4m$, combined with equation 2.3 which describes the transformation of a function on a group.

Finally, we define the *involution* of a feature map, which will appear in section 2.4.1 when we study the behavior of the G-convolution, and which also appears in the gradient of the G-convolution. We have:

$$f^*(g) = f(g^{-1}) \tag{2.5}$$

For $\mathbb{Z}^2$ feature maps the involution is just a point reflection, but for G-feature maps the meaning depends on the structure of G. In all cases, $f^{**} = f$.

2.3 Equivariance properties of CNNs

In this section we recall the definitions of the convolution and correlation operations used in conventional CNNs, and show that these operations are equivariant to translations but not to other transformations such as rotation. This is certainly well known and easy to see by mental visualization, but deriving it explicitly will make it easier to follow the derivation of group equivariance of the group convolution defined in the next section.

At each layer l, a regular convnet takes as input a stack of feature maps $f : \mathbb{Z}^2 \to \mathbb{R}^{K^l}$ and convolves or correlates it with a set of K^{l+1} filters $\psi^i : \mathbb{Z}^2 \to \mathbb{R}^{K^l}$:

$$\begin{aligned}[f * \psi^i](x) &= \sum_{y \in \mathbb{Z}^2} \sum_{k=1}^{K^l} f_k(y) \psi_k^i(x - y) \\ [f \star \psi^i](x) &= \sum_{y \in \mathbb{Z}^2} \sum_{k=1}^{K^l} f_k(y) \psi_k^i(y - x)\end{aligned} \tag{2.6}$$

We will use the correlation in the forward pass, and refer generically to both operations as "convolution".

Using the substitution $y \to y+t$, and leaving out the summation over feature maps for clarity, we see that a translation followed by a correlation is the same as a correlation followed by a translation:

$$\begin{aligned}[[L_t f] \star \psi](x) &= \sum_y f(y - t) \psi(y - x) \\ &= \sum_y f(y) \psi(y + t - x) \\ &= \sum_y f(y) \psi(y - (x - t)) \\ &= [L_t [f \star \psi]](x).\end{aligned} \tag{2.7}$$

And so we say that "correlation is an equivariant map for the translation group", or that "correlation and translation commute". Using an analogous computation one can show that also for the convolution, $[L_t f] * \psi = L_t [f * \psi]$.

Although convolutions are equivariant to *translation*, they are not equivariant to other isometries of the sampling lattice. For instance, as shown in the

supplementary material, rotating the image and then convolving with a fixed filter is not the same as first convolving and then rotating the result:

$$[[L_r f] \star \psi](x) = L_r[f \star [L_{r^{-1}} \psi]](x) \tag{2.8}$$

In words, this says that the correlation of a rotated image $L_r f$ with a filter ψ is the same as the rotation by r of the original image f convolved with the inverse-rotated filter $L_{r^{-1}} \psi$. Hence, if an ordinary CNN learns rotated copies of the same filter, the *stack* of feature maps is equivariant, although individual feature maps are not.

2.4 Group Equivariant Networks

In this section we will define the three layers used in a G-CNN (G-convolution, G-pooling, nonlinearity) and show that each one commutes with G transformations of the domain of the image.

2.4.1 G-Equivariant correlation

The correlation (eq. 2.6) is computed by *shifting* a filter and then computing a dot product with the feature maps. By replacing the shift by a more general transformation from some group G, we get the G-correlation used in the first layer of a G-CNN:

$$[f \star \psi](g) = \sum_{y \in \mathbb{Z}^2} \sum_k f_k(y) \psi_k(g^{-1} y). \tag{2.9}$$

Notice that both the input image f and the filter ψ are functions of the plane $\mathbb{Z}^2$, but the feature map $f \star \psi$ is a function on the discrete group G (which may contain translations as a subgroup). Hence, for all layers after the first, the filters ψ must also be functions on G, and the correlation operation becomes

$$[f \star \psi](g) = \sum_{h \in G} \sum_k f_k(h) \psi_k(g^{-1} h). \tag{2.10}$$

The equivariance of this operation is derived in complete analogy to eq. 2.7,

now using the substitution $h \to uh$:

$$\begin{aligned}
[[L_u f] \star \psi](g) &= \sum_{h \in G} \sum_k f_k(u^{-1}h)\psi(g^{-1}h) \\
&= \sum_{h \in G} \sum_k f(h)\psi(g^{-1}uh) \\
&= \sum_{h \in G} \sum_k f(h)\psi((u^{-1}g)^{-1}h) \\
&= [L_u[f \star \psi]](g)
\end{aligned} \tag{2.11}$$

The equivariance of eq. 2.9 is derived similarly. Note that although equivariance is expressed by the same formula $[L_u f] \star \psi = L_u[f \star \psi]$ for both first-layer G-correlation (eq. 2.9) and full G-correlation (2.10), the meaning of the operator L_u is different: for the first layer correlation, the inputs f and ψ are functions on $\mathbb{Z}^2$, so $L_u f$ denotes the transformation of such a function, while $L_u[f \star \psi]$ denotes the transformation of the feature map, which is a function on G. For the full G-correlation, both the inputs f and ψ and the output $f \star \psi$ are functions on G.

Note that if G is not commutative, neither the G-convolution nor the G-correlation is commutative. However, the feature maps $\psi \star f$ and $f \star \psi$ are related by the involution (eq. 2.5):

$$f \star \psi = (\psi \star f)^*. \tag{2.12}$$

Since the involution is invertible (it is its own inverse), the information content of $f \star \psi$ and $\psi \star f$ is the same. However, $f \star \psi$ is more efficient to compute when using the method described in section 2.5, because transforming a small filter is faster than transforming a large feature map.

It is customary to add a bias term to each feature map in a convolution layer. This can be done for G-conv layers as well, as long as there is only one bias per G-feature map (instead of one bias per spatial feature plane within a G-feature map). Similarly, batch normalization (Ioffe and Szegedy, 2015) should be implemented with a single scale and bias parameter per G-feature map in order to preserve equivariance. The sum of two G-equivariant feature maps is also G-equivariant, thus G-conv layers can be used in highway networks and residual networks (Srivastava et al., 2015a; He et al., 2016a).

2.4.2 Pointwise nonlinearities

Equation 2.11 shows that G-correlation preserves the transformation properties of the previous layer. What about nonlinearities and pooling?

Recall that we think of feature maps as functions on G. In this view, applying a nonlinearity $\nu : \mathbb{R} \to \mathbb{R}$ to a feature map amounts to *function composition*. We introduce the composition operator

$$C_\nu f(g) = [\nu \circ f](g) = \nu(f(g)). \tag{2.13}$$

which acts on functions by *post*-composing them with ν.

Since the left transformation operator L acts by *pre*-composition, C and L commute:

$$C_\nu L_h f = \nu \circ [f \circ h^{-1}] = [\nu \circ f] \circ h^{-1} = L_h C_\nu f, \tag{2.14}$$

so the rectified feature map inherits the transformation properties of the previous layer.

2.4.3 Subgroup pooling and coset pooling

In order to simplify the analysis, we split the pooling operation into two steps: the pooling itself (performed without stride), and a subsampling step. The non-strided max-pooling operation applied to a feature map $f : G \to \mathbb{R}$ can be modeled as an operator P that acts on f as

$$Pf(g) = \max_{k \in gU} f(k), \tag{2.15}$$

where $gU = \{gu \,|\, u \in U\}$ is the g-transformation of some pooling domain $U \subset G$ (typically a neighborhood of the identity transformation). In a regular convnet, U is usually a 2×2 or 3×3 square including the origin $(0, 0)$, and g is a translation.

As shown in the supplementary material, pooling commutes with L_h:

$$PL_h = L_h P \tag{2.16}$$

Since pooling tends to reduce the variation in a feature map, it makes sense to sub-sample the pooled feature map, or equivalently, to do a "pooling with stride". In a G-CNN, the notion of "stride" is generalized by subsampling on a subgroup $H \subset G$. That is, H is a subset of G that is itself a group (i.e. closed under multiplication and inverses). The subsampled feature map is then equivariant to H but not G.

In a standard convnet, pooling with stride 2 is the same as pooling and then subsampling on $H = \{(2i, 2j)|(i, j) \in \mathbb{Z}^2\}$ which is a subgroup of $G = \mathbb{Z}^2$. For the $p4$-CNN, we may subsample on the subgroup H containing all 4 rotations, as well as shifts by multiples of 2 pixels.

We can obtain full G-equivariance by choosing our pooling region U to be a subgroup $H \subset G$. The pooling domains gH that result are called *cosets* in group theory. The cosets partition the group into non-overlapping regions. The feature map that results from pooling over cosets is invariant to the *right*-action of H, because the cosets are similarly invariant ($ghH = gH$). Hence, we can arbitrarily choose one coset representative per coset to subsample on. The feature map that results from coset pooling may be thought of as a function on the quotient space G/H, in which two transformations are considered equivalent if they are related by a transformation in H.

As an example, in a $p4$ feature map, we can pool over all four rotations at each spatial position (the cosets of the subgroup R of rotations around the origin). The resulting feature map is a function on $\mathbb{Z}^2 \cong p4/R$, i.e. it will transform in the same way as the input image. Another example is given by a feature map on $\mathbb{Z}$, where we could pool over the cosets of the subgroup $n\mathbb{Z}$ of shifts by multiples of n. This gives a feature map on $\mathbb{Z}/n\mathbb{Z}$, which has a cyclic transformation law under translations.

This concludes our analysis of G-CNNs. Since all layer types are equivariant, we can freely stack them into deep networks and expect G-conv parameter sharing to be effective at arbitrary depth.

2.5 Efficient Implementation

Computing the G-convolution for a discrete group involves nothing more than indexing arithmetic and inner products, so it can be implemented straightforwardly using a loop or as a parallel GPU kernel. Here we present the details for a G-convolution implementation that can leverage recent advances in fast computation of planar convolutions (Mathieu et al., 2014; Vasilache et al., 2015; Lavin and Gray, 2016).

A plane symmetry group G is called *split* if any transformation $g \in G$ can be decomposed into a translation $t \in \mathbb{Z}^2$ and a transformation s in the stabilizer of the origin (i.e. s leaves the origin invariant). For the group $p4$, we can write $g = ts$ for t a translation and s a rotation about the origin, while $p4m$ splits into translations and rotation-flips. Using this split of G and the fact

that $L_g L_h = L_{gh}$, we can rewrite the G-correlation (eq. 2.9 and 2.10) as follows:

$$f \star \psi(ts) = \sum_{h \in X} \sum_{k} f_k(h) L_t [L_s \psi_k(h)] \tag{2.17}$$

where $X = \mathbb{Z}^2$ in layer one and $X = G$ in further layers.

Thus, to compute the $p4$ (or $p4m$) correlation $f \star \psi$ we can first compute $L_s\psi$ ("filter transformation") for all four rotations (or all eight rotation-flips) and then call a fast planar correlation routine on f and the augmented filter bank.

The computational cost of the algorithm presented here is roughly equal to that of a planar convolution with a filter bank that is the same size as the augmented filter bank used in the G-convolution, because the cost of the filter transformation is negligible.

2.5.1 Filter transformation

The set of filters at layer l is stored in an array $F[\cdot]$ of shape $K^l \times K^{l-1} \times S^{l-1} \times n \times n$, where K^l is the number of channels at layer l, S^{l-1} denotes the number of transformations in G that leave the origin invariant (e.g. 1, 4 or 8 for $\mathbb{Z}^2$, $p4$ or $p4m$ filters, respectively), and n is the spatial (or translational) extent of the filter. Note that typically, $S^1 = 1$ for 2D images, while $S^l = 4$ or $S^l = 8$ for $l > 1$.

The filter transformation L_s amounts to a permutation of the entries of each of the $K^l \times K^{l-1}$ scalar-valued filter channels in F. Since we are applying S^l transformations to each filter, the output of this operation is an array of shape $K^l \times S^l \times K^{l-1} \times S^{l-1} \times n \times n$, which we call F^+.

The permutation can be implemented efficiently by a GPU kernel that does a lookup into F for each output cell of F^+, using a precomputed index associated with the output cell. To precompute the indices, we define an invertible map $g(s, u, v)$ that takes an input index (valid for an array of shape $S^{l-1} \times n \times n$) and produces the associated group element g as a matrix (section 2.2.1 and 2.2.2). For each input index (s, u, v) and each transformation s', we compute $\bar{s}, \bar{u}, \bar{v} = g^{-1}(g(s', 0, 0)^{-1} g(s, u, v))$. This index is used to set $F^+[i, s', j, s, u, v] = F[i, j, \bar{s}, \bar{u}, \bar{v}]$ for all i, j.

The G-convolution for a new group can be added by simply implementing a map $g(\cdot)$ from indices to matrices

2.5.2 Planar convolution

The second part of the G-convolution algorithm is a planar convolution using the expanded filter bank F^+. If $S^{l-1} > 1$, the sum over X in eq. 2.17 involves a sum over the stabilizer. This sum can be folded into the sum over feature channels performed by the planar convolution routine by reshaping F^+ from $K^l \times S^l \times K^{l-1} \times S^{l-1} \times n \times n$ to $S^l K^l \times S^{l-1} K^{l-1} \times n \times n$. The resulting array can be interpreted as a conventional filter bank with $S^{l-1}K^{l-1}$ planar input channels and $S^l K^l$ planar output channels, which can be correlated with the feature maps f (similarly reshaped).

2.6 Experiments

2.6.1 Rotated MNIST

The rotated MNIST dataset (Larochelle et al., 2007) contains 62000 randomly rotated handwritten digits. The dataset is split into a training, validation and test sets of size 10000, 2000 and 50000, respectively.

We performed model selection using the validation set, yielding a CNN architecture (Z2CNN) with 7 layers of 3×3 convolutions (4×4 in the final layer), 20 channels in each layer, relu activation functions, batch normalization, dropout, and max-pooling after layer 2. For optimization, we used the Adam algorithm (Kingma and Ba, 2015). This baseline architecture outperforms the models tested by Larochelle et al. (2007) (when trained on 12k and evaluated on 50k), but does not match the previous state of the art, which uses prior knowledge about rotations (Schmidt and Roth, 2012) (see table 2.1).

Next, we replaced each convolution by a $p4$-convolution (eq. 2.9 and 2.10), divided the number of filters by $\sqrt{4} = 2$ (so as to keep the number of parameters approximately fixed), and added max-pooling over rotations after the last convolution layer. This architecture (P4CNN) was found to perform better without dropout, so we removed it. The P4CNN almost halves the error rate of the previous state of the art (2.28% vs 3.98% error).

We then tested the hypothesis that premature invariance is undesirable in a deep architecture. We took the Z2CNN, replaced each convolution layer by a $p4$-convolution (eq. 2.9) followed by a coset max-pooling over rotations. The resulting feature maps consist of rotation-invariant features, and have the same transformation law as the input image. We refer to this network as P4CNNRotationPooling. It outperforms the baseline and the previous state of

the art, but performs significantly worse than the P4CNN which does not pool over rotations in intermediate layers.

Network	Test Error (%)
Larochelle et al. (2007)	10.38 ± 0.27
Sohn and Lee (2012)	4.2
Schmidt and Roth (2012)	3.98
Z2CNN	5.03 ± 0.0020
P4CNNRotationPooling	3.21 ± 0.0012
P4CNN	**2.28 ± 0.0004**

Table 2.1: *Error rates on rotated MNIST (with standard deviation under variation of the random seed).*

2.6.2 CIFAR-10

The CIFAR-10 dataset consists of $60k$ images of size 32×32, divided into 10 classes. The dataset is split into $40k$ training, $10k$ validation and $10k$ testing splits.

We compared the $p4$-, $p4m$- and standard planar $\mathbb{Z}^2$ convolutions on two kinds of baseline architectures. Our first baseline is the All-CNN-C architecture by Springenberg et al. (2015), which consists of a sequence of 9 strided and non-strided convolution layers, interspersed with rectified linear activation units, and nothing else. Our second baseline is a residual network (He et al., 2016a), which consists of an initial convolution layer, followed by three stages of $2n$ convolution layers using k_i filters at stage i, followed by a final classification layer ($6n + 2$ layers in total). The first convolution in each stage $i > 1$ uses a stride of 2, so the feature map sizes are 32, 16, and 8 for the three stages. We use $n = 7$, $k_i = 32, 64, 128$ yielding a wide 44-layer network called ResNet44.

To evaluate G-CNNs, we replaced all convolution layers of the baseline architectures by $p4$ or $p4m$ convolutions. For a constant number of filters, this increases the size of the feature maps 4 or 8-fold, which in turn increases the number of parameters required per filter in the next layer. Hence, we halve the number of filters in each $p4$-conv layer, and divide it by roughly $\sqrt{8} \approx 3$ in each $p4m$-conv layer. This way, the number of parameters is left approximately invariant, while the size of the internal representation is increased. Specifically,

we used $k_i = 11, 23, 45$ for $p4m$-ResNet44.

To evaluate the impact of data augmentation, we compare the networks on CIFAR10 and augmented CIFAR10+. The latter denotes moderate data augmentation with horizontal flips and small translations, following Goodfellow et al. (2013) and many others.

The training procedure for training the All-CNN was reproduced as closely as possible from Springenberg et al. (2015). For the ResNets, we used stochastic gradient descent with initial learning rate of 0.05 and momentum 0.9. The learning rate was divided by 10 at epoch 50, 100 and 150, and training was continued for 300 epochs.

Network	G	CIFAR10	CIFAR10+	Param.
All-CNN	$\mathbb{Z}^2$	9.44	8.86	1.37M
	$p4$	8.84	7.67	1.37M
	$p4m$	7.59	7.04	1.22M
ResNet44	$\mathbb{Z}^2$	9.45	5.61	2.64M
	$p4m$	**6.46**	**4.94**	2.62M

Table 2.2: *Comparison of conventional (i.e. $\mathbb{Z}^2$), $p4$ and $p4m$ CNNs on CIFAR10 and augmented CIFAR10+. Test set error rates and number of parameters are reported.*

To the best of our knowledge, the $p4m$-CNN outperforms all published results on plain CIFAR10 (Wan et al., 2013; Goodfellow et al., 2013; Lin et al., 2014; Lee et al., 2015b; Srivastava et al., 2015b; Clevert et al., 2015; Lee et al., 2015a). However, due to radical differences in model sizes and architectures, it is difficult to infer much about the intrinsic merit of the various techniques. It is quite possible that the cited methods would yield better results when deployed in larger networks or in combination with other techniques. Extreme data augmentation and model ensembles can also further improve the numbers (Graham, 2014).

Inspired by the wide ResNets of Zagoruyko and Komodakis (2016), we trained another ResNet with 26 layers and $k_i = (71, 142, 248)$ (for planar convolutions) or $k_i = (50, 100, 150)$ (for $p4m$ convolutions). When trained with moderate data augmentation, this network achieves an error rate of 5.27% using planar convolutions, and **4.19**% with $p4m$ convolutions. This result is comparable to the 4.17% error reported by Zagoruyko and Komodakis (2016), but using fewer parameters (7.2M vs 36.5M).

2.7 Discussion & Future work

Our results show that $p4$ and $p4m$ convolution layers can be used as a drop-in replacement of standard convolutions that consistently improves the results.

G-CNNs benefit from data augmentation in the same way as convolutional networks, as long as the augmentation comes from a group larger than G. Augmenting with flips and small translations consistently improves the results for the $p4$ and $p4m$-CNN.

The CIFAR dataset is not actually symmetric, since objects typically appear upright. Nevertheless, we see substantial increases in accuracy on this dataset, indicating that there need not be a full symmetry for G-convolutions to be beneficial.

In future work, we want to implement G-CNNs that work on hexagonal lattices which have an increased number of symmetries relative to square grids, as well as G-CNNs for 3D space groups. All of the theory presented in this paper is directly applicable to these groups, and the G-convolution can be implemented in such a way that new groups can be added by simply specifying the group operation and a bijective map between the group and the set of indices.

One limitation of the method as presented here is that it only works for discrete groups. Convolution on continuous (locally compact) groups is mathematically well-defined, but may be hard to approximate in an equivariant manner. A further challenge, already identified by Gens and Domingos (2014), is that a full enumeration of transformations in a group may not be feasible if the group is large.

2.8 Conclusion

We have introduced G-CNNs, a generalization of convolutional networks that substantially increases the expressive capacity of a network without increasing the number of parameters. By exploiting symmetries, G-CNNs achieve state of the art results on rotated MNIST and CIFAR10. We have developed the general theory of G-CNNs for discrete groups, showing that all layer types are equivariant to the action of the chosen group G. Our experimental results show that G-convolutions can be used as a drop-in replacement for spatial convolutions in modern network architectures, improving their performance without further tuning.

3

Steerable G-CNNs

Based on: (Cohen and Welling, 2017).

It has long been recognized that the invariance and equivariance properties of a representation are critically important for success in many vision tasks. In this paper we present Steerable Convolutional Neural Networks, an efficient and flexible class of equivariant convolutional networks. We show that steerable CNNs achieve state of the art results on the CIFAR image classification benchmark. The mathematical theory of steerable representations reveals a *type system* in which any steerable representation is a composition of elementary *feature types*, each one associated with a particular kind of symmetry. We show how the parameter cost of a steerable filter bank depends on the types of the input and output features, and show how to use this knowledge to construct CNNs that utilize parameters effectively.

3.1 Introduction

Much of the recent progress in computer vision can be attributed to the availability of large labelled datasets and deep neural networks capable of absorbing large amounts of information. While many practical problems can now be solved, the requirement for big (labelled) data is a fundamentally unsatisfactory state of affairs. Human beings are able to learn new concepts with very few labels, and reproducing this ability is an important challenge for artificial intelligence research. From an applied perspective, improving the statistical efficiency of deep learning is vital because in many domains (e.g. medical im-

age analysis), acquiring large amounts of labelled data is costly.

To improve the statistical efficiency of machine learning methods, many have sought to learn invariant representations. In deep learning, however, intermediate layers should not be invariant, because the relative pose of local features must be preserved for further layers (Cohen and Welling, 2016; Hinton et al., 2011). Thus, one is led to the idea of *equivariance*: a network is equivariant if the representations it produces transform in a predictable way under transformations of the input. In other words, equivariant networks produce representations that are *steerable*. Steerability makes it possible to apply filters not just in every *position* (as in a standard convolution layer), but in every *pose*, thus allowing for increased parameter sharing.

Previous work has shown that equivariant CNNs yield state of the art results on classification tasks (Cohen and Welling, 2016; Dieleman et al., 2016), even though they only enforce equivariance to small groups of transformations like rotations by multiples of 90 degrees. Learning representations that are equivariant to larger groups is likely to result in further gains, but the computational cost of current methods scales linearly with the size of the group, making this impractical. In this paper we present the general theory of steerable CNNs, which covers previous approaches but also shows how the computational cost can be decoupled from the size of the symmetry group, thus paving the way for future scaling.

To better understand the structure of steerable representations, we analyze them mathematically. We show that any steerable representation is a composition of low-dimensional elementary *feature types*. Each elementary feature can be steered independently of the others, and captures a distinct characteristic of the input that has an invariant or "objective" meaning. This doctrine of "observer-independent quantities" was put forward by Weyl, 1939, ch. 1.4 and is used throughout physics. It has been applied to vision and representation learning by (Kanatani, 1990; Cohen, 2013).

The mentioned type system puts constraints on the network weights and architecture. Specifically, since an equivariant filter bank is required to map given input feature types to given output feature types, the number of parameters required by such a filter bank is reduced. Furthermore, by the same logic that tells us not to add meters to seconds, steerability considerations prevent us from adding features of different types (e.g. for residual learning (He et al., 2016a)).

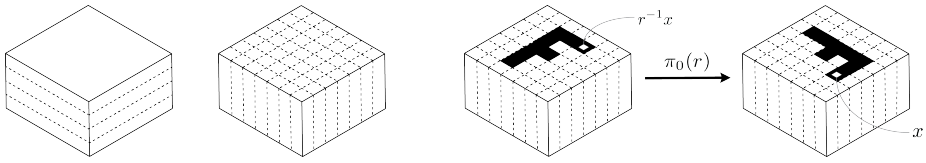
The rest of this chapter is organized as follows. The theory of steerable CNNs is introduced in Sec. 3.2. Related work is discussed in Sec. 3.3, followed by classification experiments (Sec. 3.4) and a discussion and conclusion in 3.5.

3.2 Steerable CNNs

3.2.1 Feature maps and fibers

Consider a 2D signal $f : \mathbb{Z}^2 \to \mathbb{R}^K$ with K channels. The signal may be an input to the network or a feature representation computed by a CNN. Since signals can be added and multiplied by scalars, the set of signals of this signature forms a linear space $\mathcal{F}$. Each layer of the network has its own feature space $\mathcal{F}_l$, but we will often suppress the layer index to reduce clutter.

It is customary in deep learning to describe $f \in \mathcal{F}$ as a *stack* of feature maps f_k (for $k = 1, \dots, K$). In this paper we also consider another decomposition of $\mathcal{F}$ into *fibers*. The fiber F_x at position x in the "base space" $\mathbb{Z}^2$ is the K-dimensional vector space spanned by all channels at position x. Thus, $f \in \mathcal{F}$ is comprised of *feature vectors* $f(x)$ that live in the fibers F_x (see Figure 3.1(a)).



(a) *The feature space $\mathcal{F}$ is decomposed into a* stack of feature maps *(left) and a* bundle of fibers *(right).*

(b) *An image $f \in \mathcal{F}_0$ is rotated by r using $\pi_0(r)$.*

Figure 3.1: *Feature maps, fibers, and the transformation law π_0 of $\mathcal{F}_0$.*

Given some group of transformations G that acts on points in $\mathbb{Z}^2$, we can transform signals $f \in \mathcal{F}_0$:

$$[\pi_0(g)f](x) = f(g^{-1}x) \tag{3.1}$$

This says that the pixel at $g^{-1}x$ gets moved to x by the transformation $g \in G$.

One can verify that $\pi_0(g)$ is linear (i.e. $\pi_0(g)(\alpha f + \beta f') = \alpha\pi_0(g)f + \beta\pi_0(g)f'$), and satisfies $\pi_0(gh) = \pi_0(g)\pi_0(h)$. Hence, it is a group representation (Serre, 1977).

3.2.2 Steerable representations

Let $(\mathcal{F}, \pi)$ be a feature space with a group representation and $\Phi : \mathcal{F} \to \mathcal{F}'$ a convolutional network. The feature space $\mathcal{F}'$ is said to be (linearly) *steerable*

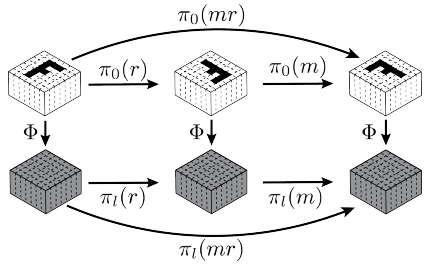


Figure 3.2: *Diagram showing the structural consistency that follows from equivariance of the network Φ and the group representation structure of π_0. The result of following any path in this diagram depends only on the beginning and endpoint but is independent of the path itself, c.f. eq. 3.2*

with respect to G, if for all transformations $g \in G$, the features Φf and $\Phi\pi(g)f$ are related by a linear transformation $\pi'(g)$ that does not depend on f. So $\pi'(g)$ allows us to "steer" the features in $\mathcal{F}'$ without referring to the input in $\mathcal{F}$ from which they were computed.

Combining the definition of steerability (i.e. $\Phi\pi(g) = \pi'(g)\Phi$) with the fact that π is a group representation, we find that π' must also be a group representation:

$$\pi'(gh)\Phi f = \Phi\pi(gh)f = \Phi\pi(g)\pi(h)f = \pi'(g)\Phi\pi(h)f = \pi'(g)\pi'(h)\Phi f \qquad (3.2)$$

That is, $\pi'(gh) = \pi'(g)\pi'(h)$ (at least in the span of the image of Φ). Figure 3.2 gives an illustration.

For simplicity, we will restrict our attention to discrete groups of transformations. The theory for continuous groups is almost completely analogous. Our running example will be the group $p4m$ which consists of translations, rotations by 90 degrees around any point, and reflections (Section 2.2.2). We further restrict our attention to groups that are constructed[1] from the group of translations $\mathbb{Z}^2$ and a group H of transformations that fixes the origin $\mathbf{0} \in \mathbb{Z}^2$. This group H is called the stabilizer subgroup. For $p4m$, we have $H = D4$, the 8-element group of reflections and rotations about the origin.

Using this division, we can first construct a filter bank that generates H-steerable fibers, and then show that convolution with such a filter bank produces a feature space that is steerable with respect to the whole group G.

[1] as a semi-direct product

3.2.3 Equivariant filter banks

A filter bank can be described as an array of dimension (K', K, s, s), where K, K' denote the number of input / output channels and s is the kernel size. For our purposes it is useful to think of a filter bank as a linear map $\Psi : \mathcal{F} \to \mathbb{R}^{K'}$ that takes as input a signal $f \in \mathcal{F}$ and produces a K'-dimensional feature vector. The filter bank only looks at an $s \times s$ patch in $\mathcal{F}$, so the matrix representing Ψ has shape $K' \times K \cdot s^2$. To correlate a signal f using Ψ, one would simply apply Ψ to translated copies of f, producing the output signal one fiber at a time.

We assume (by induction) that we have a representation π that allows us to steer $\mathcal{F}$. In order to make the output of the convolution steerable, we need the filter bank $\Psi : \mathcal{F} \to \mathbb{R}^{K'}$ to be *H-equivariant*:

$$\rho(h)\,\Psi = \Psi\,\pi(h), \quad \forall h \in H \tag{3.3}$$

for some representation ρ of H that acts on the output fibers (see Figure 3.3). Note that we only require equivariance with respect to H (which excludes translations) and not G, because translations can move patterns into and out of the receptive field of a fiber, making full translation equivariance impossible.

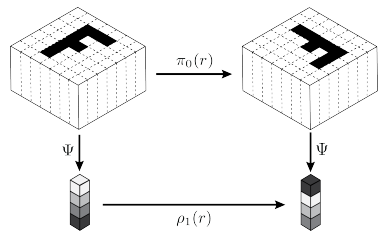


Figure 3.3: *A filter bank Ψ that is H-equivariant. In this example, ρ_1 represents the 90-degree rotation r by a permutation matrix that cyclicly shifts the 4 channels.*

The space of maps satisfying the equivariance constraint is denoted $\mathrm{Hom}_H(\pi, \rho)$, because an equivariant map Ψ is a "homomorphism of group representations", meaning it respects the structure of the representations. Equivariant maps are also sometimes called *intertwiners* (Serre, 1977).

Since the equivariance constraint (eq. 3.3) is linear in Ψ, the space $\mathrm{Hom}_H(\pi, \rho)$ of admissible filter banks is a vector space: any linear combination of maps $\Psi, \Psi' \in \mathrm{Hom}_H(\pi, \rho)$ is again an intertwiner. Hence, given π and ρ, we can compute a basis for $\mathrm{Hom}_H(\pi, \rho)$ by solving a linear system.

Computation of the intertwiner basis is done offline, before training. Once we have such a basis $\psi_1, \ldots, \psi_n$ for $\mathrm{Hom}_H(\pi, \rho)$, we can express any equivariant filter bank Ψ as a linear combination $\Psi = \sum_i \alpha_i \psi_i$ using parameters α_i. As shown in Section 3.2.8, this can be done efficiently even in high dimensions.

3.2.4 Induction

We have shown how to parameterize filter banks that intertwine π and ρ, making the output fibers H-steerable by ρ if the input space $\mathcal{F}$ is H-steerable by π. In this section we show how H-steerability of fibers F'_x leads to G-steerability of the whole feature space $\mathcal{F}'$. This happens through a natural and important construction known as the *induced representation* (Mackey, 1952; Mackey, 1953; Mackey, 1968; Serre, 1977; Taylor, 1986; Folland, 1995; Kaniuth and Taylor, 2013).

As stated before, the correlation $\Psi \star f$ could be computed by translating f before applying Ψ:

$$[\Psi \star f](x) = \Psi\Big[\pi(x)^{-1} f\Big]. \tag{3.4}$$

Where $x \in \mathbb{Z}^2$ is interpreted as a translation when given as input to π.

We can now calculate the transformation law of the output space. To do so, we apply a translation t and transformation $r \in H$ to $f \in \mathcal{F}$, yielding $\pi(tr)f$, and then perform the correlation with Ψ. With a some algebra (Appendix A), we find:

$$[\Psi \star [\pi(tr)f]](x) = \rho(r)\Big[[\Psi \star f]((tr)^{-1}x)\Big] \tag{3.5}$$

Now if we define π' as

$$\big[\pi'(tr)f\big](x) = \rho(r)\Big[f((tr)^{-1}x)\Big] \tag{3.6}$$

then $\Psi \star \pi(g)f = \pi'(g)\Psi \star f$ (see Fig. 3.4). π' is known as the representation of G induced by the representation ρ of H, and is denoted $\pi' = \mathrm{ind}_H^G \rho$.

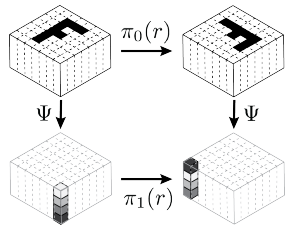


Figure 3.4: *The representation π_1 induced from the permutation representation ρ_1 shown in fig. 3.3. A single fiber is highlighted. It is transported to a new location, and acted on by ρ_1.*

When parsing eq. 3.6, it is important to keep in mind that (as indicated by the square brackets) π' acts on the whole feature space $\mathcal{F}'$ while ρ acts on individual fibers.

If we compare the induced representation (eq. 3.6) to the representation π_0 defined in eq. 3.1, we see that the difference lies only in the presence of a factor $\rho(r)$ applied to the fibers. This factor describes how the feature channels are mixed by the transformation. The color channels in the input space do not get mixed by geometrical transformations, so we say that π_0 is induced from the trivial representation $\rho_0(h) = I$.

Now that we have a G-steerable feature space $\mathcal{F}'$, we can iterate the procedure by computing a basis for the space of intertwiners between π' (restricted to H) and some ρ' of our choosing.

3.2.5 Feature types and character theory

By now, the reader may be wondering how to choose ρ, or indeed what the space of representations that we can choose from looks like in the first place. We will answer these questions in this section by showing that each representation has a *type* (encoded as a short list of integers) that corresponds to a certain symmetry or invariance of the feature. We further show how the number of parameters of an equivariant filter bank depends on the types of the representations π and ρ that it intertwines. Our discussion will make use of a number of important elementary results from group representation theory which are stated but not proved. The reader wishing to go deeper may consult chapters 1 and 2 of the excellent book by Serre (1977).

Recall that a group representation is a set of invertible linear maps $\rho(g) : \mathbb{R}^K \to \mathbb{R}^K$ satisfying $\rho(gh) = \rho(g)\rho(h)$ for all elements $g, h \in H$. It can be shown that any representation is a direct sum (i.e. `block_diag` plus change of basis) of a number of "elementary" representations associated with G. These building blocks are called irreducible representations (or irreps), because they can themselves not be block-diagonalized. In other words, if φ_i are the irreducible representations of H, any representation ρ of H can be written in block-diagonal form:

$$\rho(g) = A \begin{bmatrix} \varphi_{i_1}(g) & & \\ & \ddots & \\ & & \varphi_{i_n} \end{bmatrix} A^{-1} \tag{3.7}$$

for some basis matrix A, and some i_k that index the irreps (each irrep may

Irrep	Basis in $\mathcal{F}_0$	e	r	r^2	r^3	m	mr	mr^2	mr^3
A1		$[1]$	$[1]$	$[1]$	$[1]$	$[1]$	$[1]$	$[1]$	$[1]$
A2		$[1]$	$[1]$	$[1]$	$[1]$	$[-1]$	$[-1]$	$[-1]$	$[-1]$
B1		$[1]$	$[-1]$	$[1]$	$[-1]$	$[1]$	$[-1]$	$[1]$	$[-1]$
B2		$[1]$	$[-1]$	$[1]$	$[-1]$	$[-1]$	$[1]$	$[-1]$	$[1]$
E		$\begin{bmatrix}1&0\\0&1\end{bmatrix}$	$\begin{bmatrix}0&-1\\1&0\end{bmatrix}$	$\begin{bmatrix}-1&0\\0&-1\end{bmatrix}$	$\begin{bmatrix}0&1\\-1&0\end{bmatrix}$	$\begin{bmatrix}-1&0\\0&1\end{bmatrix}$	$\begin{bmatrix}0&1\\1&0\end{bmatrix}$	$\begin{bmatrix}1&0\\0&-1\end{bmatrix}$	$\begin{bmatrix}0&-1\\-1&0\end{bmatrix}$

Table 3.1: *The irreducible representations of the roto-reflection group D4. This group is generated by 90-degree rotations r and mirror reflections m, and has 5 irreps labelled A1, A2, B1, B2, E. Left: decomposition of π_0 (eq. 3.1) in the space $\mathcal{F}_0$ of 3×3 filters with one channel. This representation turns out to have type $(3,0,1,1,2)$, meaning there are three copies of A1, one copy of B1, one copy of B2, and two copies of the 2D irrep E (A2 does not appear). Any 3×3 filter can be written as a linear combination of the 9 basis filters shown. Right: the representation matrices of each irrep, for each element of the group D4. The reader may verify that these are valid representations, and that the characters (traces) are orthogonal.*

occur 0 or more times).

Each irreducible representation corresponds to a type of symmetry, as shown in table 3.1. For example, as can be seen in this table, the representations $B1$ and $B2$ represent the 90-degree rotation r as the matrix $\begin{bmatrix}-1\end{bmatrix}$, so the basis filters for these representations change sign when rotated by r. It should be noted that in the higher layers $l > 0$, elementary basis filters can look different because they depend on the representation π_l that is being decomposed.

The fact that all representations can be decomposed into a direct sum of irreducibles implies that each representation has a basis-independent *type*: which irreducible representations appear in it, and with what multiplicity. For example, the input representation π_0 (table 3.1) has type $(3,0,1,1,2)$. This means that, for instance, $\pi_0(r)$ is block-diagonalized as:

$$A^{-1}\pi_0(r)A = \texttt{block\_diag}\left(\begin{bmatrix}1\end{bmatrix},\begin{bmatrix}1\end{bmatrix},\begin{bmatrix}1\end{bmatrix},\begin{bmatrix}-1\end{bmatrix},\begin{bmatrix}-1\end{bmatrix},\begin{bmatrix}0&-1\\1&0\end{bmatrix},\begin{bmatrix}0&1\\-1&0\end{bmatrix}\right). \tag{3.8}$$

Where the block matrix contains $(3,0,1,1,2)$ copies of the irreps A1, A2, B1, B2,

and E, evaluated at r (see column r in table 3.1). The change of basis matrix A is constructed from the basis filters shown in table 3.1 (and the same A block-diagonalizes $\pi_0(g)$ for all g).

So the most general way in which we can choose a representation ρ is to choose multiplicities $m_i \geq 0$ and a basis matrix A. In Section 3.2.7 we will find that there is an important restriction on this freedom, which alleviates the need to choose a basis. The choice of multiplicities is then the only hyperparameter, analogous to the choice of the number of channels in an ordinary CNN. Indeed, the multiplicities determine the number of channels: $K = \sum_i m_i \dim \varphi_i$.

3.2.6 Determining the type of the induced representation

By choosing the type of ρ, we also determine the type of $\pi = \text{ind}_H^G \rho$ (restricted to H), but what is it? Explicit formulas exist (Reeder, 2014; Serre, 1977) but are rather complicated, so we will present a simple computational procedure that can be used to determine the type of any representation. This procedure relies on the *character* $\chi_\rho(g) = \text{Tr}(\rho(g))$ of the representation to be decomposed. The most important fact about characters is that the characters of irreps φ_i, φ_j are orthogonal:

$$\langle \chi_{\varphi_i}, \chi_{\varphi_j} \rangle \equiv \frac{1}{|H|} \sum_{h \in H} \chi_{\varphi_i}(h) \chi_{\varphi_j}(h) = \delta_{ij}. \tag{3.9}$$

Furthermore, since the trace of a direct sum equals the sum of the traces (i.e. $\chi_{\rho \oplus \rho'} = \chi_\rho + \chi_{\rho'}$), and every representation ρ is a direct sum of irreps, it follows that we can obtain the multiplicity of irrep φ_i in ρ by computing the inner product with the i-th character:

$$\langle \chi_\rho, \chi_{\varphi_i} \rangle = \langle \chi_{\oplus_j m_j \varphi_j}, \chi_{\varphi_i} \rangle = \left\langle \sum_j m_j \chi_{\varphi_j}, \chi_{\varphi_i} \right\rangle = \sum_j m_j \langle \chi_{\varphi_j}, \chi_{\varphi_i} \rangle = m_i \tag{3.10}$$

So a simple dot product of characters is all we need to determine the type of a representation. As we will see next, the type of the input and output representation of a layer determines the parameter cost of that layer.

The parameter cost of equivariant convolution layers

Steerable CNNs use parameters much more efficiently than ordinary CNNs. In this section we show how the number of parameters required by an equivariant

layer is determined by the feature types of the input and output space, and how the efficiency of a choice of feature types may be evaluated.

In section 3.2.3, we found that a filter bank Ψ is equivariant if and only if it lies in the vector space called $\mathrm{Hom}_H(\pi, \rho)$. It follows that the number of parameters for such a filter bank is equal to the dimensionality of this space, $n = \dim \mathrm{Hom}_H(\pi, \rho)$. This number is known as the *intertwining number* of π and ρ and plays an important role in the theory of group representations.

As with multiplicities, the intertwining number is easily computed using characters. It can be shown (Reeder, 2014) that the intertwining number equals:

$$\dim \mathrm{Hom}_H(\pi, \rho) = \langle \chi_\pi, \chi_\rho \rangle. \tag{3.11}$$

By linearity and the orthogonality of characters, we find that $\dim \mathrm{Hom}_H(\pi, \rho) = \sum_i m_i m'_i$, for representations π, ρ of type $(m_1, \dots, m_J)$ and $(m'_1, \dots, m'_J)$, respectively. Thus, as far as the number of parameters of a steerable convolution layer is concerned, the only choice we have to make for ρ is its type – a short list of integers m_i.

The efficiency of a choice of type can be assessed using a quantity we call the *parameter utilization*:

$$\mu = \frac{\dim \pi \cdot \dim \rho}{\dim \mathrm{Hom}_H(\pi, \rho)}. \tag{3.12}$$

The numerator equals $s^2 K \cdot K'$: the number of parameters for a non-equivariant filter bank. The denominator equals the parameter cost of an equivariant filter bank with the same filter size and number of input/output channels. Typical values of μ in effective architectures are around $|H|$, e.g. $\mu = 8$ for $H = D4$. Such a layer utilizes its parameters 8 times more intensively than an ordinary convolution layer.

3.2.7 Equivariant nonlinearities & capsules

In the previous section we showed that only the basis-independent types of π and ρ play a role in determining the parameter cost of an equivariant filter bank. An equivalent representation $\rho'(g) = A\rho(g)A^{-1}$ will have the same type, and hence the same parameter cost as ρ. However, when it comes to nonlinearities, different bases behave differently.

Just like a convolution layer (eq. 3.3), a layer of nonlinearities must commute with the group action. An elementwise nonlinearity $\nu : \mathbb{R} \to \mathbb{R}$ (or more generally, a fiber-wise nonlinearity $\nu : \mathbb{R}^K \to \mathbb{R}^{K'}$) is admissible for an input

representation ρ if there exists an output representation ρ' such that ν applied after ρ equals ρ' applied after ν.

Since commutation with nonlinearities depends on the basis, we need a more granular notion than the feature type. We define a ρ-capsule as a (typically low-dimensional) feature vector that transforms according to a representation ρ (we may also refer to ρ as the capsule). Thus, while a capsule has a type, not all representations of that type are equivalent as capsules. Given a catalogue of capsules ρ^i (for $i = 1,\dots,C$) with multiplicities m_i, we can construct a fiber as a *stack of capsules* that is steerable by a block-diagonal representation ρ with m_i copies of ρ^i on the diagonal.

Like the capsules of Hinton et al., 2011, our capsules encode the pose of a pattern in the input, and consist of a number of units (dimensions) that do not get mixed with the units of other capsules by symmetries. In this sense, a stack of capsules is *disentangled* (Cohen and Welling, 2014).

We have found a few simple types of capsules and corresponding admissible nonlinearities. It is easy to see that any nonlinearity is admissible for ρ when the latter is realized by permutation matrices: permuting a list of coordinates and then applying a nonlinearity is the same as applying the nonlinearity and then permuting. If ρ is realized by a signed permutation matrix, then $\mathtt{CReLU}(\alpha) = (\mathtt{ReLU}(\alpha), \mathtt{ReLU}(-\alpha))$ introduced by Shang et al. (2016), or any concatenated nonlinearity $\nu'(\alpha) = (\nu(\alpha), \nu(-\alpha))$, will be admissible. Any scale-free concatenated nonlinearity such as CReLU is admissible for a representation realized by monomial matrices (having the same nonzero pattern as a permutation matrix). Finally, we can always make a representation of a finite group *orthogonal* by a suitable choice of basis, which means that we can use any nonlinearity that acts only on the length of the vector.

For many groups, the irreps can be realized using signed permutation matrices, so we can use irreducible φ_i-capsules with concatenated nonlinearities such as CReLU. Another class of capsules, which we call quotient capsules, are naturally realized by permutation matrices, and are thus compatible with any nonlinearity. These are described in Appendix C.

3.2.8 Computational efficiency

Modern convolutional networks often use on the order of hundreds of channels K per layer Zagoruyko and Komodakis, 2016. When using 3×3 filters, a filter bank can have on the order of $9K^2 \approx 10^6$ dimensions. The number of parameters for an equivariant filter bank is about $\mu \approx 10$ times smaller, but a basis for the space of equivariant filter banks would still be about $10^6 \times 10^5$,

which is too large to be practical.

Fortunately, the block-diagonal structure of π and ρ induces a block structure in Ψ. Suppose $\pi = \texttt{block\_diag}(\pi^1, \dots, \pi^P)$ and $\rho = \texttt{block\_diag}(\rho^1, \dots, \rho^Q)$. Then an intertwiner is a matrix of shape $K' \times Ks^2$, where $K' = \sum_i \dim \rho^i$ and $Ks^2 = \sum_i \dim \pi^i$. This matrix has the following block structure:

$$\Psi = \begin{bmatrix} h_{11} \in \mathrm{Hom}_H(\rho^1, \pi^1) & \cdots & h_{1P} \in \mathrm{Hom}_H(\rho^1, \pi^P) \\ \vdots & \ddots & \vdots \\ h_{R1} \in \mathrm{Hom}_H(\rho^R, \pi^1) & \cdots & h_{RP} \in \mathrm{Hom}_H(\rho^R, \pi^P) \end{bmatrix} \tag{3.13}$$

Each block h_{ij} corresponds to an input-output pair of capsules, and can be parameterized by a linear combination of basis matrices $\psi_k^{ij} \in \mathrm{Hom}_H(\rho^i, \pi^j)$.

In practice, we typically use many copies of the same capsule (say n_i copies of ρ^i and m_j copies of π^j). Therefore, many of the blocks h_{ij} can be constructed using the same intertwiner basis. If we order equivalent capsules to be adjacent, the intertwiner consists of "blocks of blocks". Each superblock H_{ij} has shape $n_i \dim \rho^i \times m_j \dim \pi^j$, and consists of subblocks of shape $\dim \rho^i \times \dim \pi^j$.

The computation graph for an equivariant convolution layer is constructed as follows. Given a catalogue of capsules ρ^i and corresponding post-activation capsules $\mathrm{Act}_\nu \rho^i$, we compute the induced representations $\pi^i = \mathrm{ind}_H^G \mathrm{Act}_\nu \rho^i$ and the bases for $\mathrm{Hom}_H(\rho^i, \pi^j)$ in an offline step. The bases are stored as matrices ψ^{ij} of shape $\dim \rho^i \cdot \dim \pi^j \times \dim \mathrm{Hom}_H(\rho^i, \pi^j)$. Then, given a list of input / output multiplicities n_i, m_j for the capsules, a parameter matrix Θ^{ij} of shape $\dim \mathrm{Hom}_H(\rho^i, \pi^j) \times n_i m_j$ is instantiated. The superblocks H_{ij} are obtained by a matrix multiplication $\psi^{ij}\Theta^{ij}$ plus reshaping to shape $\dim \rho^i \cdot \dim \pi^j \times n_i m_j$. Once all superblocks are filled in, the matrix Ψ is reshaped from $K' \times Ks^2$ to $K' \times K \times s \times s$ and convolved with the input.

3.2.9 Using steerable CNNs in practice

A full understanding of the theory of steerable CNNs requires some knowledge of group representation theory, but using steerable CNN technology is not much harder than using ordinary CNNs. Instead of choosing a number of channels for a given layer, one chooses a list of multiplicities m_i for each capsule in a library of capsules provided by the developer. To preserve equivariance, the activation function applied to a capsule must be chosen from a list of admissible nonlinearities for that capsule (which sometimes includes all non-

linearities). Finally, one must respect the type system and only add identical capsules (e.g. in ResNets). These constraints can all be checked automatically.

3.3 Related Work

Steerable filters were first studied for applications in signal processing and low-level vision (Freeman and Adelson, 1991; Greenspan et al., 1994; Simoncelli and Freeman, 1995). More or less explicit connections between steerability and group representation theory have been observed by Lenz, 1989; Koenderink and Van Doorn, 1990; Teo, 1998; Krajsek and Mester, 2007. As we have tried to demonstrate in this paper, representation theory is indeed the natural mathematical framework in which to study steerability.

In machine learning, equivariant kernels were studied by Reisert, 2008; Skibbe, 2013. In the context of neural networks, various authors have studied equivariant representations. Capsules were introduced in Hinton et al., 2011, and significantly improved by Tieleman, 2014. A theoretical account of equivariant representation learning in the brain is given by Anselmi et al., 2014. Group equivariant scattering networks were defined and studied by Mallat, 2012 for compact groups, and by Sifre and Mallat, 2013; Oyallon and Mallat, 2015 for the roto-translation group. Jacobsen et al., 2016 describe a network that uses a fixed set of (possibly steerable) basis filters with learned weights. Lenc and Vedaldi, 2015 showed empirically that convolutional networks tend to learn equivariant representations, which suggests that equivariance could be a good inductive bias.

Invariant and equivariant CNNs have been studied by Gens and Domingos, 2014; Kanazawa et al., 2014; Dieleman et al., 2015; Dieleman et al., 2016; Cohen and Welling, 2016; Marcos et al., 2016. All of these models, as well as scattering networks, implicitly use the *regular representation*: feature maps are (often implicitly) conceived of as functions on G, and the action of G on the space of functions on G is known as the regular representation (Serre, 1977, Appendix B). Our work is the first to consider other kinds of equivariance in the context of CNNs.

The idea of adding a type system to neural networks has been explored by Olah, 2015; Balduzzi and Ghifary, 2016. We have shown that a type system emerges naturally from the decomposition of a linear representation of a mathematical structure (a group, in our case) associated with the representation learned by a neural network.

3.4 Experiments

We implemented steerable CNNs in Chainer (Tokui et al., 2015) and performed experiments on the CIFAR10 dataset (Krizhevsky, 2009) to determine if steerability is a useful inductive bias, and to determine the relative merits of the various types of capsules. In order to run experiments faster, and to see how steerable CNNs perform in the small-data regime, we used only 2000 training samples for our initial experiments.

As a baseline, we used the competitive wide residual networks (ResNets) architecture (He et al., 2016a; He et al., 2016b; Zagoruyko and Komodakis, 2016). We tuned the capacity of this network for the reduced dataset size and settled on a 20 layer architecture (three residual blocks per stage, with two layers each, for three stages with feature maps of size 32×32, 16×16 and 8×8, various widths). We compared the baseline architecture to various kinds of steerable CNN, obtained by replacing the convolution layers by steerable convolution layers. To make sure that differences in performance were not simply due to underfitting or overfitting, we tuned the width (number of channels, K) using a validation set. The rest of the training procedure is identical to Cohen and Welling, 2016, and is fixed for all of our experiments.

We first tested steerable CNNs that consist entirely of a single kind of capsule. We found that architectures with only one type do not perform very well (roughly 30-40% error, vs. 30% for plain ResNets trained on 2k samples from CIFAR10), except for those that use the regular representation capsule (Appendix C), which outperforms standard CNNs (26.75% error). This is not too surprising, because many capsules are quite restrictive in the spatial patterns they can express. The strong performance of regular capsules is consistent with the results of Cohen and Welling, 2016, and can be explained by the fact that the regular representation contains all other (irreducible and quotient) representations as subrepresentations, and can therefore learn arbitrary spatial patterns.

We then created networks that use a mix of the more successful kinds of capsules. After a few preliminary experiments, we settled on a residual network that uses one mix of capsules for the input and output layer of a residual block, and another for the intermediate layer. The first representation consists of quotient capsules: regular, qm, qmr2, qmr3 (see Appendix C) followed by ReLUs. The second consists of irreducible capsules: A1, A2, B1, B2, E(2x) followed by CReLUs. On CIFAR10 with 2k labels, this architecture works better than standard ResNets and regular capsules at 24.48% error.

Net	Depth	Width	#Params	#Labels	Dataset	Test error
Ladder	10	96		4k	C10ss	20.4
steer	14	(280, 112)	4.4M	4k	C10	23.66
steer	20	(160, 64)	2.2M	4k	C10	24.56
steer	14	(280, 112)	4.4M	4k	C10+	16.44
steer	20	(160, 64)	2.2M	4k	C10+	16.42
ResNet	1001	16	10.2M	50k	C10+	4.62
Wide	28	160	36.5M	50k	C10+	4.17
Dense	100	2400	27.2M	50k	C10+	3.74
steer	26	(280, 112)	9.1M	50k	C10+	3.74
steer	20	(440, 176)	16.7M	50k	C10+	3.95
steer	14	(400, 160)	9.1M	50k	C10+	**3.65**
ResNet	1001	16	10.2M	50k	C100+	22.71
Wide	28	160	36.5M	50k	C100+	20.50
Dense	100	2400	27.2M	50k	C100+	19.25
steer	20	(280, 112)	6.9M	50k	C100+	19.84
steer	14	(400, 160)	9.1M	50k	C100+	**18.82**

Table 3.2: *Comparison of results of steerable CNNs vs. previous state of the art methods. A plus (+) indicates modest data augmentation (shifts and flips). Width for steerable CNNs is reported as a pair of numbers, one for the input / output layer of a ResNet block, and one for the intermediate layer.*

When tested on CIFAR10 with 4k labels (table 3.2), the method comes close to the state of the art in *semi-supervised* methods, that use additional unlabelled data (Rasmus et al., 2015), and better than transfer learning approaches such as DCGAN which achieves 26.2% error (Radford et al., 2015). When tested on the full CIFAR10 and CIFAR100 dataset, the steerable CNN substantially outperforms the ResNet (He et al., 2016a) baseline and achieves state of the art results (improving over wide and dense nets (Zagoruyko and Komodakis, 2016; Huang et al., 2016)).

3.5 Conclusion & Future Work

We have presented a theoretical framework for understanding steerable representations in convolutional networks, and have shown that steerability is a useful inductive bias that can improve model accuracy, particularly when lit-

tle data is available. Our experiments show that a simple steerable architecture achieves state of the art results on CIFAR10 and CIFAR100, outperforming recent architectures such as wide and dense residual networks.

The mathematical connection between representation learning and representation theory that we have established improves our understanding of the inner workings of (equivariant) convolutional networks, revealing the humble CNN as an elegant geometrical computation engine. We expect that this new tool (representation theory), developed over more than a century by mathematicians and physicists, will greatly benefit future investigations in this area.

For concreteness, we have used the group of flips and rotations by multiples of 90 degrees as a running example throughout this paper. This group already has some nontrivial characteristics (such as non-commutativity), but it is still small and discrete. The theory of steerable CNNs, however, readily extends to the continuous setting. Evaluating steerable CNNs for large, continuous and high-dimensional groups is an important piece of future work.

Another direction for future work is *learning* the feature types, which may be easier in the continuous setting because (for non-compact groups) the irreps live in a continuous space where optimization may be possible. Beyond classification, steerable CNNs are likely to be useful in geometrical tasks such as action recognition, pose and motion estimation, and continuous control tasks.

Acknowledgments

We kindly thank Kenta Oono, Shuang Wu, Thomas Kipf and the anonymous reviewers for their feedback and suggestions. This research was supported by Facebook, Google and NWO (grant number NAI.14.108).

Appendix A: Induction

In this section we will show that a stack of feature maps produced by convolution with an H-equivariant filter bank transforms according to the induced representation. That is, we will derive eq. 3.5, repeated here for convenience:

$$[\Psi \star [\pi_l(tr)f]](x) = \rho_{l+1}(r)\Big[[\Psi \star f]((tr)^{-1}x)\Big] \tag{3.14}$$

In the main text, we mentioned that $x \in \mathbb{Z}^2$ can be interpreted as a point or as a translation. Here we make this difference explicit, by writing $x \in \mathbb{Z}^2$ for a point and $\bar{x} \in G$ for a translation. (The operation $\bar{\cdot}$ defines a *section* of the projection map $G \to \mathbb{Z}^2$ that forgets the non-translational part of the transformation (Kaniuth and Taylor, 2013)).

With this notation, the convolution is defined as:

$$[\Psi \star f](x) = \Psi\pi(\bar{x}^{-1})f \tag{3.15}$$

Although the induced representation can be described in a more general setting, we will use an explicit matrix representation of G to make it easier to check our computations. A general element of G is written as:

$$g = tr = \begin{bmatrix} I & T \\ 0 & 1 \end{bmatrix}\begin{bmatrix} R & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \tag{3.16}$$

Where R is the matrix representation of r (e.g. a 2×2 rotation / reflection matrix), and T is a translation vector. The section we use is:

$$\overline{x} = \begin{bmatrix} I & x \\ 0 & 1 \end{bmatrix} \tag{3.17}$$

Finally, we will distinguish the action of G on itself, written gh for $g, h \in G$ (implemented as matrix-matrix multiplication) and its action on $\mathbb{Z}^2$, written $g \cdot x$ for $g \in G$ and $x \in \mathbb{Z}^2$ (implemented as matrix-vector multiplication by adding a homogeneous coordinate to x).

To keep notation uncluttered, we will write $\pi = \pi_l$ and $\rho = \rho_{l+1}$. In full detail, the derivation of the transformation law for the feature space induced

by ρ proceeds as follows:

$$\begin{aligned}
[\Psi \star [\pi(tr)f]](x) &= \Psi\pi(\bar{x}^{-1})\pi(tr)f \\
&= \Psi\pi(\bar{x}^{-1}tr)f \\
&= \Psi\pi(rr^{-1}\bar{x}^{-1}tr)f \\
&= \Psi\pi(r)\pi(r^{-1}\bar{x}^{-1}tr)f \\
&= \rho(r)\Psi\pi(r^{-1}\bar{x}^{-1}tr)f \\
&= \rho(r)\Psi\pi((r^{-1}t^{-1}\bar{x}r)^{-1})f \\
&= \rho(r)\Psi\pi\left(\overline{(tr)^{-1}\cdot x}^{-1}\right)f \\
&= \rho(r)[\Psi \star f]((tr)^{-1}\cdot x)
\end{aligned} \tag{3.18}$$

The last line is the result shown in the paper. The justification of each step is:

1. Definition of $\star$
2. π is a homomorphism / group representation
3. rr^{-1} is the identity, so can always multiply by it
4. π is a homomorphism / group representation
5. $\Psi \in \mathrm{Hom}_H(\pi, \rho)$ is equivariant to $r \in H$.
6. Invert twice.
7. $\overline{(tr)^{-1}\cdot x} = r^{-1}t^{-1}\bar{x}r$ can be checked by multiplying the matrices / vectors.
8. Definition of $\star$

The derivation above is somewhat involved and messy, so the reader may prefer to think geometrically (using the figures in the paper) instead of algebraically. This complexity is an artifact of the lack of abstraction in our presentation. The induced representation is really a very natural object to consider (abstractly, it is the "adjoint functor" to the restriction functor. A more abstract treatment of the induced representation can be found in Serre, 1977; Mackey, 1952; Reeder, 2014. A treatment that is close to our own, but more general is the "alternate description" found on page 49 of Kaniuth and Taylor, 2013.

Appendix B: Relation to Group Equivariant CNNs

In this section we show that the recently introduced Group Equivariant Convolutional Networks (G-CNNs, Cohen and Welling, 2016) are a special kind of steerable CNN. Specifically, a G-CNN is a steerable CNN with *regular* capsules.

In a G-CNN, the feature maps (except those of the input) are thought of as functions $f : G \to \mathbb{R}^K$ instead of functions on the plane $f : \mathbb{Z}^2 \to \mathbb{R}^K$, as we do here. It is shown that the feature maps transform according to

$$\pi(g)f(h) = f(g^{-1}h). \tag{3.19}$$

This defines a linear representation of G known as the *regular representation*. It is easy to see that the regular representation is naturally realized by permutation matrices. Furthermore, it is known that the regular representation of G is induced by the regular representation of H. The latter is defined in Appendix C, and is what we refer to as "regular capsules" in the paper.

Appendix C: Regular and Quotient Features

Let H be a finite group. A subgroup of H is a subset that is also itself a group (i.e. closed under composition and inverses). The (left) coset of a subgroup K in H are the sets $hK = \{hk | k \in K\}$. The cosets are disjoint and jointly cover the whole group H (i.e. they partition H). The set of all cosets of K in H is denoted H/K, and is also called the quotient of H by K.

The coset space caries a natural left action by H. Let $a, b \in H$, then $a \cdot bK = (ab)K$.

This action translates into an action on the space of functions on H/K. Let $\mathcal{Q}$ denote the space of functions $f : H/K \to \mathbb{R}$. Then we have the following representation of H:

$$\rho(a)f(bK) = f(a^{-1} \cdot bK). \tag{3.20}$$

The function f attaches a value to every coset. The H-action permutes these values, because it permutes the cosets. Hence, ρ can be realized by permutation matrices. For small groups the explicit computations can easily be done by hand, while for large groups this task can be automated.

In this way, we get one permutation representation for each subgroup K of H. In particular, for the subgroup $K = \{e\}$ (the trivial subgroup containing only the identity e), we have $H/K \cong H$. The representation in the space of functions on H is known as the "regular representation". Using such regular

representations in a steerable CNN is equivalent to using the group convolutions introduced in Cohen and Welling, 2016, so steerable CNNs are a strict generalization of G-CNNs. At the other extreme, we take $K = H$, which gives the quotient $H/K \cong \{e\}$, the trivial group, which gives the trivial representation $A1$.

For the roto-reflection group $H = D4$, we have the following subgroups and associated quotient features

Subgroup K	quotient feature name	dimensionality
$\{e\}$	regular	8
$\{e, m\}$	qm	4
$\{e, mr\}$	qmr	4
$\{e, mr^2\}$	qmr2	4
$\{e, mr^3\}$	qmr3	4
$\{e, r^2\}$	r2	4
$\{e, r, r^2, r^3\}$	r	2
e, r^2, m, mr^2	r2m	2
e, r^2, mr, mr^3	r2mr	2
H	A1	1

4

Spherical G-CNNs

Based on: (Cohen et al., 2018c).

Convolutional Neural Networks (CNNs) have become the method of choice for learning problems involving 2D planar images. However, a number of problems of recent interest have created a demand for models that can analyze spherical images. Examples include omnidirectional vision for drones, robots, and autonomous cars, molecular regression problems, and global weather and climate modelling. A naive application of convolutional networks to a planar projection of the spherical signal is destined to fail, because the space-varying distortions introduced by such a projection will make translational weight sharing ineffective.

In this paper we introduce the building blocks for constructing spherical CNNs. We propose a definition for the spherical cross-correlation that is both expressive and rotation-equivariant. The spherical correlation satisfies a generalized Fourier theorem, which allows us to compute it efficiently using a generalized (non-commutative) Fast Fourier Transform (FFT) algorithm. We demonstrate the computational efficiency, numerical accuracy, and effectiveness of spherical CNNs applied to 3D model recognition and atomization energy regression.

4.1 Introduction

Convolutional networks are able to detect local patterns regardless of their position in the image. Like patterns in a planar image, patterns on the sphere can move around, but in this case the "move" is a 3D rotation instead of a translation. In analogy to the planar CNN, we would like to build a network that can detect patterns regardless of how they are rotated over the sphere.

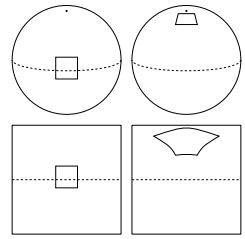


Figure 4.1: *Any planar projection of a spherical signal will result in distortions. Rotation of a spherical signal cannot be emulated by translation of its planar projection.*

As shown in Figure 4.1, there is no good way to use translational convolution or cross-correlation[1] to analyze spherical signals. The most obvious approach, then, is to change the definition of cross-correlation by replacing filter translations by rotations. Doing so, we run into a subtle but important difference between the plane and the sphere: whereas the space of moves for the plane (2D translations) is itself isomorphic to the plane, the space of moves for the sphere (3D rotations) is a different, *three-dimensional* manifold called SO(3)[2]. It follows that the result of a spherical correlation (the output feature map) is to be considered a signal on SO(3), not a signal on the sphere, S^2. For this reason, we deploy SO(3) group correlation in the higher layers of a spherical CNN (Cohen and Welling, 2016).

The implementation of a spherical CNN (S^2-CNN) involves two major challenges. Whereas a square grid of pixels has discrete translation symmetries, no perfectly symmetrical grids for the sphere exist. This means that there is no simple way to define the rotation of a spherical filter by one pixel. Instead, in order to rotate a filter we would need to perform some kind of interpolation. The other challenge is computational efficiency; SO(3) is a three-dimensional manifold, so a naive implementation of SO(3) correlation is $O(n^6)$.

We address both of these problems using techniques from non-commutative harmonic analysis (Chirikjian and Kyatkin, 2001; Folland, 1995). This field

[1] Despite the name, CNNs typically use cross-correlation instead of convolution in the forward pass. In this paper we will generally use the term cross-correlation, or correlation for short.

[2] To be more precise: although the symmetry group of the plane contains more than just translations, the translations form a subgroup that acts on the plane. In the case of the sphere there is no coherent way to define a composition for points on the sphere, and so the sphere cannot act on itself (it is not a group). For this reason, we must consider the whole of SO(3).

presents us with a far-reaching generalization of the Fourier transform, which is applicable to signals on the sphere as well as the rotation group. It is known that the SO(3) correlation satisfies a Fourier theorem with respect to the SO(3) Fourier transform, and the same is true for our definition of S^2 correlation. Hence, the S^2 and SO(3) correlation can be implemented efficiently using generalized FFT algorithms.

Because we are the first to use cross-correlation on a continuous group inside a multi-layer neural network, we rigorously evaluate the degree to which the mathematical properties predicted by the continuous theory hold in practice for our discretized implementation.

Furthermore, we demonstrate the utility of spherical CNNs for rotation invariant classification and regression problems by experiments on three datasets. First, we show that spherical CNNs are much better at rotation invariant classification of Spherical MNIST images than planar CNNs. Second, we use the CNN for classifying 3D shapes. In a third experiment we use the model for molecular energy regression, an important problem in computational chemistry.

Contributions

The main contributions of this work are the following:

1. The theory of spherical CNNs.
2. The first automatically differentiable implementation of the generalized Fourier transform for S^2 and SO(3). Our PyTorch code is easy to use, fast, and memory efficient.
3. The first empirical support for the utility of spherical CNNs for rotation-invariant learning problems.

4.2 Related Work

It is well understood that the power of CNNs stems in large part from their ability to exploit (translational) symmetries though a combination of weight sharing and translation equivariance. It thus becomes natural to consider generalizations that exploit larger groups of symmetries, and indeed this has been the subject of several recent papers by Gens and Domingos, 2014; Olah, 2014; Dieleman et al., 2015; Dieleman et al., 2016; Cohen and Welling, 2016; Ravanbakhsh et al., 2017a; Zaheer et al., 2017; Guttenberg et al., 2016; Cohen and

Welling, 2017. With the exception of SO(2)-steerable networks (Worrall et al., 2017; Weiler et al., 2018b), these networks are all limited to discrete groups, such as discrete rotations acting on planar images or permutations acting on point clouds. Other very recent work is concerned with the analysis of spherical images, but does not define an equivariant architecture (Su and Grauman, 2017a; Boomsma and Frellsen, 2017). Our work is the first to achieve equivariance to a continuous, non-commutative group (SO(3)), and the first to use the generalized Fourier transform for fast group correlation. A preliminary version of this work appeared as Cohen et al., 2017.

To efficiently perform cross-correlations on the sphere and rotation group, we use generalized FFT algorithms. Generalized Fourier analysis, sometimes called abstract- or noncommutative harmonic analysis, has a long history in mathematics and many books have been written on the subject (Sugiura, 1990; Taylor, 1986; Folland, 1995). For a good engineering-oriented treatment which covers generalized FFT algorithms, see (Chirikjian and Kyatkin, 2001). Other important works include (Driscoll and Healy, 1994; Healy et al., 2003; Potts et al., 1998; Kunis and Potts, 2003; Drake et al., 2008; Maslen, 1998; Rockmore, 2004; Kostelec and Rockmore, 2007; Kostelec and Rockmore, 2008; Potts et al., 2009; Makadia et al., 2007; Gutman et al., 2008).

4.3 Correlation on the Sphere and Rotation Group

We will explain the S^2 and SO(3) correlation by analogy to the classical planar $\mathbb{Z}^2$ correlation. The planar correlation can be understood as follows:

> The value of the output feature map at translation $x \in \mathbb{Z}^2$ is computed as an inner product between the input feature map and a filter, shifted by x.

Similarly, the spherical correlation can be understood as follows:

> The value of the output feature map evaluated at rotation $R \in \mathrm{SO}(3)$ is computed as an inner product between the input feature map and a filter, rotated by R.

Because the output feature map is indexed by a rotation, it is modelled as a function on SO(3). We will discuss this issue in more detail shortly.

The above definition refers to various concepts that we have not yet defined mathematically. In what follows, we will go through the required concepts one

by one and provide a precise definition. Our goal for this section is only to present a mathematical model of spherical CNNs. Generalized Fourier theory and implementation details will be treated later.

The Unit Sphere S^2 can be defined as the set of points $x \in \mathbb{R}^3$ with norm 1. It is a two-dimensional manifold, which can be parameterized by spherical coordinates $\alpha \in [0, 2\pi]$ and $\beta \in [0, \pi]$.

Spherical Signals We model spherical images and filters as continuous functions $f : S^2 \to \mathbb{R}^K$, where K is the number of channels.

Rotations The set of rotations in three dimensions is called SO(3), the "special orthogonal group". Rotations can be represented by 3×3 matrices that preserve distance (i.e. $\|Rx\| = \|x\|$) and orientation ($\det(R) = +1$). If we represent points on the sphere as 3D unit vectors x, we can perform a rotation using the matrix-vector product Rx. The rotation group SO(3) is a three-dimensional manifold, and can be parameterized by ZYZ-Euler angles $\alpha \in [0, 2\pi]$, $\beta \in [0, \pi]$, and $\gamma \in [0, 2\pi]$.

Rotation of Spherical Signals In order to define the spherical correlation, we need to know not only how to rotate points $x \in S^2$ but also how to rotate filters (i.e. functions) on the sphere. To this end, we introduce the rotation operator L_R that takes a function f and produces a rotated function $L_R f$ by composing f with the rotation R^{-1}:

$$[L_R f](x) = f(R^{-1}x). \tag{4.1}$$

Due to the inverse on R, we have $L_{RR'} = L_R L_{R'}$.

Inner products The inner product on the vector space of spherical signals is defined as:

$$\langle \psi, f \rangle = \int_{S^2} \sum_{k=1}^{K} \psi_k(x) f_k(x) dx, \tag{4.2}$$

The integration measure dx denotes the standard rotation invariant integration measure on the sphere, which can be expressed as $d\alpha \sin(\beta) d\beta / 4\pi$ in spherical coordinates (see Appendix A). The invariance of the measure ensures that $\int_{S^2} f(Rx) dx = \int_{S^2} f(x) dx$, for any rotation $R \in \mathrm{SO}(3)$. That is, the volume under a spherical heightmap does not change when rotated. Using this fact, we can

show that $L_{R^{-1}}$ is adjoint to L_R, which implies that L_R is unitary:

$$\begin{aligned} \langle L_R \psi, f \rangle &= \int_{S^2} \sum_{k=1}^{K} \psi_k(R^{-1}x) f_k(x) dx \\ &= \int_{S^2} \sum_{k=1}^{K} \psi_k(x) f_k(Rx) dx \\ &= \langle \psi, L_{R^{-1}} f \rangle. \end{aligned} \tag{4.3}$$

Spherical Correlation With these ingredients in place, we are now ready to state mathematically what was stated in words before. For spherical signals f and ψ, we define the correlation as:

$$[\psi \star f](R) = \langle L_R \psi, f \rangle = \int_{S^2} \sum_{k=1}^{K} \psi_k(R^{-1}x) f_k(x) dx. \tag{4.4}$$

As mentioned before, the output of the spherical correlation is a function on SO(3). This is perhaps somewhat counterintuitive, and indeed the conventional definition of spherical convolution gives as output a function on the sphere. However, as shown in Appendix B, the conventional definition effectively restricts the filter to be circularly symmetric about the Z axis, which would greatly limit the expressive capacity of the network.

Rotation of SO(3) **Signals** We defined the rotation operator L_R for spherical signals (eq. 4.1), and used it to define spherical cross-correlation (eq. 4.4). To define the SO(3) correlation, we need to generalize the rotation operator so that it can act on signals defined on SO(3). As we will show, naively reusing eq. 4.1 is the way to go. That is, for $f : \mathrm{SO}(3) \to \mathbb{R}^K$, and $R, Q \in \mathrm{SO}(3)$:

$$[L_R f](Q) = f(R^{-1}Q). \tag{4.5}$$

Note that while the argument $R^{-1}x$ in Eq. 4.1 denotes the rotation of $x \in S^2$ by $R^{-1} \in \mathrm{SO}(3)$, the analogous term $R^{-1}Q$ in Eq. 4.5 denotes to the composition of rotations (i.e. matrix multiplication).

Rotation Group Correlation Using the same analogy as before, we can define the correlation of two signals on the rotation group, $f, \psi : \mathrm{SO}(3) \to \mathbb{R}^K$, as follows:

$$[\psi \star f](R) = \langle L_R \psi, f \rangle = \int_{\mathrm{SO}(3)} \sum_{k=1}^{K} \psi_k(R^{-1}Q) f_k(Q) dQ. \tag{4.6}$$

The integration measure dQ is the invariant measure on SO(3), which may be expressed in ZYZ-Euler angles as $d\alpha \sin(\beta) d\beta d\gamma/(8\pi^2)$ (see Appendix A).

Equivariance As we have seen, correlation is defined in terms of the rotation operator L_R. This operator acts naturally on the input space of the network, but what justification do we have for using it in the second layer and beyond?

The justification is provided by an important property, shared by all kinds of convolution and correlation, called equivariance. A layer Φ is equivariant if $\Phi \circ L_R = T_R \circ \Phi$, for some operator T_R. Using the definition of correlation and the unitarity of L_R, showing equivariance is a one liner:

$$[\psi \star [L_Q f]](R) = \langle L_R \psi, L_Q f \rangle = \langle L_{Q^{-1}R} \psi, f \rangle = [\psi \star f](Q^{-1}R) = [L_Q[\psi \star f]](R).$$

The derivation is valid for spherical correlation as well as rotation group correlation.

4.4 Fast Spherical Correlation with G-FFT

It is well known that correlations and convolutions can be computed efficiently using the Fast Fourier Transform (FFT). This is a result of the Fourier theorem, which states that $\widehat{f * \psi} = \hat{f} \cdot \hat{\psi}$. Since the FFT can be computed in $O(n \log n)$ time and the product $\cdot$ has linear complexity, implementing the correlation using FFTs is asymptotically faster than the naive $O(n^2)$ spatial implementation.

For functions on the sphere and rotation group, there is an analogous transform, which we will refer to as the generalized Fourier transform (GFT) and a corresponding fast algorithm (GFFT). This transform finds it roots in the representation theory of groups, but due to space constraints we will not go into details here and instead refer the interested reader to Sugiura (1990) and Folland (1995).

Conceptually, the GFT is nothing more than the linear projection of a function onto a set of orthogonal basis functions called "matrix element of irreducible unitary representations". For the circle (S^1) or line ($\mathbb{R}$), these are the familiar complex exponentials $\exp(in\theta)$. For SO(3), we have the Wigner D-functions $D^l_{mn}(R)$ indexed by $l \geq 0$ and $-l \leq m, n \leq l$. For S^2, these are the spherical harmonics[3] $Y^l_m(x)$ indexed by $l \geq 0$ and $-l \leq m \leq l$.

[3] Technically, S^2 is not a group and therefore does not have irreducible representations, but it is a quotient of groups SO(3)/SO(2) and we have the relation $Y^l_m = D^l_{m0}|_{S^2}$

Denoting the manifold (S^2 or SO(3)) by X and the corresponding basis functions by U^l (which is either vector-valued (Y^l) or matrix-valued (D^l)), we can write the GFT of a function $f : X \to \mathbb{R}$ as

$$\hat{f}^l = \int_X f(x)\overline{U^l(x)}dx. \tag{4.7}$$

This integral can be computed efficiently using a GFFT algorithm (see Section 4.4.1).

The inverse SO(3) Fourier transform is defined as:

$$f(R) = \sum_{l=0}^{b}(2l+1)\sum_{m=-l}^{l}\sum_{n=-l}^{l}\hat{f}^l_{mn}D^l_{mn}(R), \tag{4.8}$$

and similarly for S^2. The maximum frequency b is known as the bandwidth, and is related to the resolution of the spatial grid (Kostelec and Rockmore, 2007).

Using the well-known (in fact, defining) property of the Wigner D-functions that $D^l(R)D^l(R') = D^l(RR')$ and $D^l(R^{-1}) = D^l(R)^\dagger$, it can be shown (see Appendix D) that the SO(3) correlation satisfies a Fourier theorem[4]: $\widehat{\psi \star f} = \hat{f} \cdot \hat{\psi}^\dagger$, where $\cdot$ denotes matrix multiplication of the two block matrices $\hat{f}$ and $\hat{\psi}^\dagger$.

Similarly, using $Y(Rx) = D(R)Y(x)$ and $Y^l_m = D^l_{m0}|_{S^2}$, one can derive an analogous S^2 convolution theorem: $\widehat{\psi \star f}^l = \hat{f}^l \cdot \hat{\psi}^{l\dagger}$, where $\hat{f}^l$ and $\hat{\psi}^l$ are now vectors. This says that the SO(3)-FT of the S^2 correlation of two spherical signals can be computed by taking the outer product of the S^2-FTs of the signals. This is shown in figure 4.2.

4.4.1 Implementation of G-FFT and Spectral G-Conv

Here we sketch the implementation of GFFTs. For details, see (Kostelec and Rockmore, 2007).

The input of the SO(3) FFT is a spatial signal f on SO(3), sampled on a discrete grid and stored as a 3D array. The axes correspond to the ZYZ-Euler angles α, β, γ. The first step of the SO(3)-FFT is to perform a standard 2D translational FFT over the α and γ axes. The FFT'ed axes correspond to the m, n axes of the result. The second and last step is a linear contraction of the β

[4]This result is valid for real functions. For complex functions, conjugate ψ on the left hand side.

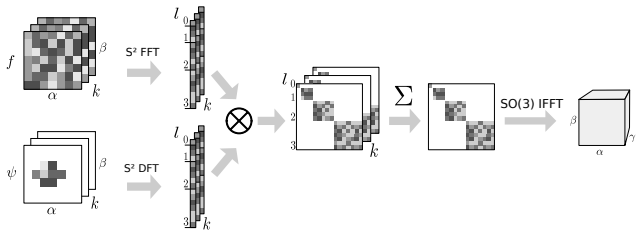


Figure 4.2: *Spherical correlation in the spectrum. The signal f and the locally-supported filter ψ are Fourier transformed, block-wise tensored, summed over input channels, and finally inverse transformed. Note that because the filter is locally supported, it is faster to use a matrix multiplication (DFT) than an FFT algorithm for it. We parameterize the sphere using spherical coordinates α, β, and* SO(3) *with ZYZ-Euler angles α, β, γ.*

axis of the FFT'ed array with a precomputed array of samples from the Wigner-d (small-d) functions $d^l_{mn}(\beta)$. Because the shape of d^l depends on l (it is $(2l+1)\times(2l+1)$), this linear contraction is implemented as a custom GPU kernel. The output is a set of Fourier coefficients $\hat{f}^l_{mn}$ for $l \geq n, m \geq -l$ and $l = 0, \ldots, L_{\max}$.

The algorithm for the S^2-FFTs is very similar, only in this case we FFT over the α axis only, and do a linear contraction with precomputed Legendre functions over the β axis.

Our code is available at `https://github.com/jonas-koehler/s2cnn`.

4.5 Experiments

In a first sequence of experiments, we evaluate the numerical stability and accuracy of our algorithm. In a second sequence of experiments, we showcase that the new cross-correlation layers we have introduced are indeed useful building blocks for several real problems involving spherical signals. Our examples for this are recognition of 3D shapes and predicting the atomization energy of molecules.

4.5.1 Equivariance error

In this paper we have presented the first instance of a group equivariant CNN for a continuous, non-commutative group. In the discrete case, one can prove that the network is exactly equivariant, but although we can prove $[L_R f] * \psi = L_R[f * \psi]$ for continuous functions f and ψ on the sphere or rotation group, this is not exactly true for the discretized version that we actually compute. Hence, it is reasonable to ask if there are any significant discretization artifacts and whether they affect the equivariance properties of the network. If equivariance can not be maintained for many layers, one may expect the weight sharing scheme to become much less effective.

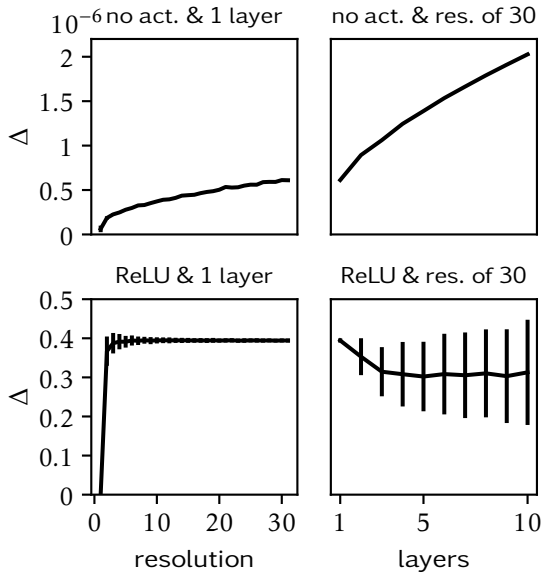


Figure 4.3: *Δ as a function of the resolution and the number of layers.*

We first tested the equivariance of the SO(3) correlation at various resolutions b. We do this by first sampling $n = 500$ random rotations R_i as well as n feature maps f_i with $K = 10$ channels. Then we compute $\Delta = \frac{1}{n}\sum_{i=1}^{n} \mathrm{std}(L_{R_i}\Phi(f_i) - \Phi(L_{R_i} f_i))/\mathrm{std}(\Phi(f_i))$, where Φ is a composition of SO(3) correlation layers with randomly initialized filters. In case of perfect equivariance, we expect this quantity to be zero. The results (figure 4.3 (top)), show that although the approximation error Δ grows with the resolution and the number of layers, it stays manageable for the range of resolutions of interest.

We repeat the experiment with ReLU activation function after each correlation operation. As shown in figure 4.3 (bottom), the error is higher but stays flat. This indicates that the error is not due to the network layers, but due to the feature map rotation, which is exact only for bandlimited functions.

4.5.2 Rotated MNIST on the Sphere

In this experiment we evaluate the generalization performance with respect to rotations of the input. For testing we propose a version MNIST dataset projected on the sphere (see fig. 4.4). We created two instances of this dataset: one in which each digit is projected on the northern hemisphere and one in which each projected digit is additionally randomly rotated.

Architecture and Hyperparameters As a baseline model, we use a simple CNN with layers conv-ReLU-conv-ReLU-FC-softmax, with filters of size 5×5, $k = 32, 64, 10$ channels, and stride 3 in both layers ($\approx$ 68K parameters). We compare to a spherical CNN with layers S^2conv-ReLU-SO(3)conv-ReLU-FC-softmax, bandwidth $b = 30, 10, 6$ and $k = 20, 40, 10$ channels ($\approx$ 58K parameters).

Figure 4.4: *Two MNIST digits projected onto the sphere using stereographic projection. Mapping back to the plane results in non-linear distortions.*

Results We trained each model on the non-rotated (NR) and the rotated (R) training set and evaluated it on the non-rotated and rotated test set. See table 4.1. While the planar CNN achieves high accuracy in the NR / NR regime, its performance in the R / R regime is much worse, while the spherical CNN is unaffected. When trained on the non-rotated dataset and evaluated on the rotated dataset (NR / R), the planar CNN does no better than random chance. The spherical CNN shows a slight decrease in performance compared to *R/R*, but still performs very well.

	NR / NR	R / R	NR / R
planar	0.98	0.23	0.11
spherical	0.96	0.95	0.94

Table 4.1: *Test accuracy for the networks evaluated on the spherical MNIST dataset. Here R = rotated, NR = non-rotated and X / Y denotes, that the network was trained on X and evaluated on Y.*

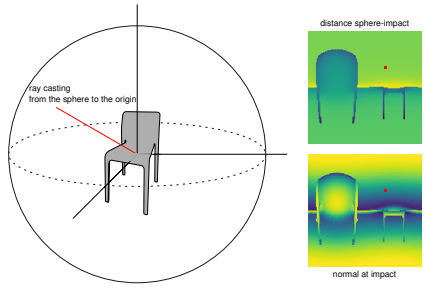


Figure 4.5: *The ray is cast from the surface of the sphere towards the origin. The first intersection with the model gives the values of the signal. The two images of the right represent two spherical signals in (α, β) coordinates. They contain respectively the distance from the sphere and the cosine of the ray with the normal of the model. The red dot corresponds to the pixel set by the red line.*

4.5.3 Recognition of 3D Shapes

Next, we applied S^2CNN to 3D shape classification. The SHREC17 task (Savva et al., 2016) contains 51300 3D models taken from the ShapeNet dataset (Chang et al., n.d.) which have to be classified into 55 common categories (tables, airplanes, persons, etc.). There is a consistently aligned regular dataset and a version in which all models are randomly perturbed by rotations. We concentrate on the latter to test the quality of our rotation equivariant representations learned by S^2CNN.

Representation We project the 3D meshes onto an enclosing sphere using a straightforward ray casting scheme (see Fig. 4.5). For each point on the sphere we send a ray towards the origin and collect 3 types of information from the intersection: ray length and cos/sin of the surface angle. We further augment this information with ray casting information for the convex hull of the model, which in total gives us 6 channels for the signal. This signal is discretized using a Driscoll-Healy grid (Driscoll and Healy, 1994) with bandwidth $b = 128$. Ignoring non-convexity of surfaces we assume this projection captures enough information of the shape to be useful for the recognition task.

Architecture and Hyperparameters Our network consists of an initial S^2conv-BN-ReLU block followed by two SO(3)conv-BN-ReLU blocks. The resulting fil-

Method	P@N	R@N	F1@N	mAP	NDCG
Tatsuma_ReVGG	0.705	0.769	0.719	0.696	0.783
Furuya_DLAN	0.814	0.683	0.706	0.656	0.754
SHREC16-Bai_GIFT	0.678	0.667	0.661	0.607	0.735
Deng_CM-VGG5-6DB	0.412	0.706	0.472	0.524	0.624
Ours	0.701 (3rd)	0.711 (2nd)	0.699 (3rd)	0.676 (2nd)	0.756 (2nd)

Table 4.2: *Results and best competing methods for SHREC17.*

ters are pooled using a max pooling layer followed by a last batch normalization and then fed into a linear layer for the final classification. It is important to note that the the max pooling happens over the group SO(3): if f_k is the k-th filter in the final layer (a function on SO(3)) the result of the pooling is $\max_{x \in SO(3)} f_k(x)$. We used 50, 70, and 350 features for the S^2 and the two SO(3) layers, respectively. Further, in each layer we reduce the resolution b, from 128, 32, 22 to 7 in the final layer. Each filter kernel ψ on SO(3) has non-local support, where $\psi(\alpha, \beta, \gamma) \neq 0$ iff $\beta = \frac{\pi}{2}$ and $\gamma = 0$ and the number of points of the discretization is proportional to the bandwidth in each layer. The final network contains ≈ 1.4M parameters, takes 8GB of memory at batch size 16, and takes 50 hours to train.

Results We evaluated our trained model using the official metrics and compared to the top three competitors in each category (see table 4.2 for results). Except for precision and F1@N, in which our model ranks third, it is the runner up on each other metric. The main competitors, Tatsuma_ReVGG and Furuya_DLAN use input representations and network architectures that are highly specialized to the SHREC17 task. Given the rather task agnostic architecture of our model and the lossy input representation we use, we interpret our models performance as strong empirical support for the effectiveness of Spherical CNNs.

4.5.4 Prediction of Atomization Energies

Finally, we apply S^2CNN on molecular energy regression. In the QM7 task the atomization energy of molecules has to be predicted from geometry and charges (Blum and Reymond, 2009; Rupp et al., 2012). Molecules contain up to $N = 23$ atoms of $T = 5$ types (H, C, N, O, S). They are given as a list of positions p_i and charges z_i for each atom i.

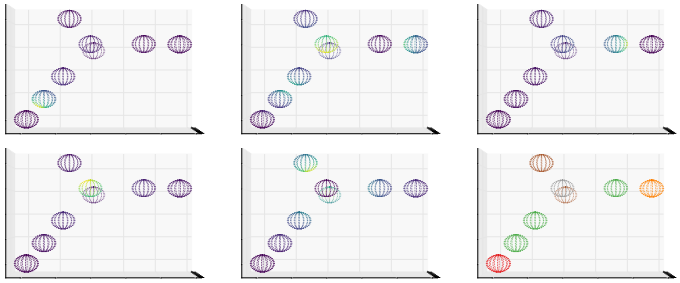


Figure 4.6: *The five potential channels U_z with $z \in \{1, 6, 7, 8, 16\}$ for a molecule containing atoms H (red), C (green), N (orange), O (brown), S (gray).*

Representation by Coulomb matrices Rupp et al., 2012 propose a rotation and translation invariant representation of molecules by defining the *Coulomb matrix* $C \in \mathbb{R}^{N \times N}$ (CM). For each pair of atoms $i \neq j$ they set $C_{ij} = (z_i z_j)/(|p_i - p_j|)$ and $C_{ii} = 0.5 z_i^{2.4}$. Diagonal elements encode the atomic energy by nuclear charge, while other elements encode Coulomb repulsion between atoms. This representation is not permutation invariant. To this end Rupp et al., 2012 propose a distance measure between Coulomb matrices used within Gaussian kernels whereas Montavon et al., 2012 propose sorting C or random sampling index permutations.

Representation as a spherical signal We utilize spherical symmetries in the geometry by defining a sphere S_i around around p_i for each atom i. The radius is kept uniform across atoms and molecules and chosen minimal such that no intersections among spheres in the training set happen. Generalizing the Coulomb matrix approach we define for each possible z and for each point x on S_i potential functions $U_z(x) = \sum_{j \neq i, z_j = z} \frac{z_i \cdot z}{|x - p_i|}$ producing a T channel spherical signal for each atom in the molecule (see figure 4.6). This representation is invariant with respect to translations and equivariant with respect to rotations. However, it is still not permutation invariant. The signal is discretized using a Driscoll-Healy (Driscoll and Healy, 1994) grid with bandwidth $b = 10$ representing the molecule as a sparse $N \times T \times 2b \times 2b$ tensor.

Architecture and Hyperparameters We use a deep ResNet style S^2CNN. Each ResNet block is made of S^2/SO(3)conv-BN-ReLU-SO(3)conv-BN after which

S^2CNN	Layer	Bandwidth	Features
	Input		5
	ResBlock	10	20
	ResBlock	8	40
	ResBlock	6	60
	ResBlock	4	80
	ResBlock	2	160
DeepSet	Layer	Input/Hidden	
	ϕ (MLP)	160/150	
	ψ (MLP)	100/50	

Table 4.3: *ResNet architecture for the molecule task.*

the input is added to the result. We share weights among atoms making filters permutation invariant, by pushing the atom dimension into the batch dimension. In each layer we downsample the bandwidth, while increasing the number of features F. After integrating the signal over SO(3) each molecule becomes a $N \times F$ tensor. For permutation invariance over atoms we follow Zaheer et al., 2017 and embed each resulting feature vector of an atom into a latent space using a MLP ϕ. Then we sum these latent representations over the atom dimension and get our final regression value for the molecule by mapping with another MLP ψ. Both ϕ and ψ are jointly optimized. Training a simple MLP only on the 5 frequencies of atom types in a molecule already gives a RMSE of ~ 19. Thus, we train the S^2CNN on the residual only, which improved convergence speed and stability over direct training. The final architecture is sketched in table 4.3. It has about 1.4M parameters, consumes 7GB of memory at batch size 20, and takes 3 hours to train.

Method	Author	RMSE
MLP / random CM	(a)	5.96
LGIKA(RF)	(b)	10.82
RBF kernels / random CM	(a)	11.40
RBF kernels / sorted CM	(a)	12.59
MLP / sorted CM	(a)	16.06
Ours		**8.47**

Table 4.4: *Experiment results for the QM7 task: (a) Montavon et al., 2012 (b) Raj et al., 2016.*

Results We evaluate by RMSE and compare our results to Montavon et al., 2012 and Raj et al., 2016 (see table 4.4). Our learned representation outper-

forms all kernel-based approaches and a MLP trained on sorted Coulomb matrices. Superior performance could only be achieved for an MLP trained on randomly permuted Coulomb matrices. However, sufficient sampling of random permutations grows exponentially with N, so this method is unlikely to scale to large molecules.

4.6 Discussion & Conclusion

In this paper we have presented the theory of Spherical CNNs and evaluated them on two important learning problems. We have defined S^2 and SO(3) cross-correlations, analyzed their properties, and implemented a Generalized FFT-based correlation algorithm. Our numerical results confirm the stability and accuracy of this algorithm, even for deep networks. Furthermore, we have shown that Spherical CNNs can effectively generalize across rotations, and achieve near state-of-the-art results on competitive 3D Model Recognition and Molecular Energy Regression challenges, without excessive feature engineering and task-tuning.

For intrinsically volumetric tasks like 3D model recognition, we believe that further improvements can be attained by generalizing further beyond SO(3) to the roto-translation group SE(3). The development of Spherical CNNs is an important first step in this direction. Another interesting generalization is the development of a Steerable CNN for the sphere (Cohen and Welling, 2017), which would make it possible to analyze vector fields such as global wind directions, as well as other sections of vector bundles over the sphere.

Perhaps the most exciting future application of the Spherical CNN is in omnidirectional vision. Although very little omnidirectional image data is currently available in public repositories, the increasing prevalence of omnidirectional sensors in drones, robots, and autonomous cars makes this a very compelling application of our work.

Appendix A: parameterization of and integration on S^2 and SO(3)

We use the ZYZ Euler parameterization for SO(3). An element $R \in \mathrm{SO}(3)$ is written as

$$R = R(\alpha, \beta, \gamma) = Z(\alpha)Y(\beta)Z(\gamma), \tag{4.9}$$

where $\alpha \in [0, 2\pi]$, $\beta \in [0, \pi]$ and $\gamma \in [0, 2\pi]$, and Z resp. Y are rotations around the Z and Y axes.

Using this parameterization, the normalized Haar measure is

$$dR = \frac{d\alpha}{2\pi} \frac{d\beta \sin(\beta)}{2} \frac{d\gamma}{2\pi} \tag{4.10}$$

We have $\int_{\mathrm{SO}(3)} dR = 1$. The Haar measure (Nachbin, 1965; Chirikjian and Kyatkin, 2001) is sometimes called the invariant measure because it has the property that $\int_{\mathrm{SO}(3)} f(R'R)dR = \int_{\mathrm{SO}(3)} f(R)dR$ (this is analogous to the more familiar property $\int_{\mathbb{R}} f(x+y)dx = \int_{\mathbb{R}} f(x)dx$ for functions on the line). This invariance property allows us to do many useful substitutions.

We have a related parameterization for the sphere. An element $x \in S^2$ is written

$$x(\alpha, \beta) = Z(\alpha)Y(\beta)n \tag{4.11}$$

where n is the north pole.

This parameterization makes explicit the fact that the sphere is a quotient $S^2 = \mathrm{SO}(3)/\mathrm{SO}(2)$, where $H = \mathrm{SO}(2)$ is the subgroup of rotations around the Z axis. Elements of this subgroup H leave the north pole invariant, and have the form $Z(\gamma)$. The point $x(\alpha, \beta) \in S^2$ is associated with the coset representative $\bar{x} = R(\alpha, \beta, 0) \in \mathrm{SO}(3)$. This element represents the coset $\bar{x}H = \{R(\alpha, \beta, \gamma) | \gamma \in [0, 2\pi]\}$.

The normalized Haar measure for the sphere is

$$dx = \frac{d\alpha}{2\pi} \frac{d\beta \sin\beta}{2} \tag{4.12}$$

The normalized Haar measure for SO(2) is

$$dh = \frac{d\gamma}{2\pi} \tag{4.13}$$

So we have $dR = dx\,dh$, again reflecting the quotient structure.

We can think of a function on S^2 as a γ-invariant function on SO(3). Given a function $f : S^2 \to \mathbb{C}$ we associate the function $\bar{f}(\alpha, \beta, \gamma) = f(\alpha, \beta)$. When using normalized Haar measures, we have:

$$\begin{aligned}\int_{\mathrm{SO}(3)} \bar{f}(R)dR &= \frac{1}{8\pi^2}\int_0^{2\pi} d\alpha \int_0^{\pi} \sin\beta d\beta \int_0^{2\pi} d\gamma \bar{f}(\alpha,\beta,\gamma) \\ &= \frac{1}{8\pi^2}\int_0^{2\pi} d\alpha \int_0^{\pi} \sin\beta d\beta f(\alpha,\beta) \int_0^{2\pi} d\gamma \\ &= \frac{1}{4\pi}\int_0^{2\pi} d\alpha \int_0^{\pi} \sin\beta d\beta f(\alpha,\beta) \\ &= \int_{S^2} f(x)dx\end{aligned} \tag{4.14}$$

This will allow us to define the Fourier transform on S^2 from the Fourier transform on SO(3), by viewing a function on S^2 as a γ-invariant function on SO(3) and taking its SO(3)-Fourier transform.

Appendix B: Correlation & Equivariance

We have defined the S^2 correlation as

$$[\psi \star f](R) = \langle L_R\psi, f\rangle = \int_{S^2} \sum_{k=1}^{K} \psi_k(R^{-1}x) f_k(x)dx. \tag{4.15}$$

Without loss of generality, we will analyze here the single-channel case $K = 1$.

This operation is equivariant:

$$\begin{aligned}[\psi \star [L_Q f]](R) &= \int_{S^2} \psi(R^{-1}x) f(Q^{-1}x)dx \\ &= \int_{S^2} \psi(R^{-1}Qx) f(x)dx \\ &= \int_{S^2} \psi((Q^{-1}R)^{-1}x) f(x)dx \\ &= [\psi \star f](Q^{-1}R) \\ &= [L_Q[\psi \star f]](R)\end{aligned} \tag{4.16}$$

A similar derivation can be made for the SO(3) correlation.

The spherical convolution defined by Driscoll and Healy, 1994 is:

$$[f * \psi](x) = \int_{\mathrm{SO}(3)} f(Rn)\psi(R^{-1}x)dR \tag{4.17}$$

where n is the north pole. Note that in this definition, the output of the spherical convolution is a function on the sphere, not a function on SO(3) as in our definition of cross-correlation. Note further that unlike our definition, this definition involves an integral over SO(3).

If we write out the integral in terms of Euler angles, noting that the north-pole n is invariant to Z-axis rotations by γ, i.e. $R(\alpha,\beta,\gamma)n = Z(\alpha)Y(\beta)Z(\gamma)n = Z(\alpha)Y(\beta)n$, we see that this definition implicitly integrates over γ in only one of the factors (namely ψ), making it invariant wrt γ rotation. In other words, the filter is first "averaged" (making it circularly symmetric) before it is combined with f (This was observed before by Makadia et al., 2007). We consider this to be much too limited for the purpose of pattern matching in spherical CNNs.

Appendix C: Generalized Fourier Transform

With each compact topological group (like SO(3)) is associated a discrete set of orthogonal functions that arise as matrix elements of irreducible unitary representations of these groups. For the circle (the group SO(2)) these are the complex exponentials (in the complex case) or sinusoids (for real functions). For SO(3), these functions are known as the Wigner D-functions.

As discussed in the paper, the Wigner D-functions are parameterized by a degree parameter $l \geq 0$ and order parameters $m, n \in [-l, \dots, l]$. In other words, we have a set of matrix-valued functions $D^l : \mathrm{SO}(3) \to \mathbb{C}^{(2l+1)\times(2l+1)}$.

The Wigner D-functions are orthogonal:

$$\langle D^l_{mn}, D^{l'}_{m'n'} \rangle = \int_0^{2\pi} \frac{d\alpha}{2\pi} \int_0^{\pi} \frac{d\beta \sin\beta}{2} \int_0^{2\pi} \frac{d\gamma}{2\pi} D^l_{mn}(\alpha,\beta,\gamma)\overline{D^{l'}_{m'n'}}(\alpha,\beta,\gamma) = \frac{\delta_{ll'}\delta_{mm'}\delta_{nn'}}{2l+1} \tag{4.18}$$

Furthermore, they are complete, meaning that any well behaved function $f : \mathrm{SO}(3) \to \mathbb{C}$ can be written as a linear combination of Wigner D-functions. This is the idea of the Generalized Fourier Transform $\mathcal{F}$ on SO(3):

$$f(R) = [\mathcal{F}^{-1}\hat{f}](R) = \sum_{l=0}^{\infty}(2l+1)\sum_{m=-l}^{l}\sum_{n=-l}^{l}\hat{f}^l_{mn}D^l_{mn}(R) \tag{4.19}$$

where $\hat{f}^l_{mn}$ are called the Fourier coefficients of f. Using the orthogonality property of the Wigner D-functions, one can see that the Fourier coefficients can be retrieved by computing the inner product with the Wigner D-functions:

$$\begin{aligned}
[\mathcal{F}f]^l_{mn} &= \int_{\mathrm{SO}(3)} f(R)\overline{D^l_{mn}(R)}dR \\
&= \int_{\mathrm{SO}(3)} \left[\sum_{l'=0}^{\infty}(2l'+1)\sum_{m'=-l'}^{l'}\sum_{n'=-l'}^{l'}\hat{f}^{l'}_{m'n'}D^{l'}_{m'n'}(R)\right]\overline{D^l_{mn}(R)}dR \\
&= \sum_{l'=0}^{\infty}(2l'+1)\sum_{m'=-l'}^{l'}\sum_{n'=-l'}^{l'}\hat{f}^{l'}_{m'n'}\int_{\mathrm{SO}(3)} D^{l'}_{m'n'}(R)\overline{D^l_{mn}}dR \\
&= \hat{f}^l_{mn}
\end{aligned} \tag{4.20}$$

Appendix D: Fourier Theorems

Fourier convolution theorems for SO(3) and $\S^2$ can be found in Kostelec and Rockmore, 2008; Makadia et al., 2007; Gutman et al., 2008. We derive them here for completeness.

To derive the convolution theorems, we will use the defining property of the Wigner D-matrices: that they are (irreducible, unitary) *representations* of SO(3). This means that they satisfy:

$$D^l(R)D^l(R') = D^l(RR'), \tag{4.21}$$

for any $R, R' \in \mathrm{SO}(3)$. Notice that the complex exponentials satisfy an analogous criterion for the circle group $S^1 \cong \mathrm{SO}(2)$. That is, $e^{inx}e^{iny} = e^{in(x+y)}$, where $x+y$ is the group operation for SO(2).

Unitarity means that $D^l(R)D^{l\dagger}(R) = I$. Irreducibility means, essentially, that the set of matrices $\{D^l(R)|R \in \mathrm{SO}(3)\}$ cannot be simultaneously block-diagonalized.

To derive the Fourier theorem for SO(3), we use the invariance of the integration measure dR: $\int_{\mathrm{SO}(3)} f(R'R)dR = \int_{\mathrm{SO}(3)} f(R)dR$.

With these facts understood, we can proceed to derive:

$$\begin{aligned}
\widehat{\psi \star f}^l &= \int_{SO(3)} (\psi \star f)(R)\overline{D^l(R)}dR \\
&= \int_{SO(3)} \int_{SO(3)} \psi(R^{-1}R')f(R')dR'\overline{D^l(R)}dR \\
&= \int_{SO(3)} \int_{SO(3)} \psi(R^{-1})f(R')\overline{D^l(R'R)}dR'dR \\
&= \int_{SO(3)} f(R')\overline{D^l(R')}dR' \int_{SO(3)} \psi(R^{-1})\overline{D^l(R)}dR \\
&= \int_{SO(3)} f(R')\overline{D^l(R')}dR' \int_{SO(3)} \psi(R)\overline{D^l(R)}^\dagger dR \\
&= \hat{f}^l \, \hat{\psi}^{l\dagger}
\end{aligned} \tag{4.22}$$

So the SO(3)-Fourier transform of the SO(3) convolution of ψ and f is equal to the matrix product of the SO(3)-Fourier transforms $\hat{f}$ and $\hat{\psi}$.

For the sphere, we can derive an analogous transform that is sometimes called the spherical harmonics transform. The spherical harmonics $Y^l_m : S^2 \to \mathbb{C}$ are a complete orthogonal family of functions. The spherical harmonics are related to the Wigner D functions by the relation $D^l_{mn}(\alpha, \beta, \gamma) = Y^l_m(\alpha, \beta)e^{in\gamma}$, so that $Y^l_m(\alpha, \beta) = D^l_{m0}(\alpha, \beta, 0)$.

The S^2 convolution of f_1 and f_2 is equivalent to the SO(3) convolution of the associated right-invariant functions $\bar{f}_1, \bar{f}_2$ (see Appendix A):

$$\begin{aligned}
[f_1 \star f_2](R) &= \int_{S^2} f_1(R^{-1}x)f_2(x)dx \\
&= \int_{SO(2)} \int_{S^2} f_1(R^{-1}x)f_2(x)dxdh \\
&= \int_{SO(3)} \bar{f}_1(R^{-1}R')\bar{f}_2(R')dR' \\
&= [\bar{f}_1 \star \bar{f}_2](R)
\end{aligned} \tag{4.23}$$

The Fourier transform of a right invariant function on SO(3) equals

$$\begin{aligned}
[\mathcal{F}\bar{f}]^l_{mn} &= \int_0^{2\pi} \frac{d\alpha}{2\pi} \int_0^{\pi} \frac{d\beta \sin\beta}{2} \int_0^{2\pi} \frac{d\gamma}{2\pi} \bar{f}(\alpha,\beta,\gamma)\overline{D^l_{mn}(\alpha,\beta,\gamma)} \\
&= \int_0^{2\pi} \frac{d\alpha}{2\pi} \int_0^{\pi} \frac{d\beta \sin\beta}{2} f(\alpha,\beta) \int_0^{2\pi} \frac{d\gamma}{2\pi} \overline{D^l_{mn}(\alpha,\beta,\gamma)} \\
&= \delta_{n0} \int_0^{2\pi} \frac{d\alpha}{2\pi} \int_0^{\pi} \frac{d\beta \sin\beta}{2} f(\alpha,\beta)\overline{D^l_{m0}(\alpha,\beta,0)} \\
&= \delta_{n0} \int_{S^2} f(x)\overline{Y^l_m(x)} dx
\end{aligned} \tag{4.24}$$

So we can think of the S^2 Fourier transform of a function on S^2 as the $n = 0$ column of the SO(3) Fourier transform of the associated right-invariant function. This is a beautiful result that we have not been able to find a reference for, though it seems likely that it has been observed before.

5

Gauge CNNs

Based on: (Cohen et al., 2019).

The principle of *equivariance to symmetry transformations* enables a theoretically grounded approach to neural network architecture design. Equivariant networks have shown excellent performance and data efficiency on vision and medical imaging problems that exhibit symmetries. Here we show how this principle can be extended beyond global symmetries to local gauge transformations. This enables the development of a very general class of convolutional neural networks on manifolds that depend only on the intrinsic geometry, and which includes many popular methods from equivariant and geometric deep learning.

5.1 Introduction

By and large, progress in deep learning has been achieved through intuition-guided experimentation. This approach is indispensable and has led to many successes, but has not produced a deep understanding of *why and when* certain architectures work well. As a result, every new application requires an extensive architecture search, which comes at a significant labor and energy cost.

Although a theory that tells us which architecture to use for any given problem is clearly out of reach, we can nevertheless come up with *general principles* to guide architecture search. One such rational design principle that has met with substantial empirical success (Winkels and Cohen, 2018a; Zaheer et al.,

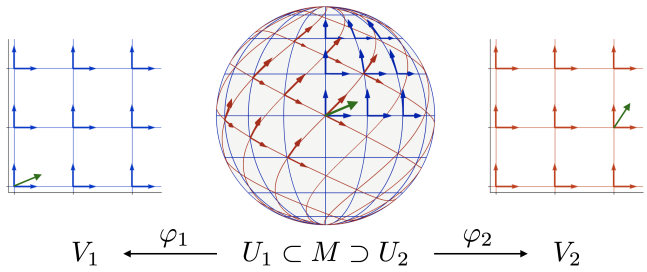


Figure 5.1: *A gauge is a smoothly varying choice of tangent frame on a subset U of a manifold M. A gauge is needed to represent geometrical quantities such as convolutional filters and feature maps (i.e. fields), but the choice of gauge is ultimately arbitrary. Hence, the network should be equivariant to gauge transformations, such as the change between red and blue gauge pictured here.*

2017; Lunter and Brown, 2018) maintains that network architectures should be equivariant to symmetries.

Besides the ubiquitous translation equivariant CNN, equivariant networks have been developed for sets, graphs, and homogeneous spaces like the sphere (see Sec. 5.3). In each case, the network is made equivariant to the global symmetries of the underlying space. However, manifolds do not in general have global symmetries, and so it is not obvious how one might develop equivariant CNNs for them.

General manifolds do however have *local gauge symmetries*, and as we will show in this paper, taking these into account is not just useful but *necessary* if one wishes to build manifold CNNs that depend only on the intrinsic geometry. To this end, we define a convolution-like operation on general manifolds M that is equivariant to local gauge transformations (Fig. 5.1). This *gauge equivariant convolution* takes as input a number of *feature fields* on M of various types (analogous to matter fields in physics), and produces as output new feature fields. Each field is represented by a number of feature maps, whose activations are interpreted as the coefficients of a geometrical object (e.g. scalar, vector, tensor, etc.) relative to a spatially varying frame (i.e. gauge). The network is constructed such that if the gauge is changed, the coefficients change in a predictable way so as to preserve their geometrical meaning. Thus, the search for a geometrically natural definition of "manifold convolution", a key problem in geometric deep learning, leads inevitably to gauge equivariance.

Gauge theory plays a central role in modern physics, but has a reputation for being abstract and difficult. So in order to keep this chapter accessible to a broad machine learning audience, we have chosen to emphasize geometrical intuition over mathematical formality. In Part II, we give a more formal account of the general theory that encompasses group equivariant CNNs, steerable CNNs, and gauge equivariant CNNs.

The rest of this chapter is organized as follows. In Sec. 5.2, we motivate the need for working with gauges, and define gauge equivariant convolution for general manifolds and fields. In section 5.3, we discuss related work on equivariant and geometrical deep learning. This chapter is only concerned with the mathematical theory of gauge equivariant CNNs; In the next chapter, we will cover the implementation of a particular gauge equivariant CNN, namely the Icosahedral CNN, and its application to omnidirection image segmentation and global climate pattern segmentation.

5.2 Gauge Equivariant Networks

Consider the problem of generalizing the classical convolution of two planar signals (e.g. a feature map and a filter) to signals defined on a manifold M. The first and most natural idea comes from thinking of planar convolution in terms of *shifting* a filter over a feature map. Observing that shifts are symmetries of the plane (mapping the plane onto itself while preserving its structure), one is led to the idea of transforming a filter on M by the symmetries of M. For instance, replacing shifts of the plane by rotations of the sphere, one obtains Spherical CNNs (Cohen et al., 2018c), Chapter 4.

This approach works for any *homogeneous space*, where by definition it is possible to move from any point $p \in M$ to any other point $q \in M$ using an appropriate symmetry transformation (Kondor and Trivedi, 2018; Cohen et al., 2018b; Cohen et al., 2018a). On less symmetrical manifolds however, it may not be possible to move the filter from any point to any other point by symmetry transformations. Hence, transforming filters by symmetry transformations will in general not provide a recipe for weight sharing between filters at all points in M.

Instead of symmetries, one can move the filter by parallel transport (Schonsheck et al., 2018), but as shown in Fig. 5.2, this leaves an ambiguity in the filter orientation, because parallel transport is *path dependent*. This can be addressed by using only *rotation invariant* filters (Boscaini et al., 2015; Bruna et al., 2014), albeit at the cost of limiting expressivity.

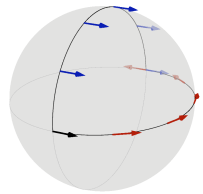


Figure 5.2: *On curved spaces, parallel transport is path dependent. The black vector is transported to the same point via two different curves, yielding different results. The same phenomenon occurs for other geometric objects, including filters.*

The key issue is that on a manifold, there is no preferred gauge (tangent frame), relative to which we can position our measurement apparatus (i.e. filters), and relative to which we can describe measurements (i.e. responses). We must choose a gauge in order to numerically represent geometrical quantities and perform computations, but since it is arbitrary, the computations should be independent of it.

This does not mean however that the *coefficients* of the feature vectors should be invariant to gauge transformations, but rather that the feature vector itself should be invariant. That is, a gauge transformation leads to a change of basis $e_i \mapsto \tilde{e}_i$ of the feature space (fiber) at $p \in M$, so the feature vector coefficients f_i should change equivariantly to ensure that the vector $\sum_i f_i e_i = \sum_i \tilde{f}_i \tilde{e}_i$ itself is unchanged.

Before showing how this is achieved, we note that on *non-parallelizable* manifolds such as the sphere, it is not possible to choose a smooth global gauge. For instance, if we extend the blue gauge pictured in Fig. 5.1 to the whole sphere, we will inevitably create a singularity where the gauge changes abruptly. Hence, in order to make the math work smoothly, it is standard practice in gauge theory to work with multiple gauges defined on overlapping charts, as in Fig. 5.1.

The basic idea of gauge equivariant convolution is as follows. Lacking alternative options, we choose arbitrarily a smooth *local* gauge on subsets $U \subset M$ (e.g. the red or blue gauge in Fig. 5.1). We can then position a filter at each point $p \in U$, defining its orientation relative to the gauge. Then, we match an input feature map against the filter at p to obtain the value of the output feature map at p. For the output to transform equivariantly, certain linear constraints are placed on the convolution kernel. We will now define this formally.

5.2.1 Gauges, Transformations, and Exponential Maps

We define a gauge as a position-dependent invertible linear map $w_p : \mathbb{R}^d \to T_pM$, where T_pM is the tangent space of M at p. This determines a frame $w_p(e_1), \ldots, w_p(e_d)$ in T_pM, where $\{e_i\}$ is the standard frame of $\mathbb{R}^d$.

A gauge transformation (Fig. 5.1) is a position-dependent change of frame, which can be described by maps $g_p \in \mathrm{GL}(d, \mathbb{R})$ (the group of invertible $d \times d$ matrices). As indicated by the subscript, the transformation g_p depends on the position $p \in U \subset M$. To change the frame, simply compose w_p with g_p, i.e. $w_p \mapsto w_p g_p$. It follows that component vectors $v \in \mathbb{R}^d$ transform as $v \mapsto g_p^{-1} v$, so that the vector $(w_p g_p)(g_p^{-1} v) = w_p v \in T_pM$ itself is invariant.

If we derive our gauge from a coordinate system for M (as shown in Fig. 5.1), then a change of coordinates leads to a gauge transformation (g_p being the Jacobian of the coordinate transformation at p). But we can also choose a gauge w_p independent of any coordinate system.

It is often useful to restrict the kinds of frames we consider, for example to only allow right-handed or orthogonal frames. Such restrictions limit the kinds of gauge transformations we can consider. For instance, if we allow only right-handed frames, g_p should have positive determinant (i.e. $g_p \in \mathrm{GL}^+(d, \mathbb{R})$), so that it does not reverse the orientation. If in addition we allow only orthogonal frames, g_p must be a rotation, i.e. $g_p \in \mathrm{SO}(d)$.

In mathematical terms, $G = \mathrm{GL}(d, \mathbb{R})$ is called the *structure group* of the theory, and limiting the kinds of frames we consider corresponds to a *reduction of the structure group* (Husemöller, 1994). Each reduction corresponds to some extra structure that is preserved, such as an *orientation* ($\mathrm{GL}^+(d, \mathbb{R})$) or *Riemannian metric* ($\mathrm{SO}(d)$). In the Icosahedral CNN (Chapter 6), we will want to preserve the hexagonal grid structure, which corresponds to a restriction to grid-aligned frames and a reduction of the structure group to $G = C_6$, the group of planar rotations by integer multiples of $2\pi/6$. For the rest of this chapter, we will work in the Riemannian setting, i.e. use $G = \mathrm{SO}(d)$.

Before we can define gauge equivariant convolution, we will need the exponential map, which gives a convenient parameterization of the local neighbourhood of $p \in M$. This map $\exp_p : T_pM \to M$ takes a tangent vector $V \in T_pM$, follows the geodesic (shortest curve) in the direction of V with speed $\|V\|$ for one unit of time, to arrive at a point $q = \exp_p V$ (see Fig. 5.3, (Lee, 2018)).

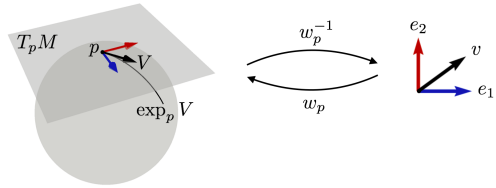


Figure 5.3: *The exponential map and the gauge w_p.*

5.2.2 Gauge Equivariant Convolution: Scalar Fields

Having defined gauges, gauge transformations, and the exponential map, we are now ready to define gauge equivariant convolution. We begin with scalar input and output fields.

We define a filter as a locally supported function $K : \mathbb{R}^d \to \mathbb{R}$, where $\mathbb{R}^d$ may be identified with T_pM *via the gauge* w_p. Then, writing $q_v = \exp_p w_p(v)$ for $v \in \mathbb{R}^d$, we define the scalar convolution of K and $f : M \to \mathbb{R}$ at p as follows:

$$(K \star f)(p) = \int_{\mathbb{R}^d} K(v) f(q_v) dv. \tag{5.1}$$

The gauge was chosen arbitrarily, so we must consider what happens if we change it. Since the filter $K : \mathbb{R}^d \to \mathbb{R}$ is a function of a coordinate vector $v \in \mathbb{R}^d$, and v gets rotated by gauge transformations, the effect of a gauge transformation is a position-dependent rotation of the filters. For the convolution output to be called a scalar field, it has to be invariant to gauge transformations (i.e. $v \mapsto g_p^{-1} v$ and $w_p \mapsto w_g g_p$). The only way to make $(K \star f)(p)$ (Eq. 5.1) invariant to rotations of the filter, is to make the filter rotation-invariant:

$$\forall g \in G \; : \; K(g^{-1} v) = K(v) \tag{5.2}$$

Thus, to map a scalar input field to a scalar output field in a gauge equivariant manner, we need to use rotationally symmetric filters. Some geometric deep learning methods, as well as graph CNNs do indeed use isotropic filters. However, this is very limiting and as we will now show, unnecessary if one considers non-scalar feature fields.

5.2.3 Feature Fields

Intuitively, a field is an assignment of some geometrical quantity (feature vector) $f(p)$ of the same type to each point $p \in M$. The type of a quantity is de-

termined by its transformation behaviour under gauge transformations. For instance, the word *vector field* is reserved for a field of tangent vectors whose coefficients transform like $v(p) \mapsto g_p^{-1} v(p)$ as we saw before. It is important to note that $f(p)$ is an element of a vector space ("fiber") $F_p \simeq \mathbb{R}^C$ attached to $p \in M$ (e.g. the tangent space T_pM). Thus, strictly speaking, f is not a function because the output space F_p changes with the argument p. All F_p are similar to a canonical feature space $\mathbb{R}^C$, but f can only be considered a function $U \to \mathbb{R}^C$ *locally*, after we have chosen a gauge, because there is no canonical way to identify all feature spaces F_p.

In the general case, the transformation behaviour of a C-dimensional geometrical quantity is described by a *representation of the structure group* G. This is a mapping $\rho : G \to \mathrm{GL}(C, \mathbb{R})$ that satisfies $\rho(gh) = \rho(g)\rho(h)$, where gh denotes the composition of transformations in G, and $\rho(g)\rho(h)$ denotes multiplication of $C \times C$ matrices $\rho(g)$ and $\rho(h)$. The simplest examples are the trivial representation $\rho(g) = 1$ which describes the transformation behaviour of scalars, and $\rho(g) = g$, which describes the transformation behaviour of (tangent) vectors. A field f that transforms like $f(p) \mapsto \rho(g_p^{-1}) f(p)$ will be called a ρ-field.

In Chapter 6 on Icosahedral CNNs, we will consider one more type of representation, namely the *regular representation* of C_6. The group C_6 can be described as the 6 planar rotations by $k \cdot 2\pi/6$, or as integers k with addition mod 6. Features that transform like the regular representation of C_6 are 6-dimensional, with one component for each rotation. One can obtain a regular feature by taking a filter at p, rotating it by $k \cdot 2\pi/6$ for $k = 0, \ldots, 5$, and matching each rotated filter against the input signal. When the gauge is changed, the filter and all rotated copies are rotated, and so the components of a regular C_6 feature are cyclically shifted. Hence, $\rho(g)$ is a 6×6 cyclic permutation matrix that shifts the coordinates by k' steps for $g = k' \cdot 2\pi/6$. Further examples of representations ρ that are useful in convolutional networks may be found in (Cohen and Welling, 2017; Weiler et al., 2018a; Thomas et al., 2018; Hy et al., 2018).

5.2.4 Gauge Equivariant Convolution: General Fields

Now consider a stack of C_{in} input feature maps on M, which represents a C_{in}-dimensional ρ_{in}-field (e.g. $C_{\text{in}} = 1$ for a single scalar, $C_{\text{in}} = d$ for a vector, $C_{\text{in}} = 6$ for a regular C_6 feature, or any multiple of these, etc.). We will define a convolution operation that takes such a field and produces as output a C_{out}-dimensional ρ_{out}-field. For this we need a filter bank with C_{out} output channels

and C_{in} input channels, which we will describe mathematically as a matrix-valued kernel $K : \mathbb{R}^d \to \mathbb{R}^{C_{\text{out}} \times C_{\text{in}}}$.

We can think of $K(v)$ as a linear map from the input feature space ("fiber") at p to the output feature space at p, these spaces being identified with $\mathbb{R}^{C_{\text{in}}}$ resp. $\mathbb{R}^{C_{\text{out}}}$ by the choice of gauge w_p at p. This suggests that we need to modify Eq. 5.1 to make sure that the kernel matrix $K(v)$ is multiplied by a feature vector at p, not one at $q_v = \exp_p w_p(v)$. This is achieved by transporting $f(q_v)$ to p along the unique[1] geodesic connecting them, before multiplying by $K(v)$.

As $f(q_v)$ is transported to p, it undergoes a transformation which will be denoted $g_{p \leftarrow q_v} \in G$ (see Fig. 5.2). This transformation acts on the feature vector $f(q_v) \in \mathbb{R}^{C_{\text{in}}}$ via the representation $\rho_{\text{in}}(g_{p \leftarrow q_v}) \in \mathbb{R}^{C_{\text{in}} \times C_{\text{in}}}$. Thus, we obtain the generalized form of Eq. 5.1 for general fields:

$$(K \star f)(p) = \int_{\mathbb{R}^d} K(v) \rho_{\text{in}}(g_{p \leftarrow q_v}) f(q_v) dv. \tag{5.3}$$

Under a gauge transformation, we have:

$$\begin{aligned} v &\mapsto g_p^{-1} v, & f(q_v) &\mapsto \rho_{\text{in}}(g_{q_v}^{-1}) f(q_v), \\ w_p &\mapsto w_p g_p, & g_{p \leftarrow q_v} &\mapsto g_p^{-1} g_{p \leftarrow q_v} g_{q_v}. \end{aligned} \tag{5.4}$$

For $K \star f$ to be well defined as a ρ_{out}-field, we want it to transform like $(K \star f)(p) \mapsto \rho_{\text{out}}(g_p^{-1})(K \star f)(p)$. Or, in other words, $\star$ should be gauge equivariant. This will be the case if and only if K satisfies

$$\forall g \in G : K(g^{-1} v) = \rho_{\text{out}}(g^{-1}) K(v) \rho_{\text{in}}(g). \tag{5.5}$$

One may verify this by making the substitutions of Eq. 5.4 in Eq. 5.3 and simplifying using $\rho(gh) = \rho(g)\rho(h)$ and Eq. 5.5, to find that $(K \star f)(p) \mapsto \rho_{\text{out}}(g_p^{-1})(K \star f)(p)$.

We note that equations 5.1 and 5.2 are special cases of 5.3 and 5.5 for $\rho_{\text{in}}(g) = \rho_{\text{out}}(g) = 1$, i.e. for scalar fields.

This concludes our presentation of the general case. A gauge equivariant $\rho_1 \to \rho_2$ convolution on M is defined relative to a local gauge by Eq. 5.3, where the kernel satisfies the equivariance constraint of Eq. 5.5. By defining gauges

[1] For points that are close enough, there is always a unique geodesic. Since the kernel has local support, p and q_v will be close for all non-zero terms.

on local charts $U_i \subset M$ that cover M and convolving inside each one, we automatically get a globally well-defined operation, because switching charts corresponds to a gauge transformation (Fig. 5.1), and the convolution is gauge equivariant.

5.2.5 Locally Flat Spaces

On flat regions of the manifold, the exponential parameterization can be simplified to $\varphi(\exp_p w_p(v)) = \varphi(p) + v$ if we use an appropriate local coordinate $\varphi(p) \in \mathbb{R}^d$ of $p \in M$. Moreover, in such a flat chart, parallel transport is trivial, i.e. $g_{p \leftarrow q_v}$ equals the identity. Thus, on a flat region, our convolution boils down to a standard convolution / correlation:

$$(K \star f)(x) = \int_{\mathbb{R}^d} K(v) f(x+v) dv. \tag{5.6}$$

Moreover, we can recover group convolutions, spherical convolutions, and convolution on other homogeneous spaces as special cases as well (Appendix A).

5.3 Related work

Equivariant Deep Learning Equivariant networks have been proposed for permutation-equivariant analysis and prediction of sets (Zaheer et al., 2017; Hartford et al., 2018), graphs (Kondor et al., 2018b; Hy et al., 2018; Maron et al., 2019), translations and rotations of the plane and 3D space (Oyallon and Mallat, 2015; Cohen and Welling, 2016; Cohen and Welling, 2017; Marcos et al., 2017; Weiler et al., 2018b; Weiler et al., 2018a; Worrall et al., 2017; Worrall and Brostow, 2018; Winkels and Cohen, 2018a; Veeling et al., 2018; Thomas et al., 2018; Bekkers et al., 2018; Hoogeboom et al., 2018), and the sphere (see below). Ravanbakhsh et al. (2017b) studied finite group equivariance. Equivariant CNNs on homogeneous spaces were studied by (Kondor and Trivedi, 2018) (scalar fields) and (Cohen et al., 2018b; Cohen et al., 2018a) (general fields). In this paper we generalize G-CNNs from homogeneous spaces to general manifolds.

Geometric Deep Learning Geometric deep learning (Bronstein et al., 2016) is concerned with the generalization of (convolutional) neural networks to manifolds. Many definitions of manifold convolution have been proposed, and

some of them (those called "intrinsic") are gauge equivariant (although to the best of our knowledge, the relevance of gauge theory has not been observed before). However, these methods are all limited to particular feature types ρ (typically scalar), and/or use a parameterization of the kernel that is not maximally flexible.

Bruna et al. (2014) and Boscaini et al. (2015) propose to use isotropic (spectral) filters (i.e. scalar field features), while (Masci et al., 2015) define a convolution that is essentially the same as our scalar-to-regular convolution, followed by a max-pooling over orientations, which in our terminology maps a regular field to a scalar field. As shown experimentally in (Cohen and Welling, 2016; Cohen and Welling, 2017) and in this paper, it is often more effective to use convolutions that preserve orientation information (e.g. regular to regular convolution). Another solution is to align the filter with the maximum curvature direction (Boscaini et al., 2016), but this approach is not intrinsic and does not work for flat surfaces or uniformly curved spaces like spheres.

(Poulenard and Ovsjanikov, 2018) define a multi-directional convolution for "directional functions" (somewhat similar to what we call regular fields), but they parameterize the kernel by a *scalar* function on the tangent space, which is very limited compared to our matrix-valued kernel (which is the most general kernel mapping ρ_1 fields to ρ_2 fields).

Spherical CNNs Besides the general theoretical framework of gauge equivariant convolution, we present in this paper a specific model (the Icosahedral CNN), which can be viewed as a fast and simple alternative to Spherical CNNs (Cohen et al., 2018c; Esteves et al., 2018; Boomsma and Frellsen, 2017; Su and Grauman, 2017b; Perraudin et al., 2018; Jiang et al., 2018; Kondor et al., 2018a). Liu et al. (2019) use a spherical grid based on a subdivision of the icosahedron, and convolve over it using a method that is similar to the one presented in Chapter 6 (and thus ignores curvature), but this method is not equivariant and does not take into account gauge transformations. We show in Sec. 6.7 that both are important for optimal performance.

Mathematics & physics To deeply understand gauge equivariant networks, we recommend studying the mathematics of gauge theory: principal & associated fiber bundles (Schuller, 2016; Husemöller, 1994; Steenrod, 1951). The work presented in this paper can be understood as replacing the principal G-bundle $H \to H/G$ used in G-CNNs over homogeneous spaces H/G (Cohen et al., 2018a) by the frame bundle of M, which is another principal G-bundle.

For more information on manifolds, fiber bundles, connections, parallel transport, the exponential map, etc., we highly recommend the lectures by Schuller, 2016, as well as the book Nakahara, 2003 which explain these concepts very clearly and at a useful level of abstraction. For further study, we recommend Sharpe, 1997; Kobayashi and Nomizu, 1963; Husemöller, 1994; Steenrod, 1951; Wendl, 2008; Crane, 2014. More details can be found in the appendix.

5.4 Conclusion

In this paper we have presented the general theory of gauge equivariant convolutional networks on manifolds. Although we have only touched on the connections to physics and geometry, there are indeed interesting connections, which we plan to elaborate on in the future. From the perspective of the mathematical theory of principal fiber bundles, our definition of manifold convolution is entirely natural. Indeed it is clear that gauge invariance is not just nice to have, but *necessary* in order for the convolution to be geometrically well-defined.

Acknowledgements We would like to thank Chiyu "Max" Jiang and Mayur Mudigonda for help obtaining and interpreting the climate data, and Erik Verlinde for helpful discussions.

Appendix A: Mathematical Theory & Physics Analogy

From the perspective of the theory of principal fiber bundles, our work can be understood as follows. A fiber bundle E is a space consisting of a base space B (the manifold M in our paper), with at each point $p \in B$ a space F_p called the fiber at p. The bundle is defined in terms of a projection map $\pi : E \to B$, which determines the fibers as $F_p = \pi^{-1}(p)$. A principal bundle is a fiber bundle where the fiber F carries a transitive and free right action of a group G (the structure group).

One can think of the fiber F_p of a principal bundle as a (generalized) space of frames at p. Due to the free and transitive action of G on F_p, we have that F_p is isomoprhic to G as a G-space, meaning that it looks like G except that it

does not have a distinguished origin or identity element as G does (i.e. there is no natural choice of frame).

A gauge transformation is then defined as a principal bundle automorphism, i.e. a map from $P \to P$ that maps fibers to fibers in a G-equivariant manner. Sometimes the automorphism is required to fix the base space, i.e. project down to the identity map via π. Such a B-automorphism will map each fiber onto itself, so it restricts to a G-space automorphism on each fiber.

Given a principal bundle P and a vector space V with representation ρ of G, we can construct the associated bundle $P \times_\rho V$, whose elements are the equivalence classes of the following equivalence relation on $P \times V$:

$$(p, v) \sim (pg, \rho(g^{-1})v). \tag{5.7}$$

The associated bundle is a fiber bundle over the same base space as P, with fiber isomorphic to V.

A (matter) field is described as a section of the associated bundle A, i.e. a map $\sigma : B \to A$ that satisfies $\pi \circ \sigma = 1_B$. Locally, one can describe a section as a function $B \to V$ (as we do in the paper), but globally this is not possible unless the bundle is trivial.

The group of automorphisms of P (gauge transformations) acts on the space of fields (sections of the associated bundle). It is this group that we wish to be equivariant to.

From this mathematical perspective, our work amounts to replacing the principal G bundle[2] $H \to H/G$ used in the work on regular and steerable G-CNNs of Cohen et al. (2018a) and Cohen et al. (2018b), by another principal G bundle, namely the frame bundle of M. Hence, this general theory can describe in a unified way the most prominent and geometrically natural methods of geometrical deep learning Masci et al., 2015; Boscaini et al., 2016, as well as all G-CNNs on homogeneous spaces.

Indeed, if we build a gauge equivariant CNN with spatial weight sharing on a homogeneous space H/G (e.g. the sphere $S^2 = \mathrm{SO}(3)/\mathrm{SO}(2)$), it will (under mild conditions) automatically be equivariant to the left action of H also. To see this, note that the left action of H on itself (the total space of the principal G bundle) can be decomposed into an action on the base space H/G (permuting the fibers), and an action on the fibers (cosets) that factors through G (see e.g. Sec. 2.1 of Cohen et al., 2018b). The action on the base space preserves the local

[2] It is more common to use the letter G for the supergroup and H for the subgroup, but that leads to a principal H-bundle $G \to G/H$, which is inconsistent with the main text, where we use a principal G bundle. So we swap H and G here.

neighbourhoods from which we compute filter responses, and equivariance to the action of G is ensured by the kernel constraint. Since G-CNNs (Cohen et al., 2018a) and gauge equivariant CNNs employ the most general equivariant map, we conclude that they are indeed the same, for bundles $H \to H/G$. Thus, "gauge theory is all you need". (We plan to expand this argument in a future paper)

Many modern theories of physics are gauge theories, meaning they are based on this mathematical framework. In such theories, any construction is required to be gauge invariant (i.e. the coefficients must be gauge equivariant), for otherwise the predictions will depend on the way in which we choose to represent physical quantities. This logic applies not just to physics theories, but, as we have argued in the paper, also to neural networks and other models used in machine learning. Hence, it is only natural that the same mathematical framework is applicable in both fields.

Appendix B: Deriving the kernel constraint

The gauge equivariant convolution is given by

$$(K \star f)(p) = \int_{\mathbb{R}^d} K(v)\rho_{\text{in}}(g_{p\leftarrow q_v})f(q_v)dv. \tag{5.8}$$

Under a gauge transformation, we have:

$$\begin{aligned} v &\mapsto g_p^{-1}v, & f(q_v) &\mapsto \rho_{\text{in}}(g_{q_v}^{-1})f(q_v), \\ w_p &\mapsto w_p g_p, & g_{p\leftarrow q_v} &\mapsto g_p^{-1} g_{p\leftarrow q_v} g_{q_v}. \end{aligned} \tag{5.9}$$

It follows that q_v is unchanged, because $q_v = \exp_p w_p v \mapsto \exp_p(w_p g_p)(g_p^{-1}v) = q_v$. Substituting the rest in the convolution equation, we find

$$\begin{aligned} &\int_{\mathbb{R}^d} K(g_p^{-1}v)\rho_{\text{in}}(g_p^{-1}g_{p\leftarrow q_v}g_{q_v})\rho_{\text{in}}(g_{q_v}^{-1})f(q_v)dv \\ &\quad = \int_{\mathbb{R}^d} K(g_p^{-1}v)\rho_{\text{in}}(g_p^{-1})\rho_{\text{in}}(g_{p\leftarrow q_v})f(q_v)dv \end{aligned} \tag{5.10}$$

Now if $K(g_p^{-1}v) = \rho_{\text{out}}(g_p^{-1})K(v)\rho_{\text{in}}(g_p)$ (i.e. K satisfies the kernel constraint), then we get

$$(K \star f)(p) \mapsto \rho_{\text{out}}(g_p^{-1})(K \star f)(p), \tag{5.11}$$

i.e. $K \star f$ transforms as a ρ_{out}-field under gauge transformations.

6

Icosahedral CNNs

Based on: (Cohen et al., 2019).

In this chapter we will describe a concrete method for performing gauge equivariant convolution (Chapter 5) on the icosahedron. The very special shape of this manifold makes it possible to implement gauge equivariant convolution in a way that is both numerically convenient (no interpolation is required), and computationally efficient (the heavy lifting is done by a single conv2d call).

6.1 The Icosahedron

The icosahedron is a regular solid with 20 faces, 30 edges, and 12 vertices (see Fig. 6.1, left). It has 60 rotational symmetries. This symmetry group will be denoted[1] $\mathcal{I}$.

6.2 The Hexagonal Grid

Whereas general manifolds, and even spheres, do not admit completely regular and symmetrical pixelations, we can define an almost perfectly regular grid of pixels on the icosahedron. This grid is constructed through a sequence of grid-refinement steps. We begin with a grid $\mathcal{H}_0$ consisting of the corners of the icosahedron itself. Then, for each triangular face, we subdivide it into 4

[1] As an abstract group, $\mathcal{I} \simeq A5$ (the alternating group A5), but we use $\mathcal{I}$ to emphasize that it is realized by a set of 3D rotations.

smaller triangles, thus introducing 3 new points on the center of the edges of the original triangle. This process is repeated r times to obtain a grid $\mathcal{H}_r$ with $N = 5 \times 2^{2r+1} + 2$ points (Fig. 6.1, left).

Each grid point (pixel) in the grid has 6 neighbours, except for the corners of the icosahedron, which have 5. Thus, one can think of the non-corner grid points as hexagonal pixels, and the corner points as pentagonal pixels.

Notice that the grid $\mathcal{H}_r$ is perfectly symmetrical, which means that if we apply an icosahedral symmetry $g \in \mathcal{I}$ to a point $p \in \mathcal{H}_r$, we will always land on another grid point, i.e. $gp \in \mathcal{H}_r$. Thus, in addition to talking about gauge equivariance, for this manifold / grid, we can also talk about (exact) equivariance to *global transformations* (3D rotations in $\mathcal{I}$). Because these global symmetries act by permuting the pixels and changing the gauge, one can see that a gauge equivariant network is automatically equivariant to global transformations. This will be demonstrated in Section 6.7.

6.3 The Atlas of Charts

We define an *atlas* consisting of 5 overlapping *charts* on the icosahedron, as shown in Fig. 6.1. Each chart is an invertible map $\varphi_i : U_i \to V_i$, where $U_i \subset \mathcal{H}_r \subset M$ and $V_i \subset \mathbb{Z}^2$. The regions U_i and V_i are shown in Fig. 6.1. The maps themselves are linear on faces, and defined by hard-coded correspondences $\varphi_i(c_j) = x_j$ between the corner points c_j in $\mathcal{H}_r$ and points x_j in the planar grid $\mathbb{Z}^2$.

Each chart covers all the points in 4 triangular faces of the icosahedron. Together, the 5 charts cover all 20 faces of the icosahedron.

We divide the charts into an *exterior* $\overline{V}_i \subset V_i$, consisting of border pixels, and an *interior* $V_i^\circ \subset V_i$, consisting of pixels whose neighbours are all contained in chart i. In order to ensure that every pixel in $\mathcal{H}_r$ (except for the 12 corners) is contained in the interior of some chart, we add a strip of pixels to the left and bottom of each chart, as shown in Fig. 6.1 (center). Then the interior of each chart (plus two exterior corners) has a nice rectangular shape $2^r \times 2^{r+1}$, and every non-corner is contained in exactly one interior V_i°.

So if we know the values of the field in the interior of each chart, we know the whole field (except for the corners, which we ignore). However, in order to compute a valid convolution output at each interior pixel (assuming a hexagonal filter with one ring, i.e. a 3×3 masked filter), we will still need the exterior pixels to be filled in as well (introducing a small amount of redundancy). See Sec. 6.6.1.

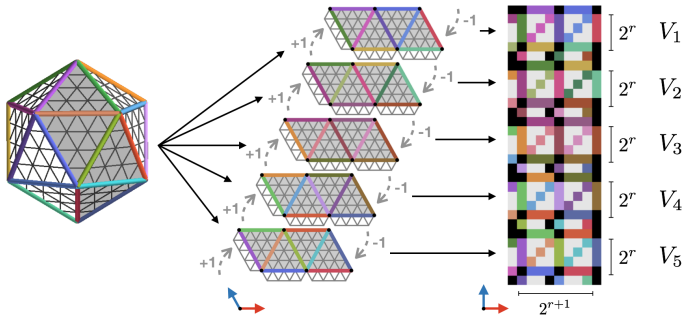


Figure 6.1: *The Icosahedron with grid $\mathcal{H}_r$ for $r = 2$ (left). We define 5 overlapping charts that cover the grid (center). Chart V_5 is highlighted in gray (left). Colored edges that appear in multiple charts are to be identified. In each chart, we define the gauge by the standard axis aligned basis vectors $e_1, e_2 \in V_i$. For points $p \in U_i \cap U_j$, the transition between charts involves a change of gauge, shown as $+1 \cdot 2\pi/6$ and $-1 \cdot 2\pi/6$ (elements of $G = C_6$). On the right we show how the signal is represented in a padded array of shape $5 \cdot (2^r + 2) \times (2^{r+1} + 2)$.*

6.4 The Gauge

For the purpose of computation, we fix a convenient gauge in each chart. This gauge is defined in each V_i as the constant orthogonal frame $e_1 = (1,0), e_2 = (0,1)$, aligned with the x and y direction of the plane (just like the red and blue gauge in Fig. 5.1). When mapped to the icosahedron via (the Jacobian of) φ_i^{-1}, the resulting frames are aligned with the grid, and the basis vectors make an angle of $2 \cdot 2\pi/6$.

Some pixels $p \in U_i \cap U_j$ are covered by multiple charts. Although the local frames $e_1 = (1,0), e_2 = (0,1)$ are numerically constant and equal in both charts V_i and V_j, the corresponding frames on the icosahedron (obtained by pushing them though φ_i^{-1} and φ_j^{-1}) may not be the same. In other words, when switching from chart i to chart j, there may be a gauge transformation $g_{ij}(p)$, which rotates the frame at $p \in U_i \cap U_j$ (see Fig. 6.1).

For the particular atlas defined in Sec. 6.3, the gauge transformations $g_{ij}(p)$ are always elements of the group C_6 (i.e. rotations by $k \cdot 2\pi/6$), so $G = C_6$ and we have a C_6-atlas.

6.5 The Signal Representation

A stack of C feature fields is represented as an array of shape $(B, C, R, 5, H, W)$, where B is the batch size, C the number of fields, R is the dimension of the fields ($R = 1$ for scalars and $R = 6$ for regular features), 5 is the number of charts, and H, W are the height and width of each local chart ($H = 2^r + 2$ and $W = 2^{r+1} + 2$ at resolution r, including a 1-pixel padding region on each side, see Fig. 6.1). We can always reshape such an array to shape $(B, CR, 5H, W)$, resulting in a $4D$ array that can be viewed as a stack of CR rectangular feature maps of shape $(5H, W)$. Such an array can be input to conv2d.

6.6 Gauge Equivariant Icosahedral Convolution

Gauge equivariant convolution on the icosahedron is implemented in three steps: G-Padding, kernel expansion, and 2d convolution / HexaConv Hoogeboom et al., 2018.

6.6.1 G-Padding

In a standard CNN, we can only compute a valid convolution output at positions (x, y) where the filter fits inside the input image in its entirety. If the output is to be of the same size as the input, one uses zero padding. Likewise, the IcoConv requires padding, only now the padding border $\overline{V}_i$ of a chart consists of pixels that are also represented in the interior of another chart (Sec. 6.3). So instead of zero padding, we copy the pixels from the neighbouring chart. We always use hexagonal filters with 1 ring, which can be represented as a 3×3 filter on a square grid, so we pad by 1 pixel.

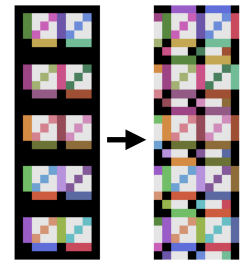


Figure 6.2: *G-Padding (scalar signal)*

As explained in Sec. 6.4, when transitioning between charts one may have to perform a gauge transformation on the features. Since scalars are invariant quantities, transition padding amounts to a simple copy in this case. Regular C_6 features (having 6 orientation channels) transform by cyclic shifts $\rho(g_{ij}(p))$ (Sec. 5.2.3), where $g_{ij} \in \{+1, 0, -1\} \cdot 2\pi/6$ (Fig. 6.1), so we must cyclically shift the channels up or down before copying to get

the correct coefficients in the new chart. The whole padding operation is implemented by four indexing + assignment operations (top, bottom, left, right) using fixed pre-computed indices (see appendix).

6.6.2 Weight Sharing & Kernel Expansion

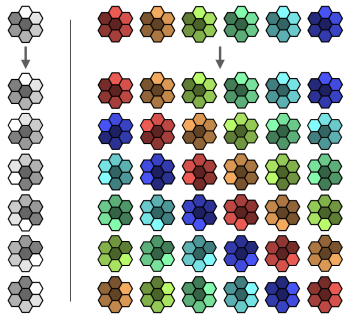


Figure 6.3: *Kernel expansion for scalar-to-regular ($R_{in} = 1, R_{out} = 6$; left) and regular-to-regular ($R_{in} = R_{out} = 6$; right) convolution. Top: free parameters. Bottom: expanded kernel used in conv2d.*

For the convolution to be gauge equivariant, the kernel must satisfy Eq. 5.5. The kernel $K : \mathbb{R}^2 \to \mathbb{R}^{R_{\text{out}}C_{\text{out}} \times R_{\text{in}}C_{\text{in}}}$ is stored in an array of shape

$$(R_{\text{out}}C_{\text{out}}, R_{\text{in}}C_{\text{in}}, 3, 3),$$

with the top-right and bottom-left pixel of each 3×3 filter fixed at zero so that it corresponds to a 1-ring hexagonal kernel.

Eq. 5.5 says that if we transform the input channels (columns) by $\rho_{\text{in}}(g)$ and the ouput channels (rows) by $\rho_{\text{out}}(g)$, the result should equal the original kernel where each channel is rotated by $g \in C_6$. This will the case if we use the weight-sharing scheme shown in Fig. 6.3.

Weight sharing can be implemented in two ways. One can construct a basis of kernels, each of which has shape $(R_{\text{out}}, R_{\text{in}}, 3, 3)$ and has value 1 at all pixels of a certain color/shade, and 0 elsewhere. Then one can construct the full kernel by linearly combining these basis filters using learned weights (one for each $C_{\text{in}} \cdot C_{\text{out}}$ input/output channels and basis kernel) Cohen and Welling, 2017; Weiler et al., 2018a. Alternatively, for scalar and regular features, one can use a set of precomputed indices to expand the kernel as shown in Fig. 6.3, using a single indexing operation.

6.6.3 Complete Algorithm

The complete algorithm can be summarized as

$$\text{GConv(f, w)} = \text{conv2d(GPad(f), expand(w))}. \tag{6.1}$$

Where f and GPad(f) both have shape $(B, C_{\text{in}}R_{\text{in}}, 5H, W)$, the weights w have shape $(C_{\text{out}}, C_{\text{in}}R_{\text{in}}, 7)$, and expand($w$) has shape $(C_{\text{out}}R_{\text{out}}, C_{\text{in}}R_{\text{in}}, 3, 3)$. The output of GConv has shape $(B, C_{\text{out}}R_{\text{out}}, 5H, W)$.

On the flat faces, being described by one of the charts, this algorithm coincides exactly with the hexagonal regular convolution introduced in Hoogeboom et al., 2018. The non-zero curvature of the icosahedron requires us to do the additional step of padding between different charts.

6.7 Experiments

6.7.1 IcoMNIST

In order to validate our implementation, highlight the potential benefits of our method, and determine the necessity of each part of the algorithm, we perform a number of experiments with the MNIST dataset, projected to the icosahedron.

We generate three different versions of the training and test sets, differing in the transformations applied to the data. In the N condition, No rotations are applied to the data. In the I condition, we apply all 60 Icosahedral symmetries (rotations) to each digit. Finally, in the R condition, we apply 60 random continuous rotations $g \in \text{SO}(3)$ to each digit before projecting. All signals are represented as explained in Sec. 6.5 / Fig. 6.1 (right), using resolution $r = 4$, i.e. as an array of shape $(1, 5 \cdot (16 + 2), 32 + 2)$.

Our main model consists of one gauge equivariant scalar-to-regular (S2R) convolution layer, followed by 6 regular-to-regular (R2R) layers and 3 FC layers (see appendix for architectural details). We also evaluate a model that uses only S2R convolution layers, followed by orientation pooling (a max over the 6 orientation channels of each regular feature, thus mapping a regular feature to a scalar), as in Masci et al., 2015. Finally, we consider a model that uses only rotation-invariant filters, i.e. scalar-to-scalar (S2S) convolutions, similar to standard graph CNNs Boscaini et al., 2015; Kipf and Welling, 2017. We also compare to the fully SO(3)-equivariant but computationally costly Spherical CNN (S2CNN). See appendix for architectural details and computational complexity analysis.

In addition, we perform an ablation study where we disable each part of the algorithm. The first baseline is obtained from the full R2R network by disabling gauge padding (Sec. 6.6.1), and is called the No Pad (NP) network. In the second baseline, we disable the kernel Expansion (Sec. 6.6.2), yielding the NE condition. The third baseline, called NP+NE uses neither gauge padding nor kernel expansion, and amounts to a standard CNN applied to the same input representation. We adapt the number of channels so that all networks have roughly the same number of parameters.

As shown in Table 6.1, icosahedral CNNs achieve excellent performance with a test accuracy of up to 99.43%, which is a strong result even on planar MNIST, for non-augmented and non-ensembled models. The full R2R model performs best in all conditions (though not significantly in the N/N condition), showing that both gauge padding and kernel expansion are necessary, and that our general (R2R) formulation works better in practice than using scalar fields (S2S or S2R). We notice also that non-equivariant models (NP+NE, NP, NE) do not generalize well to transformed data, a problem that is only partly solved by data augmentation. On the other hand, the models S2S, S2R, and R2R are exactly equivariant to symmetries $g \in \mathcal{I}$, and so generalize perfectly to $\mathcal{I}$-transformed test data, even when these were not seen during training. None of the models automatically generalize to continuously rotated inputs (R), but the equivariant models are closer, and can get even closer (> 99%) when using SO(3) data augmentation during training. The fully SO(3)-equivariant S2CNN scores slightly worse than R2R, except in N/R and I/R, as expected. The slight decrease in performance of S2CNN for rotated training conditions is likely due to the fact that it has lower grid resolution near the equator. We note that the S2CNN is slower and less scalable than Ico CNNs (see appendix).

Arch.	N/N	N/I	N/R	I/ I	I / R	R / R
S2CNN	99.38	99.38	99.38	99.12	99.13	99.12
NP+NE	99.29	25.50	16.20	98.52	47.77	94.19
NE	99.42	25.41	17.85	98.67	60.74	96.83
NP	99.27	36.76	21.4	98.99	61.62	97.87
S2S	97.81	97.81	55.64	97.72	58.37	89.92
S2R	98.99	98.99	59.76	98.62	55.57	98.74
R2R	**99.43**	**99.43**	**69.99**	**99.38**	**66.26**	**99.31**

Table 6.1: *IcoMNIST test accuracy (%) for different architectures and train / test conditions (averaged over 3 runs). See text for explanation of labels.*

6.7.2 Climate Pattern Segmentation

We evaluate our method on the climate pattern segmentation task proposed by Mudigonda et al. (2017). The goal is to segment extreme weather events (Atmospheric Rivers (AR) and Tropical Cyclones (TC)) in climate simulation data.

We use the exact same data and evaluation methodology as Jiang et al., 2018. The preprocessed data as released by Jiang et al., 2018 consists of 16-channel spherical images at resolution $r = 5$, which we reinterpret as icosahedral signals (introducing slight distortion). See Mudigonda et al., 2017 for a detailed description of the data.

We compare an R2R and S2R model (details in appendix). As shown in Table 6.2, our models outperform both competing methods in terms of per-class and mean accuracy. The difference between our R2R and S2R model seems small in terms of accuracy, but when evaluated in terms of mean average precision (a more appropriate evaluation metric for segmentation tasks), the R2R model clearly outperforms.

Model	BG	TC	AR	Mean	mAP
Mudigonda et al.	97	74	65	78.67	-
Jiang et al.	97	94	93	94.67	-
Ours (S2R)	97.3	97.8	97.3	97.5	0.686
Ours (R2R)	97.4	**97.9**	**97.8**	**97.7**	**0.759**

Table 6.2: *Climate pattern segmentation accuracy (%) for BG, TC and AR classes plus mean accuracy and average precision (mAP).*

6.7.3 Stanford 2D-3D-S

For our final experiment, we evaluate icosahedral CNNs on the 2D-3D-S dataset Armeni et al., 2017, which consists of 1413 omnidirectional RGB+D images with pixelwise semantic labels in 13 classes. Following Jiang et al. (2018), we sample the data on a grid of resolution $r = 5$ using bilinear interpolation, while using nearest-neighbour interpolation for the labels. Evaluation is performed by mean intersection over union (mIoU) and pixel accuracy (mAcc).

The network architecture is a residual U-Net Ronneberger et al., 2015; He et al., 2016a with R2R convolutions. The network consists of a downsampling and upsampling network. The downsampling network takes as input a signal

at resolution $r = 5$ and outputs feature maps at resolutions $r = 4, \ldots, 1$, with $8, 16, 32$ and 64 channels. The upsampling network is the reverse of this. We pool over orientation channels right before applying softmax.

As shown in table 6.3, our method outperforms the method of Jiang et al., 2018, which in turn greatly outperforms standard planar methods such as U-Net on this dataset.

	mAcc	mIoU
Jiang et al., 2018	0.547	0.383
Ours (R2R-U-Net)	**0.559**	**0.394**

Table 6.3: *Mean accuracy and intersection over union for 2D-3D-S omnidirectional segmentation task.*

6.8 Conclusion

In this paper we have implemented gauge equivariant convolutional networks for the icosahedron, and demonstrated their utility for segmenting omnidirectional images and global climate patterns. We have demonstrated that this method performs well on a range of different problems and is highly scalable. Our chart-based approach to manifold convolution should in principle scale to very large problems, thus opening the door to learning from high-resolution planetary scale spherical signals that arise in the earth and climate sciences.

Appendix A: Additional information on experiments

6.8.1 MNIST experiments

Our main model consists of 7 convolution layers and 3 linear layers. The first layer is a scalar-to-regular gauge equivariant convolution layer, and the following 6 layers are regular-to-regular layers. These layers have $8, 16, 16, 24, 24, 32, 64$ output channels, and stride $1, 2, 1, 2, 1, 2, 1$, respectively.

In between convolution layers, we use batch normalization Ioffe and Szegedy, 2015 and ReLU nonlinearities. When using batch normalization, we average over groups of 6 feature maps, to make sure the operation is equivariant. Any pointwise nonlinearity (like ReLU) is equivariant, because we use only trivial and regular representations realized by permutation matrices.

After the convolution layers, we perform global pooling over spatial and orientation channels, yielding an invariant representation. We map these through 3 FC layers (with $64, 32, 10$ channels) before applying softmax.

The other models are obtained from this one by replacing the convolution layers by scalar-to-regular + orientation pooling (S2R) or scalar-to-scalar (S2S) layers, or by disabling G-padding (NP) and/or kernel expansion (NE), always adjusting the number of channels to keep the number of parameters roughly the same.

The Spherical CNN (S2CNN) is obtained from the R2R model by replacing the S2R and R2R layers by spherical and SO(3) convolution layers, respectively, keeping the number of channels and strides the same. The Spherical CNN uses a different grid than the Icosahedral CNN, so we adapt the resolution / bandwidth parameter B to roughly match the resolution of the Icosahedral CNN. We use $B = 26$, to get a spherical grid of size $2B \times 2B = 52 \times 52$. Note that this grid has higher resolution at the poles, and lower resolution near the equator, which explains why the S2CNN performs a bit worse when trained on rotated data instead of digits projected onto the north-pole. To implement strides, we reduce the output bandwidth by 2 at each layer with stride.

The spherical convolution takes a scalar signal on the sphere as input, and outputs scalar signals on SO(3), which is analogous to a regular field over the sphere. SO(3) convolutions are analogous to *R2R* layers. We note that this is a stronger Spherical CNN architecture than the one used by Cohen et al., 2018c, which achieves only 96% accuracy on spherical MNIST.

The models were trained for 60 epochs, or 1 epoch of the 60× augmented dataset (where each instance is transformed by each icosahedron symmetry $g \in \mathcal{I}$, or by a random rotation $g \in \mathrm{SO}(3)$).

6.8.2 Climate experiments

For the climate experiments, we used a U-net with regular-to-regular convolutions. The first layer is a scalar-to-regular convolution with 16 output channels. The downsampling path consists of 5 regular-to-regular layers with stride 2, and $32, 64, 128, 256, 256$ output channels. The downsampling path takes as input a signal with resolution $r = 5$ (i.e. 10242 pixels), and outputs one at $r = 0$ (i.e. 12 pixels).

The decoder is the reverse of the encoder in terms of resolution and number of channels. Upsampling is performed by bilinear interpolation (which is exactly equivariant), before each convolution layer (which uses stride 1). As usual in the U-net architecture, each layer in the upsampling path takes as input the output of the previous layer, as well as the output of the encoder path at the same resolution.

Each convolution layer is followed by equivariant batchnorm and ReLU.

The model was trained for 15 epochs with batchsize 15.

6.8.3 2D-3D-S experiments

For the 2D-3D-S experiments, we used a residual U-Net with the following architecture.

The input layer is a scalar-to-regular layer with 8 channels, followed by batchnorm and relu. Then we apply 4 residual blocks with 16, 32, 64, 64 output channels, each of which uses stride=2. In the upsampling stream, we use 32, 16, 8, 8 channels, for the residual blocks, respectively. Each upsampling layer receives input from the corresponding downsampling layer, as well as the previous layer. Upsampling is performed using bilinear interpolation, and downsampling by hexagonal max pooling.

The input resolution is $r = 5$, which is downsampled to $r = 1$ by the downsampling stream.

Each residual block consists of a convolution, batchnorm, skipconnection, and ReLU.

Appendix B: Computational complexity analysis of Spherical and Icosahedral CNNs

One of the primary motivations for the development of the Icosahedral CNN is that it is faster and more scalable than Spherical CNNs as originally pro-

posed. The Spherical CNN as implemented by Cohen et al., 2018c uses feature maps on the sphere S^2 and rotation group SO(3) (the latter of which can be thought of a regular field on the sphere), sampled on the SOFT grids defined by Kostelec and Rockmore, 2007, which have shape $2B \times 2B$ and $2B \times 2B \times 2B$, respectively (here B is the bandwidth / resolution parameter). Specifically, the grid points are:

$$\begin{aligned} \alpha_{j_1} &= \frac{2\pi j_1}{2B}, \\ \beta_k &= \frac{\pi(2k+1)}{4B}, \\ \gamma_{j_2} &= \frac{2\pi j_2}{2B}, \end{aligned} \tag{6.2}$$

where (α_{j_1}, β_k) form a spherical grid and $(\alpha_{j_1}, \beta_k, \gamma_{j_2})$ form an SO(3) grid (for $j_1, k, j_2 = 0, \dots 2B-1$). These grids have two downsides.

Firstly, because the SOFT grid consists of equal-lattitude rings with a fixed number of points (2B), the spatial density of points is inhomogeneous, with a higher concentration of points near the poles. To get a sufficiently high sampling near the equator, we are forced to oversample the poles, and thus waste computational resources. For almost all applications, a more homogeneous grid is more suitable.

The second downside of the SOFT grid on SO(3) is that the spatial resolution ($2B \times 2B$; α, β) and angular resolution ($2B$; γ) are both coupled to the same resolution / bandwidth parameter B. Thus, as we increase the resolution of the spherical image, the number of rotations applied to each filter is increased as well, which is undesirable.

The grid used in the Icosahedral CNN addresses both concerns. It is spatially very homogeneous, and we apply the filters in 6 orientations, regardless of spatial resolution.

The generalized FFT algorithm used by Cohen et al., 2018a only works on the SOFT grid. Generalized FFTs for other grids exist Kunis and Potts, 2003, but are harder to implement. Moreover, although the (generalized) FFT can improve the asymptotic complexity of convolution for large input signals, the FFT-based convolution actually has worse complexity if we assume a fixed filter size. That is, the SO(3) convolution (used in most layers of a typical Spherical CNN) has complexity $O(B^3 \log B)$ which compares favorably to the naive $O(B^6)$ spatial implementation. However, if we use filters with a fixed (and usually small) size, the complexity of a naive spatial implementation reduces to $O(B^3)$, which is slightly better than the FFT-based implementation. Further-

more, because the Icosahedral CNN uses a fixed number of orientations per filter (i.e. 6), its computational complexity is even better: it is linear in the number of pixels of the grid, and so comparable to $O(B^2)$ for the SOFT grid.

The difference in complexity is clearly visible in Figures 6.4 and 6.5, below. On the horizonal axis, we show the grid resolution r for the icosahedral grid $\mathcal{H}_r$ (for the spherical CNN, we a SOFT grid with roughly the same number of spatial points). On the vertical axis, we show the amount of wallclock time (averaged over 100 runs) and memory required to run an SO(3) convolution (S2CNN) or a regular-to-regular gauge equivariant convolution (IcoNet) at that resolution. Note that since the number of grid pionts is exponential in r, and we use a logarithmic vertical axis, the figures can be considered log-log plots. Both plots were generated by running a single regular to regular convolution layer at the corresponding resolution r with 12 input and output channels. For a fair comparison with IcoCNNs we chose filter grid parameters so3_near_identity_grid(n_alpha=6, max_beta=np.pi/16, n_beta=1, max_gamma=2*np.pi, n_gamma=6) for the spherical convolution layer. To guarantee a full GPU utilization, results were measured on an as large as possible batch size per datapoint and subsequently normalized by that batch size.

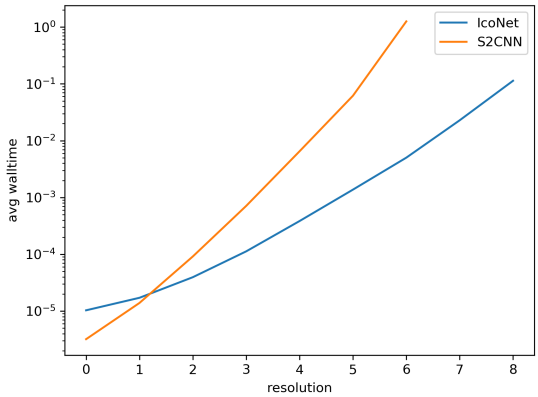


Figure 6.4: *Comparison of computational cost (in wallclock time) of Icosahedral CNNs (IcoNet) and Spherical CNNs (S2CNN, Cohen et al., 2018c), at increasing grid resolution r.*

As can be seen in Figure 6.4, the computational cost of running the S2CNN

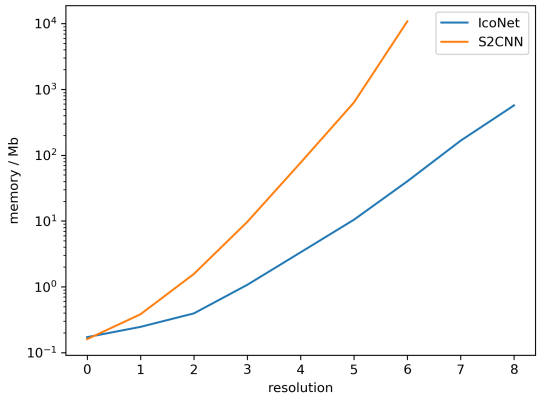


Figure 6.5: *Comparison of memory usage of Icosahedral CNNs (IcoNet) and Spherical CNNs (S2CNN, Cohen et al., 2018c), at increasing grid resolution r.*

dramatically exceeds the cost of running the IcoCNN, particularly at higher resolutions. We did not run the spherical CNN beyond resolution $r = 6$, because the network would not fit in GPU memory even when using batch size 1.

As shown in Figure 6.5, the Spherical CNN at resolution $r = 6$ uses about 10GB of memory, whereas the Icosahedral CNN uses only about 1GB. Since we used the maximum batch size with subsequent normalization for each resolution the reported memory consumption mainly reflects the memory cost of the feature maps, not the constant memory cost of the filter banks.

Aside from the theoretical asymptotic complexity, the actual computational cost depends on important implementation details. Because the heavy lifting of the Icosahedral CNN is all done by a single conv2d call, our method benefits from the extensive optimization of, and hardware support for this operation. By contrast, the generalized FFT used in the original Spherical CNN uses a conventional FFT, as well as matrix multiplcations with spectral matrices of size $1, 3, 5, 7, \ldots, 2L + 1$ (the SO(3) spectrum is matrix-valued, instead of the scalar valued spectrum for commutative groups). Implementing this in a way that is fast in practice is more challenging.

A final note on scalability. For some problems, such as the analysis of high

resolution global climate or weather data, it is unlikely that even a single feature map will fit in memory at once on current or near-term future hardware. Hence, it may be useful to split large feature maps into local charts, and process each one on a separate compute node. For the final results to be globally consistent (so that each compute node makes equivalent predictions for points in the overlap of charts), gauge equivariance is indispensable.

Appendix C: Details on G-Padding

In a conventional CNN, one has to pad the input feature map in order to compute an output of the same size. Although the icosahedron itself does not have a boundary, the charts do, and hence require padding before convolution. However, in order to faithfully simulate convolution on the icosahedron via convolution in the charts, the padding values need to be copied from another chart instead of e.g. padding by zeros. In doing so, a gauge transformation may be required.

To see why, note that the conv2d operation, which we use to perform the convolution in the charts, implicitly assumes that the signal is expressed relative to a fixed global gauge in the plane, namely the frame defined by the x and y axes. This is because the filters are shifted along the x and y directions by conv2d, and as they are shifted they are not rotated. So the meaning of "right" and "up" doesn't change as we move over the plane; the local gauge at each position is aligned with the global x and y axes.

Hence, it is this global gauge that we must use inside the charts shown in Figure 4 (right) of the main paper. It is important to note that although all frames have the same numerical expression $e_1 = (1, 0), e_2 = (0, 1)$ relative to the x and y axes, the corresponding frames on the icosahedron itself are different for different charts. Since feature vectors are represented by coefficients that have a meaning only relative to a frame, they have a different numerical expression in different charts in which they are contained. The numerical representations of a feature vector in two charts are related by a gauge transformation.

To better understand the gauge transformation intuitively, consider a pixel p on a colored edge in Fig. 4 of the main paper, that lies in multiple charts. Now consider a vector attached at this pixel (i.e. in T_pM), pointing along the colored edge. Since the colored edge may have different orientations when pictured in different charts, the vector (which is aligned with this edge) will also point in different directions in the charts, when the charts are placed on

the plane together as in Figure 4. More specifically, for the choice of charts we have made, the difference in orientation is always one "click", i.e. a rotation by plus or minus $2\pi/6$. This is the gauge transformation $g_{ij}(p)$, which describes the transformation at p when switching between chart i and j.

The transformation $g_{ij}(p)$ acts on the feature vector at p via the matrix $\rho(g_{ij}(p))$, where ρ is the representation of $G = C_6$ associated with the feature space under consideration. In this work we only consider two kinds of representations: scalar features with $\rho(g) = 1$, and regular features with ρ equal to the regular representation:

$$\rho(2\pi/6) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{6.3}$$

That is, a cyclic permutation of 6 elements. Since $2\pi/6$ is a generator of C_6, the value of ρ at the other group elements is determined by this matrix: $\rho(k{\cdot}2\pi/6) = \rho(2\pi/6)^k$. If the feature vector consists of multiple scalar or regular features, we would have a block-diagonal matrix $\rho(g_{ij}(p))$.

We implement G-padding by indexing operations on the feature maps. For each position p to be padded, we pre-compute $g_{ij}(p)$, which can be $+1 \cdot 2\pi/6$ or 0 or $-1 \cdot 2\pi/6$. We use these to precompute four indexing operations (for the top, bottom, left and right side of the charts).

Part II

General Theory of Equivariant CNNs

Introduction to Part II

> *"The aim of theory really is, to a great extent, that of systematically organizing past experience in such a way that the next generation, our students and their students and so on, will be able to absorb the essential aspects in as painless a way as possible, and this is the only way in which you can go on cumulatively building up any kind of scientific activity without eventually coming to a dead end."*
>
> – Michael Atiyah, "How research is carried out"

The purpose of this second part of the thesis is to bring together all of the methods presented in Part I into a single coherent mathematical framework, describing each one as a variation on a theme. There is quite a variety of symmetries – global, gauge, and even local symmetry groupoids – but we will find that a generic description can still be achieved. Specifically, each case involves a group acting on a principal bundle via principal bundle automorphisms.

The motivation for developing such a general theory is nicely captured by the above quote by Atiyah: By working at an abstract level, we can squeeze out redundancy and arrive at a compact description that can more easily be absorbed, and from which special cases can be derived with ease. Fundamental truths can often be seen more easily when the details that are not common between different instances are rendered invisible through abstraction.

A glance at the pages that follow may cause the reader to believe that learning this material will not in fact be "painless", Atiyah's claim notwithstanding. There is indeed a substantial amount of mathematics involved, even beyond the material we assume to be known (Esteves, 2020). However, essentially all of Chapter 7 is devoted to the introduction — as straightforwardly as possible — of standard concepts from the theory of fiber bundles, and can thus not be considered part of the theory of G-CNNs proper. From a minimum description length perspective, one might say that we pay a cost of $L(\text{bundles})$ in order to get a sizeable reduction $L(\text{G-CNNs}|\text{bundles}) \ll L(\text{G-CNNs})$. Furthermore, given that the theory of fiber bundles is a generally useful thing to have in one's mental toolbox, the cost $L(\text{bundles})$ is amortized. Thus, the claim is that provided we take the theory of fiber bundles as background knowledge, the theory of G-CNNs can be learned quite efficiently. Indeed, from this vantage point, all of the concepts are very natural and familiar, even obvious (though figuring out that G-CNNs can be described in this way was not).

In addition to providing a compact description of existing methods, the general theory can also be used to guide future research. Certainly there are many special cases (spaces, symmetries) that have not been explored and applied yet. For these cases, the theory provides a template that can be used to derive convolutional networks that respect the relevant symmetries. Needless to say, this could be far from trivial even with the help of a theory, and much engineering as well as theoretical work remains to be done.

By employing well-known mathematical concepts that have been widely used already, the theory may also suggest directions for future research by analogy to developments in physics and mathematics. Last but not least, a mathematical theory allows us to ask questions and prove theorems. The main question we are concerned with here is *universality*: that any equivariant linear map can be written as a convolution or integral transform, with a kernel satisfying certain necessary and sufficient conditions. When this can be shown, we can be sure that we are not unnecessarily restricting the linear layers of our network beyond what is necessary to be equivariant.

Overview of Part II

We begin in Chapter 7 with a self-contained[2] introduction of the theory of fiber bundles, focussing on those concepts that are relevant to the theory of equivariant CNNs. As always we have made an attempt to make the material accessible, but it cannot be denied that a greater degree of mathematical maturity is required here than in previous chapters.

Why do we need fiber bundles to describe G-CNNs? A full appreciation can only be had by reading the following chapters, but we can already provide some motivation. After writing Steerable CNNs (Chapter 3) it became clear that the right way to conceive of CNN feature spaces is as spaces of fields (the induced representation used in that paper describes the action of a symmetry group on a field). A field is somewhat similar to a function, in that it maps points in one space (the base space, e.g. sphere) to points in another space (the fiber, e.g. tangent plane). However fields are different because the output *space* depends on the input; a point x is mapped to a vector in the fiber F_x over x (e.g. the tangent plane at x). Hence, in cases where it is not possible to align all the fibers via a smooth map (as is the case for the tangent bundle of the sphere), it is not possible to describe a smooth field as a smooth function. We believe that

[2] A passing familiarity with Riemannian geometry will be helpful, though one should be able to get the gist having only so much as a mental picture of the concept of a manifold and a field.

preserving smoothness or at least continuity (similar patterns are *represented* similarly) is of utmost important for proper generalization. Hence bundles are indispensable.

Then in Chapter 8 we present a very general mathematical model for equivariant CNNs, describing symmetries as a group acting by automorphisms on a principal bundle, and feature spaces as spaces of fields (sections of an associated vector bundle). Network layers are described as equivariant linear maps between feature spaces. We give a wide variety of examples of different kinds of symmetries, representation spaces and network layers that are relevant in different applications, and study their general properties.

In Chapter 9 we study the special case of homogeneous G-CNNs, in which we have a group of symmetries H acting on a homogeneous base space $B \simeq H/G$, and representation spaces are defined as fields over B. We show how linear maps between such spaces (the network layers to be learned) can be written as integral transforms with a two-argument kernel (which is shown to be a field as well). The requirement of equivariance to symmetry transformations leads to invariance constraints on the kernel, and we show that the space of invariant two-argument kernels is isomorphic to the space of one-argument convolution kernels. This leads to the Convolution is All You Need theorem, which shows that any equivariant linear map between homogeneous representation spaces is a generalized convolution.

In Chapter 10 we repeat this analysis for general manifolds and gauge symmetries. We again derive an invariance constraint on the kernel, but show that in this case kernels that satisfy this constraint are trivial because the group of gauge transformations is very large. This motivates the introduction of a *connection*, using wich we can (without loss of generality) define a non-trivial kind of equivariant linear map. The resulting maps are gauge equivariant, but because we do not have a transitive group of symmetries, there is no spatial weight sharing. Spatial weight sharing, as used in many applications, is motivated by introducing a local symmetry groupoid whose elements are radial isometries.

7

Fiber Bundles

In this chapter we give an introduction to the theory of fiber bundles, focusing on those aspects relevant to the theory of G-CNNs presented in subsequent chapters. The material and presentation style is inspired by (Grothendieck, 1955; Rudolph and Schmidt, 2012; Hamilton, 2017).

7.1 Bundles

The fearsome theory of bundles has humble beginnings. A bundle is just a map

$$E \xrightarrow{\pi} B \tag{7.1}$$

between two spaces E and B, called the total space and base space, respectively. The map π is called the *projection* of the bundle.

Although it is possible to develop much of the theory for E and B topological manifolds and π a continuous map (as well as more general settings), we will take E and B to be smooth manifolds and π a smooth map[1]. Note that this includes finite and countable discrete spaces as a (trivial) special case.

Given a bundle $E \xrightarrow{\pi} B$, we define the *fiber* $E_x \subset E$ at $x \in B$ as the set of points that project to x, i.e.

$$E_x = \pi^{-1}(x). \tag{7.2}$$

We will always assume π to be surjective, so that $\pi^{-1}(x)$ is never empty.

[1] In the sequel, map shall mean smooth map and space / manifold shall mean smooth manifold. Group shall mean Lie group and group homomorphism shall mean Lie group homomorphism.

The most interesting and mathematically tractable bundles arise when the total space locally looks like a product $U \times F$ of a piece $U \subset B$ of the base and a canonical fiber F, with F carrying some structure (e.g. that of a smooth manifold, Lie group, or vector space). One can then think of E as a bundle of equivalent fibers $E_x \simeq F$, parameterized by points x in the base space B, and glued together according to the topology of E. We will make this idea precise in Section 7.1.4, but first we will look at a number of concepts that can be studied in a more general context.

7.1.1 Bundle Maps

When studying any category of mathematical structures, such as bundles, it is vital to be able to relate different instances of that structure. This ability is afforded by an appropriately defined notion of morphism.

Definition 7.1.1 (Bundle Morphism)**.** *A* bundle morphism *(or bundle map) between two bundles $\pi : E \to B$ and $\pi' : E' \to B'$ is a pair of maps $\alpha : E \to E'$ and $\beta : B \to B'$ that make the following diagram commute:*

$$\begin{array}{ccc} E & \xrightarrow{\alpha} & E' \\ {\scriptstyle\pi}\downarrow & & \downarrow{\scriptstyle\pi'} \\ B & \xrightarrow{\beta} & B' \end{array} \tag{7.3}$$

In other words, a bundle morphism is a map $\alpha : E \to E'$ that maps fibers to fibers. Given such an α, the map β is determined as the map that takes $x \in B$ to $\pi'(\alpha(p)))$ for an arbitrary $p \in E_x$. We can thus refer to α alone as a bundle morphism. The converse is not generally true (β does not determine α), except in so-called *natural* bundles (Kolář et al., 1993), where each β of a particular kind induces a unique α (e.g., a diffeomorphism of a smooth manifold B induces an (auto)morphism α of the frame bundle E over B – see Theorem 7.2.1).

If we restrict the bundle morphism $\alpha : E \to E'$ to the fiber at $x \in B$, we get a map between fibers that will be denoted $\alpha_x : E_x \to E'_{\beta(x)}$.

Definition 7.1.2 (Bundle B-Morphism)**.** *A B-morphism between two bundles over the same base space B is a bundle morphism that projects to the identity map on B (that is, $\beta = \mathrm{id}_B$):*

$$\begin{array}{ccc} E & \xrightarrow{\alpha} & E' \\ & {\scriptstyle\pi}\searrow \quad \swarrow{\scriptstyle\pi'} & \\ & B & \end{array} \tag{7.4}$$

Thus, a B-morphism maps the fiber E_x to the fiber E'_x and we have $\alpha_x : E_x \to E'_x$. B-morphisms are also called vertical morphisms or base-fixing morphisms.

A *bundle isomorphism* (resp. B-isomorphism) is just a bundle morphism (resp. B-morphism) where α and β are isomorphisms. In the smooth setting, isomorphisms are also called diffeomorphisms, defined as smooth invertible maps with smooth inverse. If there exists a bundle isomorphism (resp. B-isomorphism) between bundles E and E', we say they are isomorphic and write $E \simeq E'$ (resp. $E \simeq_B E'$).

Definition 7.1.3 (Composition of Bundle Morphisms)**.** *Let E, E', E'' be bundles with base space B, B', B'', respectively. Let (α, β) be a bundle morphism with signature $\alpha : E \to E'$ and $\beta : B \to B'$. Similarly, let (α', β') be a bundle morphism with signature $\alpha' : E' \to E''$ and $\beta' : B' \to B''$. Then the composition of bundle maps is defined as $\alpha' \circ \alpha : E \to E''$ and $\beta' \circ \beta : B \to B''$.*

Proposition 1. *The composition of bundle morphisms is a bundle morphism. Likewise, the composition of bundle B-morphisms is a bundle B-morphism. If both morphisms (resp. B-morphisms) are bundle isomorphisms, their composition is again an isomorphism (resp. B-isomorphism).*

Proof. Left as an exercise. ∎

Proposition 1, together with a few more basic facts (such as the existence of an identity morphism for any bundle) tells us that smooth bundles form a *category* with bundles as objects and bundle morphisms as morphisms. We will not make explicit use of category theory here, but we mention it for the interested reader and note that Proposition 1 will be used often.

7.1.2 Sections

Definition 7.1.4 (Section)**.** *A (global)* section *of a bundle is a map $\sigma : B \to E$ such that $\pi \circ \sigma = \mathrm{id}_B$. That is, the following diagram commutes:*

$$\begin{array}{ccc} B & \xrightarrow{\sigma} & E \\ & \searrow^{\mathrm{id}_B} & \downarrow \pi \\ & & B \end{array} \tag{7.5}$$

From $\pi(\sigma(x)) = x$ we can infer that $\sigma(x)$ is in the preimage of x under π, that is, $\sigma(x)$ is an element of the fiber E_x over x. Thus, a section chooses for each

$x \in B$ a point in the fiber E_x over x. We will denote the space of sections of E by $\Gamma(E)$. It is important to note that smooth global sections may not exist.

A section is similar to a function from the base to the fiber, but different because the "codomain" E_x depends on the input $x \in B$. If we can simultaneously align all the fibers E_x with some canonical fiber F in a smooth way, then we can represent a section by a function $f : B \to F$. In Section 7.1.4 we will define *local triviality*, which guarantees that locally we can always do this, and in Section 7.1.7 we show how to express a local piece of a section of a locally trivial bundle as a function.

7.1.3 Trivial Bundles

A bundle isomorphism $\alpha : E \to E'$ tells us that (and in which way) E and E' are essentially the same as bundles. This is particularly useful when we can show that the bundle under consideration is equivalent to one that we can understand easily, for then we can study the easy case and transport the results back to the bundle of interest via the isomorphism. With this in mind, we make the following definition:

Definition 7.1.5 (Product bundle)**.** *The* product bundle *with base space B and fiber F is the bundle with total space $E = B \times F$ and projection* $\mathrm{proj}_1(b, f) = b$.

In a product bundle, all fibers are isomorphic to the canonical fiber F:

$$E_x = \mathrm{proj}_1^{-1}(b) = \{(b, f) \mid f \in F\} = \{b\} \times F \simeq F.$$

Product bundles are easy to understand and many concepts defined in the theory of fiber bundles take a simple form for product bundles. This includes in particular the notion of *bundle map* and *section* (See 7.1.5 and 7.1.7).

Definition 7.1.6 (Trivial bundle)**.** *A bundle E is called* trivial *if there exists a bundle isomorphism between E and the product bundle* $\mathrm{proj}_1 : B \times F \to B$. *A bundle isomorphism $\alpha : E \to B \times F$ is called a (global) trivialization of E.*

In particular, the product bundle is trivially trivial and so we may also refer to it as the trivial bundle $B \times F$. If a bundle E is trivial, all of its fibers are isomorphic. We know this because as mentioned before, a bundle isomorphism maps fibers onto fibers in an invertible way, and the fibers of a product bundle $B \times F$ are all isomorphic to F. Specifically, the trivializing isomorphism $\alpha : E \to B \times F$ restricts to an isomorphism of fibers $\alpha_x : E_x \to F$.

Beyond telling us that the fibers are all the same, the existance of a trivialization tells us that the bundle has the same topological and manifold structure as the product $B \times F$. Note that this is a stronger statement than saying the bundle has base B and fiber F, because two bundles can have the same base space and fiber and yet be non-isomorphic (e.g. a cylinder and Mobius strip can both be viewed as bundles with the circle S^1 as base and a line segment as fiber).

7.1.4 Locally Trivial Bundles

Many bundles we are interested in are not trivial, in which case we cannot use the abovementioned strategy of transporting results about product bundles to the bundle of interest. But all hope is not lost: we will henceforth assume (since this is true for all bundles we care about) that the bundle is *locally trivial*.

Consider a bundle $\pi : E \to B$ and an open set $U \subset B$. We have two ways to form a bundle over U. Firstly, we can make $E_U = \pi^{-1}(U)$ into a bundle over base space U with projection $\pi|_U : E_U \to U$. On the other hand, given a canonical fiber F, we can form the trivial bundle $U \times F$.

Definition 7.1.7 (Locally trivial fiber bundle)**.** *A bundle $\pi : E \to B$ is called a locally trivial fiber bundle with canonical fiber F if each $x \in B$ has an open neighbourhood $U \subset B$ such that the bundle $E_U = \pi^{-1}(U)$ over U and the product bundle $U \times F$ (for some smooth manifold F) are U-isomorphic. That is, the following diagram commutes:*

$$\begin{array}{ccc} E_U & \xrightarrow{\varphi} & U \times F \\ & \searrow^{\pi} & \downarrow{\scriptstyle \mathrm{proj}_1} \\ & & U \end{array}$$

The U-isomorphism φ is called a trivialization of E_U, a local trivialization of E over U, or a bundle chart of E.

We will henceforth drop the "locally trivial" qualifier and speak only of fiber bundles or even just bundles, but they will be assumed locally trivial.

Many texts begin by defining fiber bundles as tuples (E, B, F, π) such that for any $x \in B$ there exists a neighbourhood U of x and a diffeomorphism $\varphi : \pi^{-1}(U) \to U \times F$ such that $\mathrm{proj}_1 \circ \varphi = \pi$. We find it psychologically helpful to emphasize the fact that local trivializations are *isomorphisms* between a piece of the bundle E and the trivial bundle $U \times F$, because this makes it immediately obvious that one can mentally identify the two; $E_U \simeq_U U \times F$. An immediate

consequence is that since all the fibers of $U \times F$ are isomorphic to the canonical fiber F, the same is true for the fibers of E. Indeed, the local trivialization φ, being a U-isomorphism, restricts to a fiber isomorphism $\varphi_x : E_x \to F$.

Remark. *In definition 7.1.7 it is sufficient to ask for a U-isomorphism as opposed to a general bundle isomorphism. The reason is that if E_U is isomorphic to $V \times F$ by a bundle isomorphism (α, β), then $\beta(U) = V$ and E_U is also U-isomorphic to $U \times F$.*

Thus, we can proceed with our strategy of studying trivial bundles and using the knowledge we gain to better understand any locally trivial fiber bundle. However, because locally trivial bundles may not be globally trivial, we need to also understand how the trivial pieces are put together.

Definition 7.1.8 (Bundle atlas)**.** *Let $E \to B$ be a bundle. A collection of neighbourhoods U_i covering B, together with local trivializations $\varphi_i : \pi^{-1}(U_i) \to U_i \times F$ is called a bundle atlas of E.*

A bundle is locally trivial if and only if it has an atlas of charts. From hereon we assume that a maximal atlas is given, and charts are always taken from this atlas (see Husemöller (1994), Ch. 5.2, for details).

7.1.5 Local Bundle Morphisms

We have already given a global description of bundle morphisms as maps α, β satisfying $\beta \circ \pi = \pi' \circ \alpha$ (definition 7.1.1). We have also defined local trivializations φ as bundle maps (more specifically, U-isomorphisms) between a local piece E_U of E and a trivial bundle $U \times F$. Now we will consider bundle morphisms between two product bundles, which take a simple form, and show how using a local trivialization we can reduce any bundle morphism to this form, at least locally.

Proposition 2. *Let $\mathrm{proj}_1 : U \times F \to U$ and $\mathrm{proj}_1 : U' \times F' \to U'$ be trivial bundles. A bundle morphism (α, β) from the former to the latter takes the following form:*

$$\alpha(u, f) = (\beta(u), \nu(u, f)) \tag{7.6}$$

For some map $\nu : U \times F \to F$. In particular, a U-morphism α between bundles $U \times F$ and $U \times F'$ over the same base space U takes the form

$$\alpha(u, f) = (u, \nu(u, f)) \tag{7.7}$$

Proof. We know that $\alpha(u,f)$ is an element of $U'\times F'$. To determine the U' factor, we project on the first factor and find, using the defining property of bundle morphisms that $\text{proj}_1 \circ \alpha(u,f) = \beta \circ \text{proj}_1(u,f) = \beta(u)$. For the second factor, we simply make a definition: $\nu = \text{proj}_2 \circ \alpha$. ∎

Now that we understand bundle maps for product bundles, we can use this to understand general bundle maps using local trivializations.

Proposition 3. *Let $\pi : E \to B$ and $\pi' : E' \to B'$ be locally trivial fiber bundles, and let $\alpha : E \to E'$, $\beta : B \to B'$ be a bundle map. Choose a local trivialization $\varphi_i : E_{U_i} \to U_i \times F$ of E over $U_i \subset B$ and a local trivialization $\varphi_j : E'_{V_j} \to V_j \times F'$ of E' over $V_j \subset B'$, such that $U_i \cap \beta^{-1}(V_j) \neq \emptyset$. Then the following is a well defined bundle map between product bundles:*

$$\begin{aligned} \alpha_{ij} = \varphi_j \circ \alpha \circ \varphi_i^{-1} : U_{ij}^{\beta} \times F \to V_{ij}^{\beta} \times F' \\ \beta : U_{ij}^{\beta} \to V_{ij}^{\beta}. \end{aligned} \tag{7.8}$$

where $U_{ij}^{\beta} = U_i \cap \beta^{-1}(V_j)$ and $V_{ij}^{\beta} = \beta(U_i) \cap V_j$.

Hence, by Proposition 2, it takes the form

$$\alpha_{ij}(u,f) = \varphi_j \circ \alpha \circ \varphi_i^{-1}(u,f) = (\beta(u), \nu_{ij}^{\alpha}(u,f)) \tag{7.9}$$

Proof. Let us first verify that α_{ij} is well defined on the given domain and codomain. A point $(x,f) \in U_i \times F$ will get mapped to $p = \alpha(\varphi_i^{-1}(x,v))$, which may not be in $E_{V_j} = \pi^{-1}(V_j)$ (the domain of φ_j). But if $x \in \beta^{-1}(V_j)$, then p will be in E_{V_j}. So we restrict the map to $U_{ij}^{\beta} = \beta^{-1}(V_j) \cap U_i$. This domain is then mapped to $V_{ij}^{\beta} = \beta(U_{ij}^{\beta}) = \beta(U_i) \cap V_j$.

Since α_{ij} is the composition of a B'-isomorphism φ_j, a general bundle morphism α, and another B-isomorphism, we expect that the composition acts on the base space via β. Indeed this is the case:

$$\begin{aligned} \text{proj}_1 \circ \varphi_j \circ \alpha \circ \varphi_i^{-1}(x,v) &= \pi \circ \alpha \circ \varphi_i^{-1}(x,v) \\ &= \beta \circ \pi \circ \varphi_i^{-1}(x,v) \\ &= \beta \circ \text{proj}_1(x,v) \\ &= \beta(x). \end{aligned} \tag{7.10}$$

In this derivation we used that φ_j is a B'-morphism, the definition of bundle morphisms, and the fact that φ_i^{-1} is a B-morphism. For the second factor we define $\nu_{ij}^{\alpha} = \text{proj}_2 \circ \alpha_{ij}$. ∎

7.1.6 Transition Morphisms

Given an atlas, we can consider the *transition morphisms*

$$\varphi_{ij} = \varphi_i \circ \varphi_j^{-1} : U_{ij} \times F \to U_{ij} \times F \tag{7.11}$$

defined on overlaps $U_{ij} = U_i \cap U_j$. We can think of φ_{ij} as a reparameterization. That is, we can either parameterize points $p \in \pi^{-1}(U_{ij})$ as $p = \varphi_j^{-1}(u, f)$ using φ_j or we can use φ_i and write $p = \varphi_i^{-1}(\varphi_{ij}(u, f)) = \varphi_i^{-1}(u, f')$.

Being the composition of U_{ij}-isomorphisms, the map φ_{ij} is a U_{ij}-isomorphism of product bundles[2]. As such, it must take the following simple form

$$\varphi_{ij}(u, f) = (u, \nu_{ij}(u, f)). \tag{7.12}$$

This follows from the requirement $\text{proj}_1 \circ \varphi = \text{proj}_1$, i.e. Eq. 7.4 / Def. 7.1.7.

We can also think of this result as a special case of Proposition 3. That is, we can think of the transition morphism φ_{ij} as the local representation of the identity morphism $\alpha = \text{id}_E$.

Finally, we note that the transition functions satisfy $\varphi_{ii} = \text{id}$ and $\varphi_{ij}^{-1} = \varphi_{ji}$. On triple overlaps, they satisfy the so-called cocycle condition

$$\begin{aligned} \varphi_{ij} \circ \varphi_{jk} &= \varphi_{ik} \\ \varphi_{ii} &= \text{id}. \end{aligned} \tag{7.13}$$

It is known that maps φ_{ij} satisfying these conditions are equivalent data to the global description of a bundle as a projection $\pi : E \to B$ (Husemöller, 1994).

7.1.7 Local Sections

To better understand sections (7.1.2), we study them in the trivial case and then apply the result to general fiber bundles using a local trivialization.

Proposition 4. *Let* $\text{proj}_1 : B \times F \to B$ *be an trivial bundle, and* $\sigma : B \to B \times F$ *a section. Then* σ *takes the following form:*

$$\sigma(x, a) = (x, f(x)). \tag{7.14}$$

for some map $f : B \to F$.

[2]Indeed, the transition function φ_{ij} is an isomorphism from the trivial bundle $U_{ij} \times F$ to itself, also known as a bundle automorphism. And automorphism (isomorphism from an object to itself) is another word for symmetry. Thus, we reparameterize by applying a symmetry of $U_{ij} \times F$.

Proof. A section must satisfy $\pi \circ \sigma = \mathrm{id}_B$. Since $\pi = \mathrm{proj}_1$ in the present case, we have $\mathrm{proj}_1(\sigma(x)) = x$. For the second factor we define $f = \mathrm{proj}_2 \circ \sigma$. ∎

Next we show that any section can locally be represented in this way.

Proposition 5. *Let $\sigma : B \to E$ be a section of a fiber bundle $\pi : E \to B$ with fiber F. Using a local trivialization $\varphi_i : E_{U_i} \to U_i \times F$, we can write $\sigma|_{U_i} : U_i \to E_{U_i}$ as a section of the product bundle $\mathrm{proj}_1 : U_i \times F \to U_i$:*

$$\varphi_i \circ \sigma|_{U_i}(x) = (x, f_i(x)), \tag{7.15}$$

where f_i is a map from U_i to F. Conversely, given a map $f_i : U_i \to F$, we can construct a local section $\sigma_{U_i}(x) = \varphi_i^{-1}(x, f_i(x))$ of E_{U_i}. These processes are inverses.

Proof. Since φ_i is a U_i-isomorphism, we have that $\mathrm{proj}_1 \circ \varphi_i = \pi$. It follows that $\mathrm{proj}_1 \circ \varphi_i \circ \sigma = \pi \circ \sigma = \mathrm{id}_B$, because σ is a section of $\pi : E \to B$. So $\varphi_i \circ \sigma$ is a section of the trivial bundle $\mathrm{proj}_1 : E_{U_i} \to U_i$.

We verify that

$$\pi(\sigma_{U_i}(x)) = \pi(\varphi_i^{-1}(x, f_i(x))) = \mathrm{proj}_1(x, f_i(x)) = x, \tag{7.16}$$

so σ_{U_i} is a local section of $\pi : E_{U_i} \to U_i$. That these processes are inverses follows immediately from the fact that φ_i and φ_i^{-1} are inverses. ∎

By defining functions f_i so that the corresponding local sections σ_{U_i} agree on intersections, one can specify a global section.

7.2 Foobar Bundles

In this section we will discuss various special kinds of fiber bundles that play a role in the theory of equivariant convolutional networks. The general mantra will be as follows: a Foobar bundle is a bundle whose fibers have Foobar structure, and Foobar bundle morphisms should restrict to Foobar morphisms on the fibers. Here Foobar could stand for vector space, G-space, etc., with the respective morphisms being linear maps, equivariant maps, etc. Local trivializations, being bundle isomorphisms, should be Foobar morphisms and hence respect Foobar structure as well.

7.2.1 Vector bundles

A vector bundle is a bundle of vector spaces. That is, we have a bundle $E \xrightarrow{\pi} B$, such that the fibers $E_x = \pi^{-1}(x)$ have a vector space structure. We will always assume these vector spaces to be real and finite dimensional.

A morphism of vector bundles is a bundle morphism $\alpha : E \to E'$ such that the fiber map $\alpha_x : E_x \to E'_{\beta(x)}$ is a linear map, for all $x \in B$.

A section of a vector bundle is called a field in physics, though beware that the word "vector field" is usually reserved for a section of a very particular vector bundle, namely the tangent bundle of B. A further warning is that the word field is used in mathematics for an entirely different kind of thing (e.g. the field of real or complex numbers). In the theory of equivariant networks, we will model feature maps as sections of vector bundles (i.e. fields).

Local vector bundle maps

Definition 7.2.1 (Product vector bundle)**.** *A product vector bundle is a bundle* $\mathrm{proj}_1 : B \times V \to B$*, where V is a vector space. Addition and scalar multiplication in the fibers is defined as $a(x,v) + b(x,w) = (x, av + bw)$, for $a, b \in \mathbb{R}$ and $v, w \in V$.*

Proposition 6. *A vector bundle morphism α between trivial bundles $B \times V$ and $B' \times V'$ takes the following form:*

$$\alpha(x,v) = (\beta(x), \nu(x,v)), \tag{7.17}$$

where $\nu(x,v)$ is linear in v.

Proof. Since α is a morphism of product bundles, by Proposition 2 it takes the given form. The map $\nu : B \times V \to V'$ is linear in v because $\alpha_x(v) = \nu(x,v)$, and by definition of vector bundle morphism, α_x is linear. ∎

By using local trivializations as in Proposition 3, it follows immediately that locally, a vector bundle morphism between non-trivial bundles also takes the form given above.

7.2.2 G-Bundles

Definition 7.2.2 (G-Bundle)**.** *Let G be a finite dimensional Lie group. A G-bundle is a bundle $E \xrightarrow{\pi} B$ together with a smooth right G-action $p \mapsto pg$ on E, such that the projection is G-invariant:*

$$\pi(pg) = \pi(p). \tag{7.18}$$

The group G is called the structure group *of the bundle.*

Thus, the fibers $E_x \subset E$ are mapped onto themselves by G, making them into right G-spaces. In other words, a *G-bundle* is a bundle of G-spaces.

Definition 7.2.3 (G-Bundle Morphism)**.** *Let $\pi_1 : E_1 \to B_1$ be a G_1-bundle and $\pi_2 : E_2 \to B_2$ be a G_2 bundle. Let $\theta : G_1 \to G_2$ be a group homomorphism. A G-bundle morphism over θ is a bundle morphism $\alpha : E_1 \to E_2$ that satisfies:*

$$\alpha(pg) = \alpha(p)\theta(g), \tag{7.19}$$

for all $p \in E_1$ and $g \in G_1$.

We will refer to a pair (α, θ) as a G-bundle morphism or to α as a G-bundle θ-morphism. If $G = G_1 = G_2$ and $\theta = \mathrm{id}_G$, then α is called a G-bundle G-morphism. A morphism for which $\beta = \mathrm{id}_B$ and $\theta = \mathrm{id}_G$ will be called a (B, G)-morphism. However, if we refer to α as a G-bundle morphism (resp. G-bundle B-morphism), it is assumed that this is a G-morphism (resp. (G, B)-morphism), leaving $\theta = \mathrm{id}_G$ implicit.

The definition of G-bundle and G-bundle morphism over θ is summarized by the diagram:

$$\begin{array}{ccc} E & \xrightarrow{\alpha} & E' \\ {\scriptstyle R_g}\downarrow & & \downarrow{\scriptstyle R_{\theta(g)}} \\ E & \xrightarrow{\alpha} & E' \\ {\scriptstyle \pi}\downarrow & & \downarrow{\scriptstyle \pi'} \\ B & \xrightarrow{\beta} & B' \end{array} \qquad \pi \circlearrowright \quad \pi' \circlearrowright \tag{7.20}$$

where $R_g p = pg$ denotes the right action of G_1 resp. G_2 on E_1 resp. E_2. We note that the composition of two G-bundle morphisms (α_1, θ_1) and (α_2, θ_2) is given by $(\alpha_1 \circ \alpha_2, \theta_1 \circ \theta_2)$ and is again a G-bundle morphism.

Local G-bundle maps

Definition 7.2.4 (Product G-Bundle)**.** *A product G-bundle is a bundle* $\mathrm{proj}_1 : B \times F \to B$*, where F is a right G-space. The right action on $B \times F$ is defined in terms of the right action of G on F as follows: $(u, f)g = (u, fg)$ for $u \in B$, $f \in F$, $g \in G$.*

We verify that $\mathrm{proj}_1((u, f)g) = \mathrm{proj}_1((u, fg)) = u = \mathrm{proj}_1((u, f))$, so this is indeed a G-Bundle.

Proposition 7 (Product G-morphism)**.** *Let $E = B \times F$ be a product G-bundle and $E' = B' \times F'$ a product G'-bundle. A G-bundle morphism $\alpha : E \to E'$ over $\theta : G \to G'$ takes the following form:*

$$\alpha(x, f) = (\beta(x), \nu(x, f)), \tag{7.21}$$

where $\nu : B \times F \to F'$ is a G-equivariant map, i.e. $\nu(x, fg) = \nu(x, f)\theta(g)$.

Proof. Since α is a morphism of product bundles, by Proposition 2 it takes the given form. The map $\nu : B \times F \to F'$ is G-equivariant because by definition of G-bundle morphism, α is G-equivariant, i.e. $\alpha(x, fg) = \alpha(x, f)\theta(g)$, which implies that $\nu(x, fg) = \nu(x, f)\theta(g)$. ∎

Local trivializations for a G-bundle are defined as G-bundle B-isomorphisms over $\theta = \mathrm{id}_G$, and hence are G-equivariant by definition of G-bundle morphism. Using local trivializations as in Proposition 3, we may infer that locally, any G-bundle morphism takes the form given in Proposition 7.

7.2.3 Principal G-Bundles

Definition 7.2.5 (Principal G-bundle)**.** *A* principal G-bundle *is a G-bundle P where the action of G on fibers is transitive and fixed-point free.*

Transitive means that for any $p, p' \in P_x$, there is a $g \in G$ so that $pg = p'$. Free means that if $pg = p$, then g is the identity. In other words, for any $p, p' \in P_x$, there is a *unique* $g \in G$ so that $pg = p'$. It follows that each fiber of P is isomorphic to G as a G-space. Thus, each fiber P_x looks like G, except that it lacks a preferred identity element.

A principal G-bundle morphism is just a G-bundle morphism between two G-bundles that happen to be principal. By equivariance, and the fact that each fiber of a principal bundle has only one orbit, the restriction of a principal G-bundle G-morphism α to a single fiber $\alpha_x : P_x \to P'_{\beta(x)}$ is an isomorphism of G-spaces. Hence, a principal bundle morphism is an isomorphism whenever β is an isomorphism.

Local principal bundle maps

Definition 7.2.6 (Product principal G-bundle)**.** *A product principal G-bundle is a bundle* $\mathrm{proj}_1 : B \times G \to B$*, where B is a manifold and G a Lie group. The right action of G on $B \times G$ is defined as $(u, g)g' = (u, gg')$.*

Proposition 8. *A principal G-bundle morphism α between product principal bundles* $\text{proj}_1 : B \times G$ *and* $\text{proj}_1 : B' \times G'$ *over* $\theta : G \to G'$ *takes the following form:*

$$\alpha(x, g) = (\beta(x), g^\alpha(x)\theta(g)), \tag{7.22}$$

for some map $g^\alpha : B \to G'$*. Conversely, a pair of maps* $\beta : B \to B'$*,* $g^\alpha : B \to G'$ *define a principal G-bundle morphism α between these product bundles.*

Proof. We know from Proposition 7 that since α is a G-bundle θ-morphism, it can be written as $\alpha(x, g) = (\beta(x), \nu(x, g))$, where $\nu(x, -) : G \to G'$ is G-equivariant: $\nu(x, gg') = \nu(x, g)\theta(g')$ for $x \in B$, $g, g' \in G$. Any G-equivariant map is determined on each orbit by its value on one orbit representative. But since $B \times G$ is a principal G-bundle, the fiber G has only one G-orbit (the action is transitive). So let us define $g^\alpha(x) = \nu(x, e)$, where we use the identity $e \in G$ as the orbit representative, and $g^\alpha : B \to G'$ is a function. Then ν is determined as

$$\nu(x, g) = \nu(x, eg) = \nu(x, e)\theta(g) = g^\alpha(x)\theta(g). \tag{7.23}$$

■

In words, α acts as a smooth map β on the base, and via a position-dependent left G transformation $g^\alpha(x)$ on the fiber. Moreover, a (B, G)-morphism only involves a position-dependent left G action, i.e. $(x, g) \mapsto (x, g^\alpha(x)g)$. Such maps are called local gauge transformations. We will study these in detail further on in this section.

Once again, we can use a result about product bundles and apply it to general bundles. Conjugating a principal bundle morphism α by local trivializations (which, being principal bundle B-automorphisms must be equivariant) as was done in Proposition 3, we find that α must locally take the form of a morphism between product principal bundles (Proposition 8).

Example: Frame Bundles

The prime example of a principal bundle is the bundle of frames of a vector bundle. Indeed, we can think of any principal bundle as a bundle frames for its associated vector bundles (defined in Section 7.2.4), so this example is very useful to build intuition.

Definition 7.2.7 (Frame Bundle of a Vector Bundle)**.** *Let $\pi_E : E \to B$ be a vector bundle with canonical fiber $V \simeq \mathbb{R}^n$. A frame of the fiber E_x is an ordered basis of*

E_x, or equivalently, a linear isomorphism (invertible linear map) $\psi_x : \mathbb{R}^n \to E_x$ (one obtains an ordered basis of E_x from ψ_x by mapping the standard basis of $\mathbb{R}^n$ to E_x).

Denote the space of all frames ψ_x of E_x by $F_{\mathrm{GL}}(E_x)$. The group $\mathrm{GL}(n, \mathbb{R})$ acts on $F_{\mathrm{GL}}(E_x)$ on the right by composition: $\psi_x \mapsto \psi_x \circ g$ for $\psi_x \in F_{\mathrm{GL}}(E_x)$ and $g \in \mathrm{GL}(n, \mathbb{R})$. Any two frames are related by a unique $g \in \mathrm{GL}(n, \mathbb{R})$, so this action is free and transitive. In other words, $F_{\mathrm{GL}}(E_x)$ is a principal $\mathrm{GL}(n, \mathbb{R})$-space.

The bundle of general linear frames $F_{\mathrm{GL}}(E)$ of E (also known as the frame bundle of E) is defined as follows. It has as total space the disjoint union of $F_{\mathrm{GL}}(E_x)$:

$$F_{\mathrm{GL}}(E) = \bigsqcup_{x \in B} F_{\mathrm{GL}}(E_x). \tag{7.24}$$

Thus, a point of $F_{\mathrm{GL}}(E)$ is a pair (x, ψ_x) where $x \in B$ and ψ_x is a frame for E_x. The projection $\pi_F : F_{\mathrm{GL}}(E) \to B$ sends (x, ψ_x) to x. The group $\mathrm{GL}(n, \mathbb{R})$ acts on $F_{\mathrm{GL}}(E)$ on the right via $(x, \psi_x)g = (x, \psi_x \circ g)$, making $F_{\mathrm{GL}}(E)$ into a principal $\mathrm{GL}(n, \mathbb{R})$-bundle.

The principal bundle $F_{\mathrm{GL}}(E)$ naturally obtains a smooth structure from E.

An important special case is the frame bundle of a manifold:

Definition 7.2.8 (Frame bundle of a Manifold)**.** *Let B be a smooth manifold, and TB the tangent bundle of B. Then the frame bundle of B, is just the frame bundle of the vector bundle TB, denoted variously as $F_{\mathrm{GL}}(TB)$, $F_{\mathrm{GL}}(B)$ or $F(B)$ if the structure group is clear from context.*

If we have a bundle metric on E (e.g. a Riemannian metric on TB) then we may want to restrict ourselves to orthonormal frames. If in addition we have an orientation, then we can consider positively oriented orthogonal frames. Such frames are packaged together in the bundles $F_{\mathrm{O}}(E)$ and $F_{\mathrm{SO}}(E)$. We will assume our manifold is orientable and give the definition of $F_{\mathrm{SO}}(E)$, for this is what we will make use of to describe G-CNNs on orientable Riemannian manifolds.

Definition 7.2.9 (Bundle of Oriented Orthonormal Frames)**.** *Let $\pi_E : E \to B$ be a vector bundle with canonical fiber $V \simeq \mathbb{R}^n$ carrying a bundle metric as well as an orientation. A positively oriented orthonormal frame of E_x is an orientation preserving linear isometry $\psi_x : \mathbb{R}^n \to E_x$. Denote the space of all such frames of E_x by $F_{\mathrm{SO}}(E_x)$. The group $\mathrm{SO}(n)$ acts on $F_{\mathrm{SO}}(E_x)$ on the right by composition: $\psi_x \mapsto \psi_x \circ g$ for $\psi_x \in F_{\mathrm{SO}}(E_x)$ and $g \in \mathrm{SO}(n)$. Since any two positively oriented orthonormal frames are related by a unique $g \in \mathrm{SO}(n)$, this action is free and transitive.*

Then the bundle of positively oriented orthogonal frames $F_{\mathrm{SO}}(E)$ is defined analogously to $F_{\mathrm{GL}}(E)$ (Definition 7.2.7), replacing general frames $F_{\mathrm{GL}}(E_x)$ by oriented orthonormal ones $F_{\mathrm{SO}}(E_x)$. That is, $F_{\mathrm{SO}}(E) = \bigsqcup_{x \in B} F_{\mathrm{SO}}(E_x)$.

A choice of local trivialization $\varphi : F_{\mathrm{GL}}(U) \to U \times \mathrm{GL}(n, \mathbb{R})$ corresponds to a choice of frame ψ on $U \subset B$. Define ψ by $(x, \psi_x) = \varphi^{-1}(x, e)$ for $x \in U$ and $e \in \mathrm{GL}(n, \mathbb{R})$ the identity matrix. By definition of the frame bundle, ψ_x is a map $\psi_x : \mathbb{R}^d \to T_x U$.

Theorem 7.2.1. *Let B be a manifold and $F_{\mathrm{GL}}(TB)$ the tangent frame bundle. Furthermore, let $\beta : B \to B$ be a diffeomorphism. Then β induces a unique principal bundle automorphism $\alpha : F_{\mathrm{GL}}(TB) \to F_{\mathrm{GL}}(TB)$ that covers β (i.e. $\pi \circ \alpha = \beta \circ \pi$).*

Proof. Clearly, β induces a unique linear isomorphism $\beta_{*x} : T_x B \to T_{\beta(x)} B$ for each $x \in B$. Also, α is determined by its restrictions $\alpha_x : F(T_x B) \to F(T_{\beta(x)} B)$. The pushforward β_{*x} acts on a frame $\psi_x : \mathbb{R}^n \to T_x B$ by composition on the left, i.e. $\psi_x \mapsto \psi'_{\beta(x)} = \beta_{*x} \circ \psi_x$, which is a linear isomorphism $\psi'_{\beta(x)} : \mathbb{R}^n \to T_{\beta(x)}$, i.e. a frame at $\beta(x)$, also known as an element of the fiber $F_{\mathrm{GL}}(T_x B)$. We will use this to define α as follows:

$$\alpha_x(x, \psi_x) = (\beta(x), \beta_{*x} \circ \psi_x). \tag{7.25}$$

One checks that $\pi \circ \alpha = \beta \circ \pi$ and that α is right $\mathrm{GL}(n, \mathbb{R})$-equivariant by plugging in the definitions.

The local representation $g^\alpha(x)$ of α is the local representation of β_{*x},

$$g^\alpha_{ij}(x) = (\psi^j_{\beta(x)})^{-1} \circ \beta_{*x} \circ \psi^i_x. \tag{7.26}$$

■

In a way, the frame bundle of a vector bundle is the only example of a principal bundle one needs to understand. This is because any principal bundle is isomorphic to a subbundle of the frame bundle of certain vector bundles associated with the principal bundle (Weatherall, 2014, Section 4). This subbundle contains only those frames that are adapted to a certain structure (e.g. inner product & orthogonal frames) on the fibers of the vector bundle (i.e. it is a reduction of the structure group, see 7.2.3). This point of view is not universally shared in mathematics and physics, but it is the one we will adopt here.

Example: Homogeneous Principal Bundles

Our second example of a principal bundle is a homogeneous principal bundle. These are the central objects in the theory of homogeneous G-CNNs (Ch. 9).

Definition 7.2.10 (Homogeneous Principal Bundle)**.** *Let H be a Lie group, and $G \subset H$ a closed subgroup. Then H is a principal bundle with base space H/G and structure group G. The projection is defined as:*

$$\begin{aligned} H &\xrightarrow{\pi} H/G \\ h &\mapsto hG \end{aligned} \tag{7.27}$$

The right action of G on H is given by the group operation in H, that is, $h \mapsto hg$ for $h \in H$ and $g \in G$.

The fibers of H are cosets, i.e. $H_x = \pi^{-1}(x) = hG \subset H$ where $x = hG \in H/G$. Here hG plays the dual role of an element x of H/G and a fiber $H_x \subset H$.

To verify that this is indeed a principal G-bundle, we need to check that the projection is G invariant, i.e. that $\pi(hg) = \pi(h)$. Indeed this is the case, because $\pi(hg) = hgG = hG = \pi(h)$. This makes H into a G-bundle. It is also a principal G-bundle, because the right action of G on cosets is transitive and free.

Sections of a principal bundle: gauges

If we think of a principal bundle as a generalized bundle of frames, then a section of this bundle constitutes a choice of frame at each position on the base space. Such a choice is also called a *gauge*. The following theorem shows that a choice of section of a principal bundle P is equivalent to a choice of local trivialization of P, so we may refer to either one as a gauge.

Theorem 7.2.2. *Let $\pi : P \to B$ be a principal G-bundle. A choice of section $s : U \to P$ on $U \subset B$ determines a local trivialization $\varphi_s : P \to U \times G$ as follows:*

$$\varphi_s^{-1}(x, g) = s(x)g. \tag{7.28}$$

Conversely, a local trivialization $\varphi : P \to U \times G$ determines a local section $s : U \to P$ as follows:

$$s^{\varphi}(x) = \varphi^{-1}(x, e). \tag{7.29}$$

These processes are inverse to each other, in that

$$\begin{aligned} \varphi_{s^{\varphi}} &= \varphi \\ s^{\varphi_s} &= s. \end{aligned} \tag{7.30}$$

Proof. Let $s : U \to P$ be a local section of P. For φ_s to be a trivialization, it has to be a U-isomorphism and be equivariant. φ_s^{-1} is invertible, because given

$p \in P_U$, we obtain $x = \pi(p)$ and g is the unique element of G that satisfies $p = s(x)g$. For φ_s to be a U-isomorphism it has to satisfy $\text{proj}_1 \circ \varphi_s = \pi$, which is equivalent to $\text{proj}_1 = \pi \circ \varphi_s^{-1}$. And this holds, for $\pi(\varphi_s^{-1}(x,g)) = \pi(s(x)g) = \pi(s(x)) = x$. Finally, φ_s is equivariant, for $\varphi_s^{-1}(x,gh) = s(x)gh = \varphi_s^{-1}(x,g)h$.

In the other direction, we are given a local trivialization $\varphi : P \to U \times G$ and construct s^φ. This map is a section, for $\pi(s^\varphi(x)) = \pi(\varphi^{-1}(x,e)) = x$.

The processes are inverses, for

$$\begin{aligned} \varphi_{s^\varphi}^{-1}(x,g) &= s^\varphi(x)g = \varphi^{-1}(x,e)g = \varphi^{-1}(x,g) \\ s^{\varphi_s}(x) &= \varphi_s^{-1}(x,e) = s(x)e = s(x). \end{aligned} \tag{7.31}$$

■

Notice that it follows from this theorem that we can write any element $p \in P_U$ using a section $s : U \to P_U$ as $p = s(x)g = \varphi^{-1}(x,g)$, where $x = \pi(p)$ and $g \in G$. More generally, any equation we can write in terms of local trivializations of P, we can also write in terms of sections of P.

Theorem 7.2.3. *A principal bundle has a global section if and only if it is trivial.*

Proof. A bundle is trivial if and only if it has a global trivialization. By Theorem 7.2.2 it has a global trivialization if and only if it has a global section. ■

Theorem 7.2.3 tells us that when dealing with a non-trivial principal bundle, we will have to work with *local gauges*, i.e. sections $s : U \to P_U$ of trivializable pieces $P_U = \pi^{-1}(U)$ of P. For completeness, we provide the following definition.

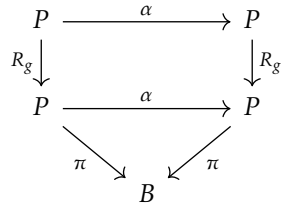
Definition 7.2.11 (Gauge)**.** *A (local) gauge is a (local) section of a principal bundle P. That is, it is a map $s : U \to P$ such that $\pi \circ s = \text{id}_U$ for $U \subseteq B$. By Theorem 7.2.3, sections of P are equivalent to trivializations $\varphi : P_U \to U \times G$, so we will also refer to (local) trivializations of P as (local) gauges.*

Gauge transformations

For reasons that will become clear shortly, we will refer to (B,G)-automorphisms of a principal bundle as gauge transformations:

Definition 7.2.12. *Gauge Transformation. A gauge transformation is a principal (B,G)-automorphism. That is, it is a map $\alpha : P \to P$ satisfying $\alpha(pg) = \alpha(p)g$ and*

$\pi \circ \alpha = \pi$. *Diagramatically:*



(7.32)

Gauge transformations form a group which we will refer to as $\mathrm{Aut}_{B,G}(P)$.

Since a gauge transformation α is a principal bundle isomorphism with $\beta = \mathrm{id}_B$, we know from Proposition 8 that for product bundles it takes the simple form $\alpha(x,g) = (x, g^\alpha(x)g)$, for some map $g^\alpha : B \to G$. It follows from local triviality that locally, any gauge transformation looks like this (Proposition 3). Hence the following definition:

Definition 7.2.13 (Local Gauge Transformation)**.** *Let $\pi : P \to B$ be a principal G-bundle, and $U \subset B$ a neighbourhood over which P is trivial. A map $g : U \to G$ is called a local gauge transformation.*

Why do we refer to these maps as gauge transformations?

Proposition 9. *Given two gauges $s : B \to P$ and $s' : B \to P$ of the same principal G-bundle, there exists a unique local gauge transformation $g^{ss'} : B \to G$ s.t.:*

$$s(x) = s'(x)g^{ss'}(x). \tag{7.33}$$

Proof. Because s and s' are sections, we know that both $s(x) \in P_x$ and $s'(x) \in P_x$. Since P is a principal bundle, the action of G on P_x is transitive and fixed-point free, and hence for any two elements of P_x there exists a unique $g \in G$ that maps one to the other. That is, there exists a unique $g^{ss'}(x) \in G$ such that $s(x) = s'(x)g^{ss'}(x)$. This map is smooth since s, s' are smooth. ∎

If sections and trivializations of P are essentially the same, and a change of section corresponds to a gauge transformation, then a change of trivialization (a transition map) should also correspond to a gauge transformation. Indeed this is the case:

Proposition 10. *A transition map $\varphi_{ij} = \varphi_i \circ \varphi_j^{-1} : U_{ij} \times G \to U_{ij} \times G$ between two local trivializations φ_i, φ_j of principal G-bundle P is a gauge transformation*

in the sense of Definition 7.2.12 and thus determines a local gauge transformation $g_{ij} : U_{ij} \to G$ in the sense of Definition 7.2.13. Furthermore g_{ij} is the gauge transformation between the sections s_i and s_j corresponding to φ_i and φ_j, respectively.

Proof. The transition map φ_{ij} is a composition of two principal U_{ij}-isomorphisms and hence is a principal U_{ij}-isomorphism. Indeed, it is a U_{ij}-isomorphism of $U_{ij} \times G$ to itself, i.e. an automorphism. According to Definition 7.2.12, such a map is called a gauge transformation of $U_{ij} \times G$.

By Proposition 8 a U_{ij}-isomorphism of product principal bundles takes the form:

$$\varphi_{ij}(x, g) = (x, g_{ij}(x)g), \tag{7.34}$$

so φ_{ij} determines (and is determined by) a local gauge transformation g_{ij} in the sense of Definition 7.2.13.

Now, let $s_i(x) = \varphi_i^{-1}(x, e)$ and $s_j(x) = \varphi_j^{-1}(x, e)$ be the sections corresponding to the local trivializations. By Proposition 9, there is a map (local gauge transformation) $h_{ij} : U_{ij} \to G$ such that $s_i(x) = s_j(x)h_{ij}(x)$. We show that $h_{ij} = g_{ij}$:

$$\begin{aligned} \varphi_{ij}(x, g) &= \varphi_i(\varphi_j^{-1}(x, e))g \\ &= \varphi_i(s_j(x))g \\ &= \varphi_i(s_i(x))h_{ij}(x)g \\ &= (x, e)h_{ij}(x)g \\ &= (x, h_{ij}(x)g). \end{aligned} \tag{7.35}$$

■

Change of Structure Group

We think of a principal bundle as a bundle of generalized frames, with any two frames related by a unique element of G. It is often convenient to limit the kind of frames under consideration, for example restricting from general linear to orthogonal frames (Definition 7.2.9). In this example, the linear frames are related by elements of the group $\mathrm{GL}(d, \mathbb{R})$, while the orthonormal ones are related by elements of $\mathrm{O}(d)$. If we further restrict to right-handed frames (which is only possible on an orientable manifold), the group is further restricted to $\mathrm{SO}(d)$. This is called a restriction of the structure group. More generally, one can consider lifts or changes of structure group. These notions can be elegantly captured via the notion of a principal B-morphism over θ (Definition 7.2.3).

Definition 7.2.14 (Change of Structure Group)**.** *Let $\pi_1 : P_1 \to B$ be a principal G_1 bundle and $\pi_2 : P_2 \to B$ a principal G_2 bundle. Let $\theta : G_1 \to G_2$ be a group homomorphism. Then a principal B-morphism $\alpha : P_1 \to P_2$ over θ is called a change of structure group of P_2 from G_2 to G_1.*

That is, a change of structure group of P_2 is a map $\alpha : P_1 \to P_2$ that satisfies:

$$\begin{aligned} \alpha(pg) &= \alpha(p)\theta(g) \\ \pi_2(\alpha(p)) &= \pi_1(p). \end{aligned} \tag{7.36}$$

for $g \in G_1$ and $p \in P_1$.

Intuitively, α_x maps the frames at x in P_1 to frames at x in P_2, and θ maps a change of P_1-frame $g \in G_1$ to a change of P_2-frame $\theta(g) \in G_2$.

A change of structure group from G_1 to G_2 is not always possible. For instance, changing the structure group of a frame bundle to the trivial group $G_1 = \{e\}$ is tantamount to a smooth global choice of frame, which does not exist in general. Similarly, we can only reduce to from $\mathrm{GL}(d, \mathbb{R})$ to $\mathrm{SO}(d)$ if the manifold is orientable. On the other hand, we can always reduce from $\mathrm{GL}(d, \mathbb{R})$ to $\mathrm{O}(d)$, because any manifold can be equiped with a metric.

Note further that a change of structure group depends on a group homomorphism $\theta : G_1 \to G_2$. We can always define a trivial homomorphism $\theta(g) = e$, but injective or surjective homomorphisms may not exist.

Definition 7.2.15 (Reduction of Structure Group)**.** *A reduction of the structure group from G_2 to G_1 is a principal B-morphism over θ, where $\theta : G_1 \to G_2$ is an injective homomorphism.*

Recall that if θ is injective, then G_1 is isomorphic to its image $\theta(G_1)$. Hence, the typical case (and essentially the only one) is the inclusion of a subgroup, e.g. $\theta : \mathrm{SO}(d) \hookrightarrow \mathrm{O}(d)$.

Reductions of the structure group are used in physics to model spontaneous symmetry breaking (Nikolova and Rizov, 1984; Keyl, 1991; Rudolph and Schmidt, 2017). Similarly, in the theory of G-CNNs we use reductions to limit the symmetries to which the network is equivariant in higher layers / larger length scales (See section 8.1).

Definition 7.2.16 (Lift of Structure Group)**.** *A lift of the structure group from G_2 to G_1 is a principal B-morphism over θ, where $\theta : G_1 \to G_2$ is a surjective homomorphism.*

Recall that if θ is surjective, then G_2 is isomorphic to $G_1/\ker\theta$. A typical example is a double cover $\theta : \mathrm{spin}(d) \to \mathrm{SO}(d)$ with kernel $\mathbb{Z}_2$. A non-example is a homomorphism $\theta : \mathrm{O}(d) \to \mathrm{SO}(d)$ (such a map cannot be surjective).

7.2.4 Associated Bundles

Given a principal G-bundle $P \xrightarrow{\pi_P} B$, we can replace the fibers $P_x \simeq G$ by a vector space V that carries a linear representation ρ of G to obtain a new G-bundle called the associated vector bundle. The notion of associated bundle can actually be defined in a more general way for G-bundles (not necessarily principal) and smooth G-actions on a manifold F (not just linear representations acting on a vector space), but we will not need this level of generality here.

The basic idea is that the elements of an associated bundle are geometric objects (elements of a vector space with a representation) that can be expressed relative to a gauge. The same geometric object has a different representation depending on the gauge, so an element of the associated bundle is defined as an equivalence class of pairs of frames and vectors.

Definition 7.2.17 (Associated bundle)**.** *Let V be a vector space, and let $\rho : G \to \mathrm{GL}(V)$ a representation of G on V. Furthermore, let $\pi_P : P \to B$ be a principal G-bundle. We define an equivalence relation on pairs $(p, v) \in P \times V$:*

$$(p, v) \sim (pg, \rho(g^{-1})v). \tag{7.37}$$

Then the bundle associated to P via (ρ, V) is defined as follows. It has total space

$$A = P \times_\rho V = (P \times V)/\sim \tag{7.38}$$

That is, the elements of $P \times_\rho V$ are the equivalence classes (orbits)

$$[p, v] = \{(pg, \rho(g^{-1})v) \mid g \in G)\}. \tag{7.39}$$

The projection of the associated bundle is given by

$$\pi_A([p, v]) = \pi_P(p). \tag{7.40}$$

This is well defined, for if we take another representative of the orbit $[p, v]$, we can always write it as $[pg, \rho(g^{-1})v]$, which (by Eq. 7.18) gives the same answer:

$$\pi_A([pg, \rho(g^{-1})v]) = \pi_P(pg) = \pi_P(p). \tag{7.41}$$

The associated bundle has the same base space B as the principal bundle P from which it was constructed, and fibers isomorphic to V.

We will now define a function that will be useful later in computations related to associated bundles. Given an element $a = [p, v] \in P \times_\rho V$ and a point

$p' = pg$ (i.e. $\pi_P(p') = \pi_P(p)$), we want to find the vector v' such that $a = [p', v']$. We define the map

$$r([p, v], p') = \rho(g^{-1})v \qquad \text{where } p' = pg. \tag{7.42}$$

This is well defined, for if we take the representative $[p'', v''] = [p, v]$, we have $p'' = ph$ and $v'' = \rho(h^{-1})v$, which implies that $p'' = ph = p'g^{-1}h$ and thus $p' = p''h^{-1}g$. It follows that

$$r([p'', v''], p') = \rho(g^{-1}h)v'' = \rho(g^{-1})v = r([p, v], p'). \tag{7.43}$$

So r does not depend on the particular representatives p and v of an orbit $[p, v]$.

Note that r has the property that $[p, r(a, p)] = a$ for any element $a \in P \times_\rho V$ and $\pi_P(p) = \pi_A(a)$.

Local associated bundle maps and sections

An associated bundle is a vector bundle, so bundle maps and sections, and their local forms, look like those of vector bundles, treated in Sec. 7.2.1. A *principal* G-bundle morphism $\alpha : P \to P'$ induces an associated bundle morphism from $P \times_\rho V$ to $P' \times_\rho V$. We will show this in Section 7.3.1 for principal bundle automorphisms, which induce associated bundle automorphisms. In particular, this is how gauge transformations (principal B-automorphisms) act on the associated bundle.

Lifting Sections to the Principal Bundle

In general, sections are more cumbersome to work with than functions. We have already seen that on trivializable local regions $U \subset B$, a section can be represented as a function from base to fiber. However, this has the downside of providing a local and gauge-dependent picture only. Switching between different local representations of a section requires extra bookkeeping. The following theorem shows how (for associated bundles) we can represent sections globally as functions on P, subject to linear constraints.

The basic idea is that we can encode the value of the section relative to any frame simultaneously. Recall that a point $p \in P$ encodes a position in the base space $x = \pi(p)$ and a frame at x. So a function $P \to V$ can tell us the value of a section relative to each frame at each point, and thus provide a gauge-independent representation of the field. For this to make sense though, the function has to satisfy a constraint on the values it takes at different frames at the same point in the base space. This constraint is given below.

Theorem 7.2.4. *Let P be a principal G-bundle, (ρ, V) a representation of G, and $A = P \times_\rho V$ the associated bundle. Then the space $\Gamma(A)$ of sections of A is isomorphic to the space of G-equivariant maps from P to V, denoted $\hat{\Gamma}(A)$ and called the space of lifted sections:*

$$\Gamma(A) \simeq \hat{\Gamma}(A) \equiv \{f : P \to V \mid f(pg) = \rho(g^{-1}) f(p),\ \forall g \in G, p \in P\} \tag{7.44}$$

Proof. We explicitly construct the isomorphism. Let $\sigma \in \Gamma(P \times_\rho V)$. Then we define the lifting isomorphism $\Lambda : \Gamma(P \times_\rho V) \to \hat{\Gamma}(P \times_\rho V)$,

$$(\Lambda\sigma)(p) = v \qquad \text{such that } \sigma(\pi(p)) = [p, v] \tag{7.45}$$

Another way to write this is

$$(\Lambda\sigma)(p) = r(\sigma(\pi(p)), p), \tag{7.46}$$

where r is as defined in eq. 7.42.

We must check that $\Lambda\sigma$ is G-equivariant, i.e. that $\Lambda\sigma \in \hat{\Gamma}(A)$:

$$\begin{aligned}(\Lambda\sigma)(pg) &= r(\sigma(\pi(pg)), pg) \\ &= r(\sigma(\pi(p)), pg) \\ &= \rho(g^{-1}) r(\sigma(\pi(p)), p) \\ &= \rho(g^{-1})(\Lambda\sigma)(p).\end{aligned} \tag{7.47}$$

Given a map $f \in \hat{\Gamma}(A)$, we can construct a section $\Lambda^{-1} f \in \Gamma(A)$:

$$(\Lambda^{-1} f)(x) = [s(x), f(s(x))], \tag{7.48}$$

where $s : U \subset B \to P$ is a local section of the principal bundle. One can verify that this construction is independent of the choice of s (hint: two sections s, s' are related by $s(x) = s'(x)g(x)$ for some local gauge transformation $g(x) \in G$), and that Λ^{-1} as constructed is indeed inverse to Λ. ∎

By describing sections as functions, many calculations become easier, and the result can always be re-expressed in terms of (local) sections.

7.3 Principal Bundle Automorphisms

In the general theory of equivariant convolutional networks, we aim to make networks that are equivariant to symmetry transformations of various kinds. We will describe these symmetries as acting on the principal bundle via automorphisms, so we study principal bundle automorphisms in this section.

In general, an automorphism is an isomorphism from a mathematical object to itself. The set of automorphisms of a given object form a group. According to the definitions we have already given, a principal bundle automorphism is a pair of diffeomorphisms $\alpha : P \to P$ and $\beta : B \to B$ satisfying $\pi \circ \alpha = \beta \circ \pi$ (because it is a bundle morphism; Definition 7.1.1), and $\alpha(pg) = \alpha(p)g$ (because it is a G-bundle morphism; Definition 7.2.3). Thus, a principal bundle automorphism is a diffeomorphism $\alpha : P \to P$ that maps fibers to fibers in an equivariant way. We will refer to the group of principal bundle automorphisms of P as $\mathrm{Aut}(P)$, and the group of principal (B, G)-automorphisms (i.e. gauge transformations) as $\mathrm{Aut}_{B,G}(P)$.

7.3.1 Action of automorphisms on associated bundles

The group $\mathrm{Aut}(P)$ acts on bundles associated to P via

$$\alpha([p, v]) = [\alpha(p), v], \tag{7.49}$$

where $\alpha \in \mathrm{Aut}(P)$ (note that the above is a slight abuse of notation, since we apply $\alpha(\cdot)$ to elements of the principal as well as associated bundles). This action is well defined, for if we have another representative of the equivalence class $[p, v]$, say $[pg, \rho(g^{-1})v]$, it gets mapped to the same element of the associated bundle $A = P \times_\rho V$:

$$\begin{aligned} \alpha([pg, \rho(g^{-1})v]) &= [\alpha(pg), \rho(g^{-1})v] \\ &= [\alpha(p)g, \rho(g^{-1})v] \\ &= [\alpha(p), v] \\ &= \alpha([p, v]). \end{aligned} \tag{7.50}$$

One verifies that this action is linear on fibers, and that $\pi_A \circ \alpha = \beta \circ \pi_A$. Hence, each principal bundle automorphism determines an associated bundle automorphism.

7.3.2 Action of automorphisms on sections of $P \times_\rho V$

Definition 7.3.1 (Action of $\mathrm{Aut}(P)$ on $\Gamma(A)$)**.** *The group of principal bundle automorphisms* $\mathrm{Aut}(P)$ *acts on the space of sections* $\Gamma(A)$ *of the associated vector bundle* $A = P \times_\rho V$ *via*

$$(\alpha \cdot \sigma)(x) = \alpha(\sigma(\beta^{-1}(x))) \tag{7.51}$$

When parsing this definition, notice that on the left hand side, $\alpha\cdot$ denotes the action of $\mathrm{Aut}(P)$ on $\Gamma(A)$, whereas on the right hand side $\alpha(\cdot)$ denotes the action of $\mathrm{Aut}(P)$ on A.

To check that this definition makes sense, note that we have the following signatures:

$$B \xrightarrow{\beta^{-1}} B \xrightarrow{\sigma} A \xrightarrow{\alpha} A, \tag{7.52}$$

so the composition is a map from B to A. For it to be a section of A, it has to map $x \in B$ to a point in the fiber $A_x \subset A$, which indeed it does:

$$\begin{aligned} \pi_A \circ \alpha \circ \sigma \circ \beta^{-1} &= \beta \circ \pi_A \circ \sigma \circ \beta^{-1} \\ &= \beta \circ \beta^{-1} \\ &= \mathrm{id}_B \end{aligned} \tag{7.53}$$

Where we used $\pi_A \circ \alpha = \beta \circ \pi_A$ (since α is an automorphism of A), followed by $\pi_A \circ \sigma = \mathrm{id}_B$ (the defining property of σ being a section of A). So we have shown that $\alpha \cdot \sigma \in \Gamma(A)$ for all $\sigma \in \Gamma(A)$. One can check that the action is associative and that $\mathrm{id}_P \cdot \sigma = \sigma$.

7.3.3 Local action of $\mathrm{Aut}(P)$ on $P \times_\rho V$

We have the action $\alpha([p, v]) = [\alpha(p), v]$ of $\mathrm{Aut}(P)$ on $P \times_\rho V$, and a pair of local trivializations φ_i, φ_j for $P \times_\rho V$ with domains $U_j \cap \beta(U_i) \neq \emptyset$ as before. Since the associated bundle is a G-bundle, the action trivializes as

$$\varphi_j(\alpha(\varphi_i^{-1}(b, v))) = (\beta(b),\ \rho(g_{ij}^\alpha(b))v). \tag{7.54}$$

7.3.4 Local action of $\mathrm{Aut}(P)$ on associated sections

We would also like to get a local expression for the action of $\mathrm{Aut}(P)$ on $\Gamma(A)$, for $A = P \times_\rho V$ an associated vector bundle. So let $\sigma \in \Gamma(A)$ be a section of A and $\alpha \in \mathrm{Aut}(P)$ a principal bundle automorphism over β. As in Prop. 3 we

pick charts (U_i, φ_i) and (U_j, φ_j) of A with $U_j \cap \beta(U_i) \neq \emptyset$. Then we write the following map from $U_j \cap \beta(U_i)$ to $U_j \cap \beta(U_i) \times V$

$$\begin{aligned} \varphi_j \circ (\alpha \cdot \sigma) &= \varphi_j \circ \alpha \circ \sigma \circ \beta^{-1} \\ &= (\varphi_j \circ \alpha \circ \varphi_i^{-1}) \circ (\varphi_i \circ \sigma) \circ \beta^{-1} \end{aligned} \tag{7.55}$$

in which we recognize a trivialized automorphism $\varphi_j \circ \alpha \circ \varphi_i^{-1}$ (eq. 7.54), a trivialized section $f_i = \text{proj}_2 \circ \varphi_i \circ \sigma$ (eq. 7.15), and a transformation β^{-1}. Combining these, we obtain the local expression of $\alpha \cdot \sigma$,

$$\begin{aligned} \varphi_j \circ (\alpha \cdot \sigma)(x) &= (\beta(\beta^{-1}(x)),\ \rho(g_{ij}^{\alpha}(\beta^{-1}(x)))\ f_j(\beta^{-1}(x))) \\ &= (x,\ \rho(g_{ij}^{\alpha}(y))\ f_i(y)), \end{aligned} \tag{7.56}$$

where we set $y = \beta^{-1}(x)$. Thus we have found a local expression of $\alpha \cdot \sigma$ in terms of the local expression g_{ij}^{α} of α and local expression f_i of σ.

7.3.5 Lifting the action of automorphisms

We have an action $\alpha\cdot : \Gamma(P \times_\rho V) \to \Gamma(P \times_\rho V)$ of $\text{Aut}(P)$ on $\Gamma(P \times_\rho V)$ and a lifting isomorphism $\Lambda : \Gamma(P \times_\rho V) \to \hat{\Gamma}(P \times_\rho V)$ (see Section 7.2.4), so we can lift the action of $\text{Aut}(P)$ by conjugating with Λ. One can show with a bit of symbol manipulation that this results in the following very simple expression for the action of $\text{Aut}(P)$ on $\hat{\Gamma}(P \times_\rho V)$:

$$(\alpha \cdot f)(p) = f(\alpha^{-1}(p)), \tag{7.57}$$

where $f \in \hat{\Gamma}(P \times_\rho V)$ is a lifted section. Note that we use the same symbol $\cdot$ to denote the action of $\text{Aut}(P)$ on sections as well as lifted sections.

We easily verify that $\alpha \cdot (\alpha' \cdot f) = (\alpha \circ \alpha') \cdot f$, and that

$$\begin{aligned} (\alpha \cdot f)(pg) &= f(\alpha^{-1}(pg)) \\ &= f(\alpha^{-1}(p)g) \\ &= \rho(g^{-1}) f(\alpha^{-1}(p)) \\ &= \rho(g^{-1})(\alpha \cdot f)(p). \end{aligned} \tag{7.58}$$

So $\alpha \cdot f \in \hat{\Gamma}(A)$. Moreover, the action of α on $\hat{\Gamma}(A)$ is linear.

8

General Theory of G-CNNs

We have presented in Part I a sequence of increasingly general convolutional neural networks, which we propose to refer to generically as G-CNNs or, if no convolution is involved, G-NNs. The letter *G* may be taken to stand for Group, Gauge, Generalized, or Geometrical, or, if one prefers, Gauss, Galois, Grassmann, or Grothendieck. So as to leave no confusion regarding the meaning of the term, we present here a precise and general definition. With the benefit of hindsight, it is possible to give a definition that covers each method presented in Part I as a special case (something that was obviously not possible when we first used the term G-CNN in (Cohen and Welling, 2016)).

We think of G-CNNs as a general framework for geometric deep learning that puts symmetry front and center. Although the framework does not say which network architecture is best for a given application, it provides a large collection of building blocks and rules for combining them into geometric models. It is both permissive (in that many kinds of networks can be built) and restrictive (in that it forbids certain layers to be composed, and tells us that certain kinds of layers cannot exist). We will provide in this chapter many examples to illustrate both the breadth and specificity of the framework.

The definition of G-CNNs (Section 8.1) consists of two parts: a definition of *representation spaces* (Section 8.1.1; examples in Section 8.1.2) and a definition of *layers*, also known as *maps between representation spaces* (Section 8.1.3). Among other things, the definition says that G-CNN representation spaces are spaces of fields over a manifold B (sections of an associated vector bundle). Each representation space comes with a principal and associated vector bun-

dle, and a group of (global or gauge) symmetries acting by principal bundle automorphisms. A layer is defined in terms of a space of equivariant maps between representation spaces, as well as certain maps that relate the symmetry groups and principal bundles of the input and output representation spaces. The notion of layer is flexible enough to support symmetry breaking, a change of base space and field type, and other design patterns.

8.1 Definition of G-CNNs

We begin by defining representation spaces and their symmetries.

8.1.1 Representation Spaces

Definition 8.1.1 (G-CNN Representation Space)**.** *A G-CNN representation space is defined in terms of the data* $(P, B, G, \pi_P, S, \rho, V, \alpha)$ *(all assumed smooth):*

1. *A principal bundle* $\pi_P : P \to B$ *with structure group* G*.*
2. *A group of symmetries* S *acting on* P *via principal bundle automorphisms, i.e. via a group homomorphism* $\alpha : S \to \mathrm{Aut}(P)$*. The action is denoted by* $s \cdot p = \alpha_s(p)$ *for* $p \in P$ *and* $s \in S$*.*
3. *A linear representation* $\rho : G \to \mathrm{GL}(V)$ *of* G *on vector space* V*.*

Together these determine:

1. *The associated bundle* $A = P \times_\rho V$*.*
2. *The representation space* $\Gamma(A)$ *proper, i.e. the space of* ρ*-fields over* B*.*
3. *The action of* S *on* A *and* $\Gamma(A)$*.*

It is customary to denote a principal bundle (P, B, G, π_P) *by* P*, a representation* (ρ, V) *by* ρ *and the action of* S *on* P *by automorphisms by* $s \cdot p$ *for* $s \in S$ *and* $p \in P$*. Hence, we will usually denote a representation space simply by* (P, S, ρ)*, leaving implicit the base space* B*, structure group* G *and projection* π_P *of* P *as well as the homomorphism* α*. Furthermore, we will refer to both* $\Gamma(A)$ *and the tuple* (P, S, ρ) *as a representation space. We will also refer to a representation space as a feature space, convolutional feature space, or space of feature maps, because all of these are common in the literature. Likewise, an individual* $\sigma \in \Gamma(A)$ *will be refered to as a section, field, feature field, stack of feature maps, or feature map.*

Thus, in the G-CNN framework, a feature space is modelled mathematically as a space of fields over some manifold B. This manifold also serves as the base space of the principal bundle $\pi_P : P \to B$. As discussed in the previous chapter, a principal G-bundle P over B can be thought of as a bundle whose fiber P_x at each point $x \in B$ is a space of reference frames at B. Depending on the specific P under consideration, the fiber P_x may contain frames of the tangent space T_xB, or of another vector space (the fiber of some vector bundle). In any case, by definition of principal bundle, any two frames $p, q \in P_x$ are related by a unique $g \in G$ via $p = qg$ (e.g. orthogonal frames are related by orthogonal transformations). Once we make a smooth choice of frame at each position in some region $U \subset B$ (i.e. choose a section $s : U \to P$) we can represent a field relative to this frame as a function from U to V. But conceptually, the field exists independent of a choice of frame.

If the associated bundle is trivial, a field can be described as a globally defined function from B to V (see Section 7.1.7). This happens for instance when ρ is trivial (the field is scalar) or when the principal bundle is trivial (e.g. when $B = \mathbb{R}^2$). For example, in the case of regular and steerable G-CNNs on the (discretized) plane discussed in Chapters 2 and 3, the bundles are trivial and so in those chapters we described feature maps as functions. In Chapter 4 we implicitly worked with a non-trivial principal bundle $\pi : \mathrm{SO}(3) \to S^2$, but only used scalar fields, thus avoiding the need for the full mathematical machinery. However, if one wants to describe for instance a spherical CNN with vector field feature maps, or gauge equivariant CNNs on general manifolds, a proper mathematical description really does require principal and associated bundles, local sections, and so forth. Let us look at some examples.

8.1.2 Examples of Representation Spaces

Example 8.1.1 (Planar Images and Translations)**.** *Our first and most trivial example is a mainstay of computer vision: planar images. Although in practice we can sense only a finite number of discrete pixels, we consider here an idealized image plane $\mathbb{R}^2$. To describe the space of planar images as a G-CNN representation space (Def. 8.1.1), we choose the following data:*

1. *The base space $B = \mathbb{R}^2$ is the image plane. The principal bundle may be chosen as the frame bundle $P = F_{\mathrm{GL}}(\mathbb{R}^2) \simeq \mathbb{R}^2 \times \mathrm{GL}(2, \mathbb{R})$ (see Section 7.2.3). This is a trivial bundle with projection $\pi_P = \mathrm{proj}_1 : \mathbb{R}^2 \times \mathrm{GL}(2, \mathbb{R}) \to \mathbb{R}^2$ and structure group $G = \mathrm{GL}(2, \mathbb{R})$.*

2. *For the symmetry group, we will choose $S \simeq \mathbb{R}^2$, the 2D translation group.*

Translations can act as principal bundle automorphism because a translation $\beta : \mathbb{R}^2 \to \mathbb{R}^2$, being a diffeomorphism, can be lifted uniquely to an automorphism $\alpha : F_{\mathrm{GL}}(\mathbb{R}^2) \to F_{\mathrm{GL}}(\mathbb{R}^2)$ of the frame bundle – see Theorem 7.2.1.

3. *A grayscale image is a scalar field so we choose $V = \mathbb{R}$ and the trivial representation $\rho(g) = 1$ of the structure group $\mathrm{GL}(2, \mathbb{R})$. Similarly, an RGB image is a collection of 3 scalar fields, so in this case we choose $V = \mathbb{R}^3$ and $\rho(g) = I_3$ (the 3D identity matrix). More generally, a stack of n feature maps in a conventional planar CNN representation space corresponds to $V = \mathbb{R}^n$ and $\rho(g) = I_n$.*

Since ρ is trivial, the associated bundle (Def. 7.2.17) $A = P \times_\rho V$ is just $\mathbb{R}^2 \times V$ with projection $\mathrm{proj}_1 : \mathbb{R}^2 \times V \to \mathbb{R}^2$. Technically, a ρ-field is a section $\sigma \in \Gamma(A)$ (Def. 7.1.4) of this bundle, i.e. a map $\sigma : \mathbb{R}^2 \to \mathbb{R}^2 \times V$ that satisfies $\mathrm{proj}_1 \circ \sigma = \mathrm{id}_{\mathbb{R}^2}$. Since $\mathbb{R}^2 \times V$ is a trivial bundle, a section is essentially just a map $f_\sigma : \mathbb{R}^2 \to V$, i.e. a map that assigns to each position $x \in \mathbb{R}^2$ a scalar or RGB value $v \in V$. That is, sections of this associated vector bundle are grayscale/RGB images.

The symmetries are translations $\beta_t(x) = x + t$. It follows from Eq. 7.56 and the fact that $\rho(g) = 1$ that these act on fields f_σ as $f_\sigma(x) \mapsto f_\sigma(x - t)$. This we recognize as the regular representation of $\mathbb{R}^2$.

Since the frame bundle of $\mathbb{R}^2$ is trivial, it is possible to reduce the structure group (Def. 7.2.15) from $\mathrm{GL}(2, \mathbb{R})$ to the trivial group $\{e\}$, in which case the principal bundle $P \simeq \mathbb{R}^2 \times \{e\} \simeq \mathbb{R}^2$. This amounts to choosing a global frame. In order to have a well defined lift of translations to P, the frames should be globally aligned (e.g. aligned with the X, Y coordinate axes), so that translations t map the frame at x to the frame at $x + t$ for any $x \in \mathbb{R}^2$.

Clearly, the principal bundle framework is complete overkill for describing planar images and translations, but it is nevertheless important to show that the general definition captures relevant simple cases as well, and we hope that it can serve to illustrate the general definition. The examples that follow will gradually include more and more aspects of the general definition.

Example 8.1.2 (Planar Images and Roto-translations)**.** *As in the previous example, take $B = \mathbb{R}^2$, $P = F_{\mathrm{GL}}(\mathbb{R}^2) \simeq \mathbb{R}^2 \times \mathrm{GL}(2, \mathbb{R})$ with structure group $G = \mathrm{GL}(2, \mathbb{R})$. We will also again take the trivial representation $\rho(g) = 1$ to model grayscale images, or $\rho(g) = I_3$ to model RGB images.*

Unlike the previous example, we choose symmetry group $S = \mathrm{SE}(2) = \mathbb{R}^2 \rtimes \mathrm{SO}(2)$, the Special Euclidean group which consists of rotations and translations. An element $\beta_{r,t} \in \mathrm{SE}(2)$ acts on $\mathbb{R}^2$ as $\beta_{r,t}(x) = rx + t$ (here r is a rotation and t a translation). As any diffeomorphism, we can lift $\beta_{r,t}$ to a principal bundle automorphism $\alpha_{r,t} : P \to P$ of the frame bundle. By Proposition 8, this map takes the

form $\alpha_{r,t}(x,g) = (\beta_{r,t}(x), g^{\alpha_{r,t}}(x)g)$, *for some map* $g^{\alpha_{r,t}} : \mathbb{R}^2 \to \mathrm{GL}(2,\mathbb{R})$. *The map* $g^{\alpha_{r,t}}$ *corresponds to the differential of* $\beta_{r,t}$, *so we have* $g^{\alpha_{r,t}}(x) = r$. *Thus, the automorphism* $\alpha_{r,t}$ *takes the form:*

$$\alpha_{r,t}(x,g) = (rx + t, rg). \tag{8.1}$$

The frame $g \in \mathrm{GL}(2,\mathbb{R})$ *at* $x \in \mathbb{R}^2$ *is mapped to the frame* rg *at* $rx + t$.

As in the previous example, we obtain the associated bundle $A = P \times_\rho V = \mathbb{R}^2 \times V$. *Sections (feature maps)* $\sigma \in \Gamma(A)$ *correspond to globally defined maps* $f_\sigma : \mathbb{R}^2 \to V$. *The action of* $\beta_{r,t} \in \mathrm{SE}(2)$ *on* $\Gamma(A)$ *is given by*

$$(\beta_{r,t} \cdot f)(x) = \rho(g^{\alpha_{r,t}}(x)) f(\beta_{r,t}^{-1}(x)) = f(r^{-1}(x - t)). \tag{8.2}$$

This follows from Equation 7.56 and the fact that $\rho(g) = 1$ *for all* $g \in G$. *The result says that to rotate and translate an image we take the pixel at* $r^{-1}(x - t)$ *and move it to the pixel at* x, *without acting on the grayscale or RGB pixel itself.*

In the previous two examples, we used trivial representations to model scalar fields. We now look at a case where the group not only moves feature vectors / fibers around, but also acts on them.

Example 8.1.3 (Optical Flow)**.** *In computer vision, the motion between two frames in a video is often described by a planar vector field, called optical flow. The space of vector fields on* $\mathbb{R}^2$ *is a representation space, so a G-CNN can take such a vector field as input, produce it as output, or use it as an intermediate representation.*

To model the space of optical flows as a G-CNN representation space, we choose the same B, P, G *as in Example 8.1.1, i.e. the image plane* $B = \mathbb{R}^2$ *and the frame bundle* $P = F_{\mathrm{GL}}(\mathbb{R}^2)$ *with structure group* $G = \mathrm{GL}(2,\mathbb{R})$.

What distinguishes a vector field from a pair of scalar fields is that the coefficients of a 2D vector will undergo a change if we change the frame, whereas a scalar is invariant to changes of frame. So whereas we used the trivial representation to model scalar fields, for a vector field we use the standard matrix representation $\rho(g) = g$ *for* $g \in \mathrm{GL}(2,\mathbb{R})$ *acting on* $V = \mathbb{R}^2$ *(not to be confused with* $B = \mathbb{R}^2$*).*

This difference does not play a role if we only consider translation symmetries. If we consider changing the frame (gauge symmetry) or take $S \simeq \mathrm{SE}(2)$ *however, then a difference appears. Here we consider* $S = \mathrm{SE}(2)$. *As before, we have* $\beta_{r,t}(x) = rx + t$ *and* $\alpha_{r,t}(x,g) = (rx + t, rg)$.

As an aside, if we reduce the structure group from $\mathrm{GL}(2,\mathbb{R})$ *to* $\mathrm{SO}(2)$ *(i.e. consider only positively oriented orthogonal frames) then we obtain the bundle* $F_{\mathrm{SO}}(2)$, *which is equivalent to* $\mathrm{SE}(2)$ *as a manifold. Moreover, the action of* $\alpha_{r,t}$ *given above*

is then just the composition of elements (t,r) and (x,g) in SE(2)*, i.e. it is the left action of* SE(2) *on itself.*

Since the associated bundle $A = F_{\mathrm{GL}}(\mathbb{R}^2)\times_{\mathrm{GL}(2,\mathbb{R})}\mathbb{R}^2 \simeq \mathbb{R}^2\times\mathbb{R}^2$ is trivial, a vector field on $\mathbb{R}^2$ can be described globally as a function $f:\mathbb{R}^2\to\mathbb{R}^2$. By Equation 7.56, fields transform as follows:

$$(\alpha_{r,t}\cdot f)(x) = rf(r^{-1}(x-t)). \tag{8.3}$$

We recognize this as the representation of SE(2) *induced by the standard representation $\rho(r) = r$ of* SO(2) *(c.f. Equation 3.6). Conceptually, this says that to rotate and translate a vector field, we first move the vector at $r^{-1}(x-t)$ to x (just like for an RGB image) but then in addition we rotate the vector itself by r.*

We note that although the previous example is about defining a representation space that is appropriate for a certain kind of input data (optical flows), one can equally well use it as an internal representation in the neural network.

Example 8.1.4 (Harmonic Networks)**.** *Harmonic Networks were introduced by Worrall et al. (2017) and can be understood in the G-CNN framework. They are* SE(2) *equivariant convolutional networks whose features transform according to irreducible representations of* SO(2)*.*

Let $B = \mathbb{R}^2$ and let $P = F_{\mathrm{SO}}(\mathbb{R}^2)$ be the bundle of right-hand oriented orthogonal frames on the plane. P has structure group $G =$ SO(2). As symmetry group, consider again $S =$ SE(2), acting on B via $\beta_{r,t}(x) = rx+t$ and on P via $\alpha_{r,t}(x,g) = (rx+t, rg)$.

What distinguishes Harmonic Networks from previous examples is the choice of representation ρ. Namely, Harmonic Networks use irreducible representations of SO(2)*. These are labelled by integers n with $\rho_0(\theta) = 1$ and*

$$\rho_n(\theta) = \begin{bmatrix} \cos(n\theta) & -\sin(n\theta) \\ \sin(n\theta) & \cos(n\theta) \end{bmatrix}. \tag{8.4}$$

Here θ is the rotation angle of an element of SO(2)*. Note that ρ_1 is the standard representation of* SO(2)*, which corresponds to a vector field representation space, as discussed in the previous example.*

In practice we will want more than two channels, so the representation ρ that is used is block-diagonal with C irreducible blocks:

$$\rho(\theta) = \begin{bmatrix} \rho_{n_1}(\theta) & & \\ & \ddots & \\ & & \rho_{n_C}(\theta) \end{bmatrix} \tag{8.5}$$

where each irreducible representation ρ_n may occur multiple times, and typically the maximum n is kept fairly small for numerical reasons.

We note that there are some superficial differences between our description here and that in Worrall et al. (2017). Firstly, Worrall et al. did not use representation theoretic language. Secondly, we have for consistency used real-valued representations, whereas Worrall et al. use complex numbers $\rho_n(\theta) = e^{in\theta}$.

Here we have described the representation space used by Harmonic Networks as a G-CNN representation space, but we have not described the network layers used to map from one representation space to another (maps will be covered in the next section). It can be shown however that the space of equivariant linear maps between two Harmonic Network representation spaces takes the form described by Worrall et al. (see Example 8.2.2).

Finally, we note that analogous methods for 3D data have been developed: Tensor Field Networks (Thomas et al., 2018) and 3D Steerable CNNs (Weiler et al., 2018a). The latter goes into more detail on the representation theoretic description of these networks for the 3D case.

The following example is similar to the vector field case (Example 8.1.3), only now we work with base space $\mathbb{R}^3$ and a different representation ρ.

Example 8.1.5 (Diffusion Tensor Image)**.** *DTI is a technique that can be used to measure water diffusion in neural tissue and thereby track fiber bundles of the biological kind. A diffusion tensor image is a three dimensional signal with at each point in $B = \mathbb{R}^3$ a 3×3 matrix (2-tensor), whose eigenvectors and eigenvalues indicate the direction and amount of water diffusion.*

To model the space of diffusion tensor images as a G-CNN representation space, we choose $B = \mathbb{R}^3$, $P = F_{\mathrm{SO}}(\mathbb{R}^3) \simeq \mathbb{R}^3 \times \mathrm{SO}(3)$ the orthogonal frame bundle. We choose $S = \mathrm{SE}(3)$ as the symmetry group. For the vector space V (canonical fiber of the associated bundle), we choose $\mathbb{R}^{3\times3}$, the space of 3×3 matrices. The group $\mathrm{SO}(3)$ acts on $\mathbb{R}^{3\times3}$ as a change of basis matrix:

$$\rho(r)M = rMr^{-1}, \tag{8.6}$$

where $M \in \mathbb{R}^{3\times3}$. Another way to write this is $\rho = \tilde{\rho} \otimes \tilde{\rho}$, where $\tilde{\rho}$ is the standard representation $\tilde{\rho}(r) = r$ of $\mathrm{SO}(3)$ on vectors in $\mathbb{R}^3$. As an aside, it can be shown that this representation decomposes into irreducible representations of dimension 1 (the trace), 3 (the anti-symmetric part) and 5 (the traceless symmetric part) (Weiler et al., 2018a).

Similar to the previous examples, we have the associated bundle $A = \mathbb{R}^3 \times \mathbb{R}^{3\times3}$ and the space of sections $\Gamma(A)$ which is equivalent to the space of maps $f : \mathbb{R}^3 \to$

$\mathbb{R}^{3\times 3}$ *because A is trivial. The action of* SE(3) *on fields is derived from Eq. 7.56 as*

$$(\alpha_{r,t} \cdot f)(x) = \rho(r) f(r^{-1}(x-t)). \tag{8.7}$$

This is called the representation of SE(3) *induced by the representation* ρ *of* SO(3).

So far, we have considered examples where the symmetry group S is a finite dimensional Lie group such as $\mathbb{R}^n$ or $\mathrm{SE}(n)$. The following example considers instead the group $S = \mathrm{Aut}_{B,G}(P)$ of gauge transformations of a frame bundle P, also known as *external* gauge transformations.

Example 8.1.6 (External gauge symmetries)**.** *Consider again the previous example with* $B = \mathbb{R}^3$, $P = F_{\mathrm{SO}}(\mathbb{R}^3) = \mathbb{R}^3 \times \mathrm{SO}(3)$ *and* (ρ, V) *the representation of* SO(3) *on 2-tensors* $M \in V = \mathbb{R}^{3\times 3}$. *(Alternatively, consider a vector field on* $\mathbb{R}^2$ *as in Example 8.1.3.)*

DTI brain scans are generally registered, meaning that the brain volume always appears in the same position and orientation. Hence, it may not be appropriate to consider every position as equivalent, i.e. to assume that there is a translation symmetry. It is also not appropriate to assume a global SO(3) *rotation symmetry that acts by rotating* $\mathbb{R}^3$ *around the origin. Nevertheless, we may wish to consider every local* direction *to be equivalent, at least at small scales. This can be implemented by choosing as symmetry group* $S = \mathrm{Aut}_{B,G}(P)$, *the group of* (B, G)*-automorphisms of* $P = F_{\mathrm{SO}}(\mathbb{R}^3)$, *also known as gauge transformations (Section 7.2.3).*

A map $\alpha \in \mathrm{Aut}_{B,G}(P)$ *is a map* $\alpha : P \to P$ *satisfying* $\alpha(pg) = \alpha(p)g$ *as well as* $\mathrm{proj}_1 \circ \alpha = \mathrm{proj}_1$ *(where* proj_1 *is the projection of* $P = F_{\mathrm{SO}}(\mathbb{R}^3)$*). We know from the discussion in Sec. 7.2.3 that on a trivializing neighbourhood (in this case all of B) we can represent a gauge transformation as a map* $g^\alpha : B \to G$, *in this case* $g^\alpha : \mathbb{R}^3 \to \mathrm{SO}(3)$. *A gauge transformation acts on a field* $f : B \to V$ *as* $f(x) \mapsto \rho(g^\alpha(x))^{-1} f(x)$.

In Chapter 10 we will see that a linear map that is equivariant to external gauge transformations has rotational weight sharing. But in the absence of translation symmetry, it will not have translational weight sharing. In the neural networks literature, maps that use local filters but no translational weight sharing are called locally connected layers. Thus, assuming locality, the $\mathrm{Aut}_{B,G}(P)$*-equivariant linear maps are locally connected layers with rotational weight sharing. Under the stated assumptions, these would be appropriate for analyzing registered DTI scans.*

In the previous example we considered *external* gauge symmetries. That is, the gauge/frame is related to the directions in the base space by virtue of being a section of the *tangent frame bundle* $P = F_{\mathrm{GL}}(B)$. In physics, there is another

kind of gauge and gauge transformation called *internal*. In fact, the word gauge and gauge transformation is sometimes reserved for internal symmetries only. We now consider an example of internal gauge symmetries that may be useful in computer vision.

Example 8.1.7 (Planar Image with internal gauge symmetry)**.** *Consider again the image plane $B = \mathbb{R}^2$ and fiber $V = \mathbb{R}^3$ corresponding to the RGB or HSV (Hue-Saturation-Value) channels of a color image. Instead of taking $P = F_{\mathrm{GL}}(B)$ as in Ex. 8.1.1, we could take $P = B \times S_3$, where S_3 is the permutation group on 3 elements, which permutes the RGB color channels via a 3×3 permutation matrix $\rho(g)$ for $g \in S_3$. Alternatively, we could take $P = B \times \mathrm{SO}(2)$, where $\mathrm{SO}(2)$ acts on HSV pixels via ρ by rotating along the hue coordinate. So we have $G = S_3$ or $G = \mathrm{SO}(2)$.*

A gauge transformation is now a map $g : B \to G$, acting on color images $f : \mathbb{R}^2 \to \mathbb{R}^3$ via $f(x) \mapsto \rho(g(x))^{-1} f(x)$. That is, we can independently change the values of the color channels by $g(x) \in G$ at each pixel $x \in B$. If we reinterpret these values as RGB intensities, then such a transformation can lead to a large change in the image and we may question whether this should be considered a symmetry. But if we remember that the values have a meaning only relative to the new and perhaps unusual gauge $s'(x) = s(x)g(x)$ then nothing has changed*; the same information is contained in the image, just expressed differently.*

In Chapter 10 we will study gauge equivariant linear maps between representation spaces. We will then see that to build a non-trivial gauge equivariant linear map, we need to introduce a connection*, which in the present example would allow us to transport an RGB pixel from x to y in $\mathbb{R}^2$ so that we can compare it to the pixel at y, and say whether they are* the same*, even if their numerical representation at x and y is different. Then, this internal gauge symmetry says nothing more than that each color channel / hue should be treated identically. A gauge equivariant linear map will thus have weight sharing between filters related by a color-space transformation.*

All of the bundles considered so far in this section are trivial in the technical sense (Section 7.1.3). In this case we can choose a global gauge $s : B \to P$ (Theorem 7.2.2) relative to which we can express a field as a function $f : B \to V$. In the following example, the principal bundle is non-trivial but the associated bundle is still trivial, because we choose ρ to be trivial representation.

Example 8.1.8 (Scalar field on S^2)**.** *We may wish to represent a scalar field, such as the temperature distribution on earth. To this end, let $B = S^2$ and $P = F_{\mathrm{SO}}(S^2)$ the bundle of oriented orthogonal frames S^2. Let $V = \mathbb{R}$ and $\rho(g) = 1$. Fields (sections of $P \times_\rho V$) are then scalar functions $f : B \to \mathbb{R}$. In Chapter 4 on spherical*

G-CNNs, we assumed that the input of the network was of this kind. As in that chapter, we take $S = \mathrm{SO}(3)$ as symmetry group. Similar to Example 8.1.1 (scalar field on the plane with translation symmetry), the action on the representation space is $f(x) \mapsto f(\beta^{-1}(x))$ for $\beta \in \mathrm{SO}(3)$.

In the next example, both principal and associated bundle are non-trivial.

Example 8.1.9 (Vector field on S^2)**.** *Consider again the sphere $B = S^2$ with the bundle of oriented orthogonal frames $P = F_{\mathrm{SO}}(S^2)$, and symmetry group $S = \mathrm{SO}(3)$. As in Example 8.1.3, we model a vector field (e.g. wind directions on earth) by choosing $V = \mathbb{R}^2$ and $\rho(r) = r$ the standard representation of $G = \mathrm{SO}(2)$ on 2D tangent vectors.*

The bundle $F_{\mathrm{SO}}(S^2)$ is non-trivial, so there is no global gauge $s : S^2 \to F_{\mathrm{SO}}(S^2)$. This follows from the hedgehog theorem, which says that there exists no smooth nowhere vanishing vector field on the sphere (let alone two such vector fields which are orthogonal everywhere, i.e. a frame).

Because of this non-triviality, it is not possible to express a smooth vector or tensor field as a globally defined smooth map $f : S^2 \to V$. Instead we need to choose local charts $U_i \subset S^2$ (e.g. the North and South half sphere with overlap), and choose a gauge on those charts. The field is then expressed as a collection of maps $f_i : U_i \to V$, one per chart, that agree on overlaps.

The action of $S = \mathrm{SO}(3)$ on such local data is given by Eq. 7.56. A more simple description can be given for lifted sections $f : P \to V$ (Section 7.2.4), although their redundant nature makes them unsuitable for use in an efficient implementation. As derived in Section 7.3.5, the action of a principal bundle automorphism (β, α) on lifted sections of the associated bundle is given by $f(p) \mapsto f(\alpha^{-1}(p))$. In the present example, β is an element of $\mathrm{SO}(3)$ and α is the unique lift of β to $F_{\mathrm{SO}}(S^2)$.

The following example covers a general class of G-CNNs: the homogeneous G-CNNs. We give a sketch here, and a more detailed study in Chapter 9.

Example 8.1.10 (Homogeneous G-CNN representation space)**.** *Let $P = H$ be a compact Lie group, and $G \subset H$ a closed subgroup. As discussed in Section 7.2.3, H is a principal bundle over $B = H/G$ with projection $\pi : H \to H/G$ that maps each $h \in H$ to the coset $\pi(h) = hG$. For the group of symmetries we choose $S = H$, acting on $P = H$ by the left action $h \mapsto h'h$. The left action is indeed a principal bundle automorphism. Choosing any representation (ρ, V) of G, we obtain the associated bundle $A = H \times_\rho V$ and representation space $\Gamma(A)$. The action of $S = H$ on $\Gamma(A)$ is known as the representation of H induced by the representation ρ of G.*

Example 8.1.11 (SHOT descriptors on a manifold)**.** *The SHOT descriptor (Salti et al., 2014) can be used to characterize the local surface geometry of a mesh or point*

cloud as a finite dimensional vector. SHOT is sometimes used as input in geometric deep learning by attaching to each point in a mesh a SHOT descriptor (Monti et al., 2016; Bronstein et al., 2016). In other words, the input data can be thought of as a field of SHOT descriptors on a 2D surface embedded in 3D Euclidean space.

SHOT is based on similar principles as the popular SIFT descriptor used in computer vision (Lowe, 2004). We will not go into detail, but note that a SHOT descriptor consists of a histogram whose bins correspond to local orientations and these orientations are defined relative to a local orthogonal reference frame. So, up to discretization, a shot descriptor vector transforms according to a certain representation ρ of $\mathrm{O}(2)$. As such, when processing a field of SHOT vectors on a mesh, one should treat them as a ρ-feature and usually assume external gauge symmetry, rather than treating the SHOT vector as a list of scalars as is sometimes done.

Formally, the representation space can be described as follows. We have a manifold B (approximated by a mesh) and the orthogonal frame bundle $P = F_{\mathrm{O}}(B)$ with structure group $G = \mathrm{O}(2)$. As symmetries we take external gauge symmetries, i.e. $S = \mathrm{Aut}_{B,G}(P)$. On a trivializing neighbourhood $U \subset B$ a gauge transformation is a map $g : U \to \mathrm{O}(2)$. A local gauge transformation acts on a local signal $f : U \to V$ by $f(x) \mapsto \rho(g(x))f(x)$, where ρ is the SHOT representation.

Representation Spaces in Physics

We have seen that the definition of representation space (8.1.1) captures many practically relevant examples of feature spaces encountered in applications of geometric deep learning. The mathematical language we used to describe these diverse situations is also the de-facto standard language of theoretical physics (and in fact was developed in symbiosis with physics). Indeed, fiber bundles are indispensable in gauge theory, and as discussed by Weatherall (2014), the theory of relativity can equally well be described in this framework.

In this analogy, a feature map $\sigma \in \Gamma(A)$ of an associated vector bundle A is like a matter field. In physics, the base space B usually plays the role of space-time. The principal bundle is the bundle of frames of (the fibers of) A. In general relativity, these are the frames of the tangent spaces of B (which are related to directions in the base space), whereas in gauge theory the fibers of A are not necessarily related to B, but rather encode other "internal" features such as electromagnetic charge, color charge, etc. (Weatherall, 2014). It is also possible to combine internal and external symmetries, which suggests an interesting direction for future research in machine learning. For more details and examples, we refer to reader to (Weatherall, 2014; Rudolph and Schmidt, 2017; Bleecker, 1981; Hamilton, 2017).

8.1.3 Layers: Maps between Representation Spaces

Having defined feature spaces and their symmetries, we can now proceed to define G-CNN layers. These are linear or non-linear maps between representation spaces that respect their structure and symmetries. We will begin by giving definitions, and then proceed with examples to clarify them.

Definition 8.1.2 (Equivariant Layer)**.** *An equivariant layer with input representation space (P_1, S_1, ρ_1) (Def. 8.1.1) and output representation space (P_2, S_2, ρ_2) consists of*

1. *A group homomorphism $\chi : S_2 \to S_1$ relating symmetries in the output space to symmetries in the input space.*

2. *A space $\mathcal{H}$ of S_2-equivariant maps*

$$\Phi : \Gamma(A_1) \to \Gamma(A_2), \tag{8.8}$$

where $A_i = P_i \times_{\rho_i} V_i$ for $(i = 1, 2)$ and $\Gamma(A_i)$ is the input/output representation space. In this context, S_2-equivariance means that for all $s \in S_2$ and $\Phi \in \mathcal{H}$,

$$\Phi \chi(s) = s\, \Phi. \tag{8.9}$$

In the simplest case, $P_1 = P_2 = P$, $S_1 = S_2 = S$ and $\chi = \mathrm{id}_S$, but as we will discuss in Section 8.2, allowing for changes in the principal bundle and symmetry group enables many useful network design patterns.

In many cases, particularly when the symmetry group S includes gauge transformations, one would like to specify how the principal bundles P_2 and P_1 are related as well. This is done using a principal bundle morphism $\eta : P_2 \to P_1$ (e.g. an embedding), which tells us for each point $x \in B_2$ and each frame $p \in \pi_2^{-1}(x)$ at x to which point $y = \pi_1(\eta(p)) \in B_1$ and frame $\eta(p) \in \pi_1^{-1}(y)$ it correponds. This correspondence should be compatible with the group homomorphism χ, as in the following definition.

Definition 8.1.3 (Equivariant Linked Layer)**.** *An equivariant linked layer with input representation space (P_1, S_1, ρ_1), output representation space (P_2, S_2, ρ_2) and linking morphism η consists of*

1. *A principal bundle morphism $\eta : P_2 \to P_1$ relating the positions and frames in the output space to positions and frames in the input space.*

2. *A group homomorphism $\chi : S_2 \to S_1$ relating symmetries in the output space to symmetries in the input space. This map must be compatible with the linking morphism η in the following sense:*

$$\eta \circ s = \chi(s) \circ \eta \tag{8.10}$$

for all $s \in S_2$. This may also be written in full detail as $\eta \circ \alpha^2_s = \alpha^1_{\chi(s)} \circ \eta$, where $\alpha^i : S_i \to \mathrm{Aut}(P_i)$ (for $i = 1, 2$) are part of the definition of the input and output representation space (Def. 8.1.1).

3. *A space $\mathcal{H}$ of S_2-equivariant maps*

$$\Phi : \Gamma(A_1) \to \Gamma(A_2), \tag{8.11}$$

In this context, equivariance means that for all $s \in S_2$ and $\Phi \in \mathcal{H}$,

$$\Phi\chi(s) = s\,\Phi. \tag{8.12}$$

We note that in case the symmetry group S consists of gauge transformations, one will want to choose in addition a *principal connection* compatible with η (see Chapter 10). We will now proceed to clarify the meaning of and motivation for these definitions using examples.

8.2 Discussion & Examples

8.2.1 Simple examples

The simplest way to instantiate a G-CNN layer is to let the principal bundle $P_1 = P_2 = P$ be the same for the input and output representation space, and likewise let $S_1 = S_2 = S$. Then we may choose $\chi = \mathrm{id}_S$, so that a layer is nothing more than a space of maps $\Phi : \Gamma(A_1) \to \Gamma(A_2)$ satisfying $\Phi s = s\Phi$ (here $s \in S$, A_i are defined as in Definition 8.1.2). Indeed, we can take it to be a linked layer by choosing $\eta : P_2 \to P_1$ to be the identity as well: $\eta = \mathrm{id}_P$. The compatibility condition Eq. 8.10 is satisfied because $\eta \circ s = s = \chi(s) = \chi(s) \circ \eta$ in this case. The following examples are of this kind.

Example 8.2.1 (Translation equivariant planar convolution layer)**.** *The classical convolution layer used in CNNs can be described as follows. For both the input and output space we choose the representation space described in Example 8.1.1, i.e. the space of planar images / feature maps with translation symmetry. That is, for both*

input ($i = 1$) and output ($i = 2$) space we let P_i be the frame bundle of $B_i = \mathbb{R}^2$ with structure group $G_i = \mathrm{GL}(2, \mathbb{R})$ (or a reduction theoreof) and $S_i = \mathbb{R}^2$ acting on P_i by translation. For ρ_i, V_i we use the trivial representation $\rho_i(g) = I_{n_i}$ on $V_i = \mathbb{R}^{n_i}$, where n_i is the number of channels in representation space i. Feature maps are then described as sections $f \in \Gamma(A_i)$ of $A_i = P_i \times_{\rho_i} V_i$, which are just functions $f : \mathbb{R}^2 \to \mathbb{R}^{n_i}$ because $A_i \simeq B_i \times V_i$ are trivial bundles. For details, see Ex. 8.1.1.

According to definition 8.1.2, an equivariant layer consists of a group homomorphism $\chi : S_2 \to S_1$ and a space $\mathcal{H}$ of S_2 equivariant maps $\Phi : \Gamma(A_1) \to \Gamma(A_2)$. In the present example, we have $S_2 = S_1 = \mathbb{R}^2$, so the obvious way of relating symmetries in the output space to symmetries in the input space is to choose $\chi = \mathrm{id}_{\mathbb{R}^2}$. Similarly, as mentioned before, we may also choose $\eta = \mathrm{id}_P$.

Consider now a linear layer. For $\mathcal{H}$ we choose the space of all *equivariant linear maps $\Phi : \Gamma(A_1) \to \Gamma(A_2)$. In the following chapters, we will study the question of universality: whether a certain kind of linear layer is general enough to allow any equivariant linear map between certain representation spaces to be learned. In the present case, it is well known that convolutions or cross-correlations are universal, in that any equivariant linear operator Φ can be written as:*

$$(\Phi f)(x) = \int_{\mathbb{R}^2} k(y - x) f(y) dy, \tag{8.13}$$

where $k : \mathbb{R}^2 \to \mathbb{R}^{n_2 \times n_1}$ is an unconstrained matrix-valued kernel. The kernel k and operator Φ are in one-to-one correspondence. In CNNs, it is common to limit the support of k to a small region, and to discretize the input space so that k is represented as an array of learnable parameters of shape $w \times h \times n_2 \times n_1$ (where w and h are the width and height of the kernel support). In the discrete case, the integral above is then replaced by a sum.

Non-linear layers will be considered in Section 8.2.9.

Example 8.2.2 (Roto-translation equivariant planar G-CNN)**.** *A typical roto-translation equivariant (steerable) planar G-CNN (Chapter 3; Worrall et al., 2017; Weiler et al., 2018a; Weiler and Cesa, 2019), can be described as follows. For the input space, we take a space of planar images with rotation and translation symmetries (Example 8.1.2). That is, let P_1 be the frame bundle of $B_1 = \mathbb{R}^2$ with structure group $G_1 = \mathrm{GL}(2, \mathbb{R})$ (or a reduction to $G_1 = \mathrm{SO}(2)$) and $S_1 = \mathrm{SE}(2)$, the group of rotations and translations in 2D. If the input is a grayscale image, we would take the trivial representation $\rho_1(g) = 1$ and for an RGB image we would take $\rho_1(g) = I_3$ (three copies of the trivial representation).*

For subsequent representation spaces i we will use the same principal bundle and symmetry group, $\chi = \mathrm{id}_S$ and $\eta = \mathrm{id}_P$, but a potentially different representation ρ_i

of G. Thus, $\Gamma(A_i)$ is the space of ρ_i fields over B. The symmetry group $\mathrm{SE}(2)$ *acts on the representation space $\Gamma(A_i)$ via the induced representation $\pi_i = \mathrm{ind}_{\mathrm{SO}(2)}^{\mathrm{SE}(2)} \rho_i$. Intuitively, this means that π_i will move fibers around according to the action of* $\mathrm{SE}(2)$ *on $B = \mathbb{R}^2$ and simultaneously act within the fibers by the representation ρ_i of* $\mathrm{SO}(2)$.

As discussed in the examples of representation spaces, one obtains Harmonic Networks by choosing ρ_i to be block-diagonal with irreducible representations as blocks (see Example 8.1.4). Similarly, one obtains a regular G-CNN similar to the one presented in Chapter 2 but for continuous rotations by choosing ρ_i to be the regular representation of $G = \mathrm{SO}(2)$ on the space of band-limited functions sampled on a finite set of points.

Since we chose $\chi = \mathrm{id}_S$ and $\eta = \mathrm{id}_P$, to define a layer all we need to do is choose the space $\mathcal{H}$ of equivariant maps. For a linear layer, we may choose the space of all equivariant linear maps between the input and output representation space. That is, $\mathcal{H} = \hom_{\mathrm{SE}(2)}(\Gamma(A_1), \Gamma(A_2))$. Since $\Gamma(A_i)$ are induced representations, we are really asking about the space of equivariant linear maps (intertwiners) between induced representations. This is a fundamental topic, studied by the founder of representation theory Frobenius (for finite groups), and Mackey (for locally compact groups).

For the present example it can be shown that any linear map $\Phi \in \mathcal{H}$ can be written as a cross-correlation with a constrained kernel. That is, we can write Φ as

$$(\Phi f)(x) = \int_{\mathbb{R}^2} k(y - x) f(y) dy, \tag{8.14}$$

where $f \in \Gamma(A_1)$ and the kernel satisfies for all $r \in \mathrm{SO}(2), x \in \mathbb{R}^2$:

$$k(rx) = \rho_2(r) k(x) \rho_1(r^{-1}). \tag{8.15}$$

It can be shown that for the respective representations (irreducible or regular), the weight sharing schemes of Harmonic Networks and Regular G-CNNs arise from this constraint (see e.g. Weiler et al., 2018a).

8.2.2 Equivariance & Compositional Properties

We know that in a vanilla G-CNN with the same principal bundle P and symmetry group S in each representation space, equivariance is transitive: if each of the layers Φ_l satisfies $s\Phi_l = \Phi_l s$ for all $s \in S$, then the same is true for their composition $\Phi = \Phi_l \circ \cdots \circ \Phi_1$. So the composition of equivariant maps is an equivariant map. A neural network made of equivariant layers is equivariant.

The general definition of G-CNN layer (8.1.2 and 8.1.3), with a potentially different symmetry group and principal bundle at each layer, was chosen to have a similar property. Specifically, if the layers Φ_l satisfy $\Phi_l \chi_l(s) = s\Phi_l$ for all $s \in S_{l+1}$, then the same is true for their composition: $\Phi\chi(s) = s\Phi$ (where $\chi = \chi_l \circ \cdots \circ \chi_1$ and $\Phi = \Phi_l \circ \cdots \circ \Phi_1$). So the composition of equivariant layers is still equivariant. The only difference is that now composite map Φ is equivariant with respect to the group S_{l+1} which is associated with the output space. This group acts on the output space via principal bundle automorphisms $\alpha_s^{l+1} : S_{l+1} \to \mathrm{Aut}(P_{l+1})$ (where α^l is part of the data used to define representation space l – see Def. 8.1.1), and on the input space via $\alpha^1 \circ \chi : S_{l+1} \to \mathrm{Aut}(P_1)$.

If the layers are all linked (Def. 8.1.3), i.e. we have principal bundle morphisms $\eta_l : P_l \to P_{l-1}$, then we can form the composite $\eta = \eta_l \circ \cdots \circ \eta_1$ which is itself a bundle morphism $\eta : P_{l+1} \to P_1$. Moreover, if for each l we have $\chi_l(s) \circ \eta_l = \eta_l \circ s$ for $s \in S_{l+1}$ (compatibility, Eq. 8.10) then the same is true for their composition: $\chi(s) \circ \eta = \eta \circ s$ for $s \in S_{l+1}$.

To summarize, a composition of equivariant layers (Def. 8.1.2) is an equivariant layer, and the composition of equivariant linked layers (Def. 8.1.3) is an equivariant linked layer.

8.2.3 Symmetry breaking & reduction of the structure group

It is often desirable to break some of the symmetries at higher layers of the network (which correspond to greater length scales). For instance, we may want to have a planar G-CNN with lower layers that are E(2)-equivariant, followed by layers that are SE(2)-equivariant, followed by some layers that are only $\mathbb{R}^2$-equivariant. This can be achieved in our framework by letting η be a reduction of the structure group, and χ a subgroup inclusion.

This is one of the reasons why the maps η_l and χ_l are defined as they are, pointing from representation space $l+1$ to l. If we had defined them to point from l to $l+1$, then the whole network must be S_1-equivariant, and it would be impossible to break symmetry. It is possible to define a morphism $\tilde{\chi}_l : S_l \to S_{l+1}$ from a group S_l to a subgroup S_{l+1}, but we would still have a (perhaps trivial) action of S_l on all subsequent layers, which means that the rest of the network is S_l-equivariant. Instead what we want is for every symmetry in S_l to have an interpretation in lower layers, but not necessarily the other way around.

Another problem with having morphisms $\tilde{\chi} : S_l \to S_{l+1}$ from a group S_l to a subgroup S_{l+1}, is that there may not be any surjective or even non-trivial such morphisms. For instance, if $S_l \simeq \mathrm{SE}(2)$ and $S_{l+1} \simeq \mathbb{R}^2$ one might consider

mapping roto-translation $(r,t) \in \mathrm{SE}(2)$ to translation $t \in \mathbb{R}^2$, but this is not a homomorphism because $(r,t)(r',t') = (rr', t+rt')$ but $t+t' \neq t+rt'$. In this case, no surjective homomorphisms exist, and the only morphism available is $\tilde{\chi}(r,t) = e$. If instead we have $\chi_l : S_{l+1} \to S_l$ pointing from a subgroup S_{l+1} to a group S_l, we always have a good (i.e. injective) homomorphism (the subgroup inclusion).

This motivates the definition of χ_l, but what about η_l? As mentioned, we can use the notion of a reduction of the structure group for symmetry breaking. Recall from Definition 7.2.15 that a princpal G_{l+1}-bundle $P_{l+1} \to B_{l+1}$ is a reduction of the structure group of the principal G_l-bundle $P_l \to B_l$ if they have the same base space $B_l = B_{l+1} = B$ and there exists a principal bundle B-morphism $\eta_l : P_{l+1} \to P_l$ over θ_l, where $\theta_l : G_{l+1} \to G_l$ is an injective homomorphism. That is, η_l should satisfy $\eta_l(pg) = \eta_l(p)\theta_l(g)$ for all $p \in P_{l+1}$ and $g \in G_{l+1}$, and $\pi \circ \eta_l = \pi$.

Note that we can only have a principal bundle morphism over θ if we have a group homomorphism θ, and here the same argument we made about symmetry groups S applies: if G_{l+1} is a subgroup of G_l, then there may not exist surjective homomorphisms from G_l to G_{l+1}, whereas there always exists an injective homomorphism from G_{l+1} to G_l. Do note however that even if such an injective homomorphism exists, there may be a topological obstruction to the existence of a reduction of the structure group. For instance, a reduction to $G_{l+1} = \{e\}$ is tantamount to a smooth global choice of frame, which does not exist unless the principal bundle is trivial (Theorem 7.2.3). Similarly, a reduction from $G_l = \mathrm{O}(d)$ to $G_{l+1} = \mathrm{SO}(d)$ amounts to a choice of orientation, which does not exist on non-orientable manifolds such as the Mobius band or Klein bottle.

So it is clear that if we want to reduce or break some of the symmetries from layer l in layer $l+1$, we need a morphism η_l pointing from the bundle with smaller structure group at $l+1$ to the one with bigger group at l. Although we do not currently know how deep this analogy goes, we note that this usage of a reduction of the structure group is consistent with its use in physics as a way of describing spontaneous symmetry breaking (Rudolph and Schmidt, 2017, Section 7.3, The Higgs Mechanism).

Example 8.2.3 (Reduction from O(2) to SO(2))**.** *We will define input and output representation spaces and a layer between them. The input space is defined by* $B_1 = \mathbb{R}^2$, $P_1 = F_{\mathrm{O}}(\mathbb{R}^2)$, $G = \mathrm{O}(2)$, $S = \mathrm{E}(2)$ *and* ρ_1 *some representation of* $\mathrm{O}(2)$. *The output space is defined by* $B_2 = \mathbb{R}^2$, $P_2 = F_{\mathrm{SO}}(\mathbb{R}^2)$, $G = \mathrm{SO}(2)$, $S = \mathrm{SE}(2)$ *and* ρ_2 *some representation of* $\mathrm{SO}(2)$. *These data determine the associated bundles* $A_i = P_i \times_{\rho_i} V_i$ *and representation spaces* $\Gamma(A_i)$ *for* $i = 1, 2$.

To define a layer, we first need to choose a morphism $\chi : \mathrm{SE}(2) \to \mathrm{E}(2)$. *Let us*

choose the subgroup inclusion. Similarly, let $\eta : F_{\mathrm{SO}}(\mathbb{R}^2) \to F_{\mathrm{O}}(\mathbb{R}^2)$ *be the inclusion of* $F_{\mathrm{SO}}(\mathbb{R}^2)$ *into* $F_{\mathrm{O}}(\mathbb{R}^2)$*. This is a B-morphism, for* $\pi_{F_{\mathrm{O}}(\mathbb{R}^2)} \circ \eta = \pi_{F_{\mathrm{SO}}(\mathbb{R}^2)}$*. Moreover,* η *satisfies* $\eta(pg) = \eta(p)\theta(g)$ *for* $p \in F_{\mathrm{SO}}(\mathbb{R}^2)$ *and* $g \in \mathrm{SO}(2)$ *where* $\theta : \mathrm{SO}(2) \to \mathrm{O}(2)$ *is the subgroup inclusion. Thus* η *is a reduction of the structure group.*

Since $F_{\mathrm{SO}}(\mathbb{R}^2)$ *is a reduction of* $F_{\mathrm{O}}(\mathbb{R}^2)$*, the associated bundle* $A_1 = F_{\mathrm{O}}(\mathbb{R}^2) \times_{\rho_1} V$ *is isomorphic to* $A_1' = F_{\mathrm{SO}}(\mathbb{R}^2) \times_{\rho_1'} V$*, where* $\rho_1' = \rho_1 \circ \theta$ *is a representation of* SO(2) *(see 1.6.7 of (Rudolph and Schmidt, 2017)). What this means is that we may simply reinterpret a section* $f : \mathbb{R}^2 \to V$ *of* A_1 *as a section of* A_1'*, which like* A_2 *has structure group* SO(2)*. Hence, the space* $\mathcal{H}$ *of equivariant linear maps from* $\Gamma(A_1)$ *to* $\Gamma(A_2)$ *is equivalent to the one studied in Example 8.2.2 where the input and output space use the same structure group* $G = \mathrm{SO}(2)$*.*

In the above example, a reduction η from O(2) to SO(2) exists because the base space $\mathbb{R}^2$ is orientable (or we can say that it is orientable because such a reduction exists). In this case we can go further and reduce the structure group to the trivial group $G = \{e\}$. In the case of the sphere, which can be endowed with a metric, and is orientable but not parallelizable, we can reduce the frame bundle from GL(2) to O(2) to SO(2), but not to $\{e\}$.

8.2.4 Lift of Structure Group

We have seen how the definition of G-CNN allows symmetry breaking by letting η be a reduction of the structure group. It is also possible to let η be a lift of the structure group (Definition 7.2.16). That is, we let $\eta_l : P_{l+1} \to P_l$ be a B-morphism over a *surjective* homomorphism $\theta_l : G_{l+1} \to G_l$.

In physics, a lift of structure group of the orthogonal frame bundle from $G_l = \mathrm{SO}(d)$ to $G_{l+1} = \mathrm{spin}(d)$ via the double cover $\theta_l : \mathrm{spin}(d) \to \mathrm{SO}(d)$ is called a *spin structure*. A spin structure does not always exist, but the conditions under which it does are known (Rudolph and Schmidt, 2017, Section 5.4). Spin structures do exist for all orientable manifolds of dimension 3 or less, which includes most of the examples of interest. For example, if we consider the principal SO(2) bundle $\pi : \mathrm{SO}(3) \to S^2$ (spherical CNNs), it is possible to lift to the Hopf bundle $\pi : S^3 \to S^2$ via the double cover $\theta : \mathrm{spin}(2) \to \mathrm{SO}(2)$ (in this case, there is an accidental isomorphism $\mathrm{spin}(2) \simeq \mathrm{SO}(2)$).

Spin structures have found applications in the mathematical description of so-called pinwheel structures in the V1 area of the visual cortex (Petitot, 2017, Section 4.8). Whether they are useful in G-CNNs we currently do not know.

8.2.5 Change of base space

Another option allowed by the definition is to change the base space from one layer to the next. That is, the principal bundle morphism $\eta_l : P_{l+1} \to P_l$ over θ projects down to a map $\beta : B_{l+1} \to B_l$ and this map may not be a diffeomorphism. In most cases, one will want to have β, θ and hence η be at least embeddings though. In this case we can think of P_{l+1} as being isomorphic to its image $\eta(P_{l+1}) \subset P_l$ (Rudolph and Schmidt, 2012, Remark 1.1.8). That is, we can think of P_{l+1} as a sub-bundle embedded in P_l via η.

Example 8.2.4 (Vertical Pooling)**.** *Consider a translation equivariant planar CNN (Example 8.1.1), with $B_l = \mathbb{R}^2$, $P_l = F_{\mathrm{GL}}(\mathbb{R}^2)$ and $S_l = \mathbb{R}^2$ for some l. We will implement a global pooling in the y-direction. The output representation space has $B_{l+1} = \mathbb{R}$, $P_{l+1} = F_{\mathrm{GL}}(\mathbb{R})$ and $S_{l+1} = \mathbb{R}$ acting by translations.*

To define the layer, we embed $\mathbb{R}$ in $\mathbb{R}^2$ by mapping it to the x-axis. That is, we define $\beta : \mathbb{R} \to \mathbb{R}^2$ by $\beta(a) = (a, 0)$. This map naturally lifts to a principal bundle morphism $\eta : F_{\mathrm{GL}}(\mathbb{R}) \to F_{\mathrm{GL}}(\mathbb{R}^2)$. Translation symmetries are mapped in the obvious way by $\chi(a) = (a, 0)$. We can now define $\Phi_l : \Gamma(A_l) \to \Gamma(A_{l+1})$ by integrating the input signal $f : \mathbb{R}^2 \to V$ along the y-axis. Clearly this map is equivariant and linear. More generally, Φ_l could be a convolution in the x-direction with an arbitrary filter defined on $\mathbb{R}^2$.

We can go in the other direction, and extend a signal on $\mathbb{R}$ to one on $\mathbb{R}^2$ by repeating:

Example 8.2.5 (Vertical Repeat)**.** *Take P_l the frame bundle of $B_l = \mathbb{R}$ and P_{l+1} the frame bundle of $\mathbb{R}^2$. The map $\eta_l : F_{\mathrm{GL}}(\mathbb{R}^2) \to F_{\mathrm{GL}}(\mathbb{R})$ can be defined as the unique lift of the projection $\beta : \mathbb{R}^2 \to \mathbb{R}$ sending $(x, y) \mapsto x$. Symmetries $S_{l+1} = \mathbb{R}^2$ are similarly mapped by $\chi_l(x, y) = x$. As map Φ we can take $(\Phi f)(x, y) = f(x)$. Alternatively, we can take $(\Phi f)(x, y) = \int_{\mathbb{R}} f(z)\psi(z - x)dz$.*

Another example of an embedding:

Example 8.2.6 (Sphere to Cylinder)**.** *Consider a spherical CNN (Chapter 4 and Examples 8.1.8 and 8.1.9), with principal bundle $P_l = \mathrm{SO}(3)$, $B_l = S^2$, $G_l = \mathrm{SO}(2)$ and $S_l = \mathrm{SO}(3)$. For the next layer, one could choose as base space B_{l+1} a cylinder of height and radius 1. We can embed this cylinder into the sphere, e.g. by wrapping it around the Z-axis. Then, we can use $S_{l+1} = \mathrm{SO}(2)$ (rotations around the Z-axis) as symmetry group, and map this group homomorphically to the $\mathrm{SO}(2)$ subgroup of $\mathrm{SO}(3)$ corresponding to Z-axis rotations. The resulting panoramic images on the cylinder can be processed in a fast way, because convolution on the cylinder is just*

planar convolution with cyclic boundary conditions along one of the two axes. One could even do this for three cylinders centered on the X, Y, and Z axes, process the cylindrical feature maps independently, and then combine them again in a way that respects discrete cubic symmetries.

Example 8.2.7 (Universal Cover)**.** *Take P_l to be the frame bundle of the circle S^1, with symmetries $S_l = \mathrm{SO}(2)$. For the next representation space, we take $P_{l+1} = \mathbb{R}$ and $S_{l+1} = \mathbb{R}$ acting by translations. Define the morphism $\eta_l : F(\mathbb{R}) \to F(S^1)$ in the obvious way, by wrapping the line around the circle. We take as group homomorphism $\chi_l : \mathbb{R} \to \mathrm{SO}(2)$ the map $\chi_l(x) = x \mod 2\pi$. The layer Φ_l could be defined as mapping a feature map $f : S^2 \to V$ to a periodic function $(\Phi_l f)(x) = f(x \mod 2\pi)$.*

8.2.6 Layers without principal bundle morphisms

The requirement that adjacent representation spaces are linked by a principal bundle morphism $\eta : P_{l+1} \to P_l$ precludes certain constructions, which is why we have also included a definition of equivariant layers without linking morphism (Definition 8.1.2).

Example 8.2.8 (Map between Homogeneous Representation Spaces)**.** *Consider a Lie group H and two closed subgroups G_1, G_2 of H. As in example 8.1.10, we construct representation spaces with principal bundle $H \to H/G_i$ with structure group G_i and as base space the homogeneous space H/G_i, and associated bundles $A_i = H \times_{\rho_i} V_i$ for some representations ρ_i of G_i.*

For example, we may take $H = \mathrm{SE}(3)$, $G_1 = \mathrm{SO}(3)$, $G_2 = \mathbb{R}^3$. Thus, in the input space ($i = 1$) we have $B_1 = \mathrm{SE}(3)/\mathrm{SO}(3) \simeq \mathbb{R}^3$ and for the output space we have $B_2 = \mathrm{SE}(3)/\mathbb{R}^3 \simeq \mathrm{SO}(3)$.

Since the group $S_i = H = \mathrm{SE}(3)$ acts on both the input and output space, we choose $\chi = \mathrm{id}_S$. The equivariance constraint is then simply $\Phi s = s\Phi$, for $s \in S$ and $\Phi \in \mathcal{H}$. Thus, we can define an equivariant layer between these representation spaces (Definition 8.1.2).

In order to define an equivariant linked layer (Definition 8.1.3), we need a linking morphism η. Being a principal bundle morphism, η defines a correspondence between the points in and frames on H/G_2 with points in and frames on H/G_1. As this example shows, there is in general no natural way to do this (to what point $x \in \mathbb{R}^3$ should we map a rotation $r \in \mathrm{SO}(3)$?). This example shows that one can think of equivariant layers that do not fit the definition of equivariant linked layer, but also suggests that such layers tend to be geometrically less natural.

8.2.7 Group homomorphism from principal morphism

Consider the case where the symmetry groups S_1 and S_2 of input and output space are gauge transformations. The definition of linked layer calls for a principal bundle morphism $\eta : P_2 \to P_1$ and a compatible group homomorphism $\chi : S_2 \to S_1$. This principal bundle morphism η provides a correspondence between frames in the output space and frames in the input space, while the group homomorphism χ provides a correspondence between gauge transformations, i.e. changes of frame. This suggests that we should be able to obtain a compatible homomorphism χ from η.

Since gauge transformations act locally, it suffices to restrict attention to the case of trivial bundles. Let $P_1 = B_1 \times G_1$ and $P_2 = B_2 \times G_2$ be trivial principal bundles. Let $\eta : P_2 \to P_1$ be a principal bundle morphism over $\theta^\eta : G_2 \to G_1$ and $\beta^\eta : B_2 \to B_1$. Let $\alpha : P_2 \to P_2$ be a principal (B, G_2)-morphism, i.e. a gauge transformation. By Proposition 8, these take the form

$$\begin{aligned} \eta(x,g) &= (\beta^\eta(x), g^\eta(x)\theta^\eta(g)) \\ \alpha(x,g) &= (x, g^\alpha(x)g). \end{aligned} \tag{8.16}$$

We want to define χ in terms of η, such that $\eta \circ \alpha = \chi(\alpha) \circ \eta$. Let us write out the two sides of this equation. For the left hand side we have:

$$\begin{aligned} \eta \circ \alpha(x,g) &= \eta(x, g^\alpha(x)g) \\ &= (\beta^\eta(x), g^\eta(x)\theta^\eta(g^\alpha(x)g)) \\ &= (\beta^\eta(x), g^\eta(x)\theta^\eta(g^\alpha(x))\theta^\eta(g))) \end{aligned} \tag{8.17}$$

For the right hand side:

$$\begin{aligned} \chi(\alpha) \circ \eta(x,g) &= \chi(\alpha)(\beta^\eta(x), g^\eta(x)\theta^\eta(g)) \\ &= (\beta^\eta(x), g^{\chi(\alpha)}(\beta^\eta(x))g^\eta(x)\theta^\eta(g)) \end{aligned} \tag{8.18}$$

Setting these two to be equal, and multiplying by $(g^\eta(x)\theta^\eta(g))^{-1}$ on the right, we obtain:

$$g^{\chi(\alpha)}(\beta^\eta(x)) = g^\eta(x)\theta^\eta(g^\alpha(x))g^\eta(x)^{-1} \tag{8.19}$$

If β^η is not injective, the constraint may not be satisfiable. If β^η is a diffeomorphism we can characterize $g^{\chi(\alpha)}$ (and thus $\chi(\alpha)$) in terms of η and α:

$$g^{\chi(\alpha)}(x) = g^\eta(\beta^{-1}(x))\theta^\eta(g^\alpha(\beta^{-1}(x)))g^\eta(\beta^{-1}(x))^{-1} \tag{8.20}$$

More generally when β is an injective immersion, this equation holds for $x \in B_1$ in the image of $\beta : B_2 \to B_1$

8.2.8 Combination layers and computation graphs

The layers defined so far all take as input a field and produce as output another field. We can compose such layers to produce a sequential / feed-forward G-CNN. In modern deep learning, neural networks often take the form of a general computation graph, i.e. a directed acyclic graph (DAG) with feature spaces as nodes and layers as arrows. So a truly general definition of G-CNN needs to include layers that take multiple inputs. We call this a combination layer.

An easy combination layer is the analog of "channel-wise stacking of feature maps": the direct sum of vector bundles. This takes two vector bundles over the same base space and returns one whose fiber is the direct sum of the fibers of the input bundles. Similarly, one can define a tensor product of bundles over the same base. As discussed in Section 8.2.9, this is a nice way to implement a number of different kinds of nonlinearities, such as gating, tensor product nonlinearities, and (self)-attention. More general constructions are possible as well, but we will leave those for future work.

Another example of a combination layer is pointwise addition of sections (i.e. feature maps). That is, we can add two sections $\sigma, \sigma' \in \Gamma(A)$ of the same associated vector bundle A. This allows us to implement residual networks as $\Phi(\sigma) + \sigma$, which works for any map $\Phi : \Gamma(A) \to \Gamma(A)$ that maps a section space to itself.

8.2.9 Equivariant Nonlinearities

A G-CNN is equivariant if and only if all layers are equivariant, including the non-linear ones. In this section we discuss a number of equivariant nonlinearities that have been defined in the literature.

The simplest case, discussed in Chapter 2, is the Regular G-CNN, where we can use any pointwise non-linearity just like in a conventional neural network. In this case we have a base space B (typically a homogeneous space) with group action of S (called G in earlier chapters), and scalar feature maps $f : B \to \mathbb{R}$ (if we have multiple scalar channels, we have $f : B \to \mathbb{R}^n$). The symmetries $g \in S$ act via the regular representation: $(L_g f)(x) = (f \circ g^{-1})(x) = f(g^{-1}x)$. The application of pointwise non-linearities $\nu : \mathbb{R} \to \mathbb{R}$ is modelled as an operator $(C_\nu f)(x) = (\nu \circ f)(x) = \nu(f(x))$. Since composition on the left and right commute, we have $L_g C_\nu = C_\nu L_g$, so C_ν is equivariant for any ν.

When we consider non-scalar fields, this approach no longer works because we now have the structure group G acting on the fiber via a non-trivial representation ρ. In these cases, one can use fiber-wise nonlinearities. Typically,

ρ is taken to be a block-diagonal matrix (i.e. a reduced representation), in which case the task of finding fiber-wise nonlinearities is reduced to finding non-linearities for the blocks (e.g. a irreducible representations).

If ρ is unitary / orthogonal (norm preserving), one can write a feature vector in polar coordinates and apply an arbitrary nonlinearity to the radius. For instance, Worrall et al., 2017 define a nonlinearity for complex scalars,

$$\mathbb{C}\text{ReLU}_b(re^{i\phi}) = \text{ReLU}(r-b)e^{i\phi}, \tag{8.21}$$

where $\text{ReLU}(x) = \max(0, x)$ and b is a positive real number.

In addition to elementwise nonlinearities, many popular neural network architectures contain multiplicative interactions in the form of plain products, gating, or attention (Jayakumar et al., 2020). Early examples include higher order Boltzmann machines (Sejnowski, 1986; Memisevic and Hinton, 2007) and LSTMs (Hochreiter and Schmidhuber, 1997). More recently, attention and self-attention have become very popular, both of which involve multiplicative interactions in an essential manner (Bahdanau et al., 2014; Vaswani et al., 2017). Channel-wise attention / gating, known as squeeze and excite networks, has also been very successful (Hu et al., 2019).

In G-CNNs, multiplicative interactions can be implemented as *tensor products* of vector bundles. Roughly speaking, the fiber of the tensor product bundle is the tensor product of the fibers of the factors, and the representation acting on this fiber is the tensor product of representations of the factors. This idea has for instance been applied by (Kondor et al., 2018a), who define a spherical CNN whose nonlinearities are all tensor products. Such a network implements a class of polynomials that can be written in a certain factorized form.

A natural way to go beyond polynomials is by gating and attention. In the neural networks literature, a gate is a scalar that has been squashed by a nonlinearity such as sigmoid of tanh, so that it is contained in a limited range such as $[0, 1]$ or $[-1, 1]$. The gate is then multiplied by an arbitrary feature. Intuitively, a $[0, 1]$ gate can decide which features are relevant (gate close to 1) or unimportant (gate close to 0). More generally, one can produce a number of scalars and apply a softmax nonlinearity to create a categorical distribution over a finite set. The softmax output can then be used to weigh feature vectors associated with the elements of this set. This is called attention.

In the context of G-CNNs, gating and attention can be understood as follows. As discussed before, we can apply any nonlinearity (including squashing functions or softmax) to a collection of scalar features (i.e. features that transforms according to the trivial representation $\rho(g) = I$). The resulting quantity

is again a scalar. Then, we can apply a tensor product between this scalar gate and an arbitrary feature vector. If this feature vector transforms according to a representation ρ of G, then the tensor product with a scalar will still transform according to the same representation ρ. In this way, one can build equivariant gating and attention modules.

Gating and attention have already been applied in G-CNNs, for instance in (Weiler et al., 2018a; Diaconu and Worrall, 2019). Since gating can positively impact gradient flow (Hochreiter and Schmidhuber, 1997), and attention has been empirically very successful (Vaswani et al., 2017), we believe this to be a promising approach to equivariant network design.

9

Homogeneous G-CNNs

In this chapter we study homogeneous G-CNNs, which are a special case of the general constructs presented in the previous chapter. We will begin by defining homogeneous representation spaces and then study linear maps between such representation spaces. A homogeneous representation space is a space of fields over a homogeneous space[1]. The main result on linear maps between such spaces is that they can always be described as a generalized convolution or cross-correlation with an equivariant kernel. The space of equivariant kernels is itself shown to be a space of fields (sections of an associated vector bundle), which turns out to have a very rich and interesting structure.

[1] In practice we will work with coset spaces H/G which are homogeneous spaces with a choice of origin, but the distinction is not very important here and so we will speak of homogeneous spaces and homogeneous G-CNNs.

9.1 Summary of results

The definition of homogeneous representation space will be given in Section 9.2 and can be sketched as follows. Recall from Definition 8.1.1 that a representation space is defined in terms of a smooth[2] principal bundle P with base space B and structure group G, a symmetry group S and a representation (ρ, V) of G. This determines a space of fields $\Gamma(P \times_\rho V)$ over B. The principal bundle describes the space of frames P_x of V of a certain kind (oriented, orthogonal, etc.) at each point $x \in B$, with any two frames at x being related by a unique $g \in G$. The representation ρ determines the type of field, such as scalar, vector, tensor, irreducible, etc. The symmetry group S acts on the principal bundle P by automorphisms, and this induces an action on the associated bundle $P \times_\rho V$ and the space of fields $\Gamma(P \times_\rho V)$.

In the case of a homogeneous G-CNN, the principal bundle has as total space a *Lie group* P, structure Lie group $G \subset P$ and base space $B = P/G$. The symmetry group is $S = P$ (so P plays a dual role) which acts on P by left multiplication (these are shown to be principal bundle automorphims). The action of $S = P$ on the representation space $\Gamma(P \times_\rho V)$ is called the representation of S induced by the representation ρ of G.

In Section 9.3 we study linear maps (not necessarily equivariant) between homogeneous representation spaces (P, G_1, ρ_1) and (P, G_2, ρ_2). The input and output spaces may have a different structure group G_i, base space $B_i = P/G_i$, and representation ρ_i, but for simplicity we assume that the total space of the principal bundle as well as the symmetry group is P for both spaces. In the terminology of the previous chapter, we are thus considering linear layers (Definition 8.1.2) with $S_1 = S_2 = P$ and group homomorphism $\chi = \mathrm{id}_P$.

We restrict our attention to smoothing operators, i.e. linear operators between homogeneous representation spaces that can be written as an integral transform with a smooth two-argument kernel. The smooth two-argument kernels can be defined as fields on $B_2 \times B_1$ (Definition 9.3.3 and 9.3.4), and we show that the space of such fields is isomorphic to the space of smoothing operators as a vector space and representation (Theorems 9.3.1 and 9.3.2).

In Section 9.4 we impose an equivariance constraint on the smoothing operator, and show that this implies an invariance constraint on the kernel (Theorem 9.4.1). We then show (Theorem 9.4.2) that the space of invariant two-

[2] As before we assume all structures to be smooth, e.g. G and S are Lie groups, etc. Lie groups are locally compact, and this should be sufficient for most results, but we stick to the smooth setting for consistency with the other chapters in Part II.

argument kernels is isomorphic to the space of one-argument kernels (Definition 9.4.5), and that equivariant smoothing operators can be written as a cross-correlation with a one-argument kernel (Theorem 9.4.3 – "Convolution is all you need").

The one-argument kernels can be represented in lifted form, or locally as a collection of functions on pieces of the base space P/G_1 or the double coset space $G_2\backslash P/G_1$. For such a collection of maps to define a local kernel, it must satisfy certain constraints, which are elucidated in Theorems 9.6.1 and 9.6.2.

Although we present here an original synthesis, the mathematical results presented in this chapter are either known in one form or another or at least not surprising to experts. Specifically, intertwiners between induced representations were studied by Mackey, who obtained most of the fundamental results (Mackey, 1952; Mackey, 1953; Mackey, 1968). In the machine learning community, Kondor and Trivedi (2018) first studied the relation between convolution and equivariance using tools from non-commutative harmonic analysis. The main results on equivariance and convolution and the structure of the space of equivariant kernels presented in this chapter (first presented in (Cohen et al., 2018b; Cohen et al., 2018a)) can be understood as a generalization of the results of Kondor & Trivedi. Specifically, Kondor & Trivedi study regular representations (i.e. representations induced from a trivial representation $\rho(g) = 1$), whereas we study equivariant maps between general induced representations. On the other hand, Kondor & Trivedi present a number of interesting results on the spectral theory that are not included here.

9.2 Homogeneous Representation Spaces

In this section we will define homogeneous representation spaces.

9.2.1 The principal bundle of cosets

In a homogeneous G-CNN, the representation spaces (Definition 8.1.1) are associated with a homogeneous principal bundle P:

Definition 9.2.1 (Homogeneous Principal Bundle)**.** *Let P be a Lie group and $G \subset P$ a Lie subgroup. Then the projection*

$$\begin{aligned} P &\xrightarrow{\pi} P/G \\ p &\mapsto pG \end{aligned} \tag{9.1}$$

defines a principal bundle with total space P, base space $B = P/G$, structure group G and fibers pG. This is called a coset principal bundle or homogeneous principal bundle. The right action of G on the total space P is given by the group operation $p \mapsto pg$ (where $g \in G$ and $p \in P$).

The fibers of the projection are exactly the left cosets pG of G in P. That is, the fiber at $x = pG \in P/G$, denoted P_x, equals $pG \subset P$. Thus the coset pG plays two roles simultaneously: it is a *point* x in the base space $B = P/G$ and a *subset* of the principal bundle P or more specifically, it is the fiber $P_x = pG$.

Let us check that a homogeneous principal bundle is indeed a principal bundle. To verify that this is a G-bundle, we verify that $\pi(pg) = \pi(p)$. Indeed this is true becuase $pgG = pG$ by the closure axiom of a group G. Furthermore, the right action is transitive and free on cosets (fibers). Thus, $P \to P/G$ is indeed a principal G bundle.

In a homogeneous G-CNN, the representation spaces are defined with respect to a homogeneous principal bundle. In different layers one would typically use a principal bundle with the same total space P or a subgroup $P' \subset P$, and potentially different structure groups $G \subset P$ and base spaces P/G.

9.2.2 Symmetries

The definition of representation space (8.1.1) refers to a symmetry group S acting on the principal bundle P via principal bundle automorphisms. In a

homogeneous G-CNN, P is a group and S is chosen to equal P. The symmetry group S then acts on P by left multiplication, which defines an automorphism:.

$$\begin{aligned} \alpha_u &: P \to P & \beta_u &: P/G \to P/G \\ \alpha_u(p) &= up & \beta_u(pG) &= upG, \end{aligned} \tag{9.2}$$

where $u, p \in P$. Clearly, $\alpha_u(pg) = \alpha_u(p)g$ for $g \in G$ (i.e. α_u is right-equivariant as required for G-bundle morphisms). We further verify that $\pi \circ \alpha_u = \beta_u \circ \pi$:

$$\pi(\alpha_u(p)) = \pi(up) = (up)G = u(pG) = u\pi(p) = \beta_u(\pi(p)) \tag{9.3}$$

So α_u (left multiplication by u) is a principal G-bundle automorphism.

9.2.3 Induced representation

Recall that the representation space of a G-CNN is defined as the space of sections of an associated bundle, i.e. $\Gamma(P \times_\rho V)$, and further recal that symmetries S which act on P by principal bundle automorphisms, also naturally act on $\Gamma(P \times_\rho V)$ (see Definition 8.1.1). Here we study this space and action in lifted form (Section 7.2.4) for the special case of homogeneous principal bundles P with structure group G and symmetries $S = P$ acting on P by left translations. This representation, which describes the action of S on ρ-fields on P/G, is known as the induced representation (Kaniuth and Taylor, 2013).

Let (ρ, V) be a representation of G, let $P \to P/G$ be a homogeneous principal bundle, and let $A = P \times_\rho V$ be the associated vector bundle. Recall that a lifted section $f \in \hat{\Gamma}(A)$ is a map $f : P \to V$ satisfying $f(pg) = \rho^{-1}(g)f(p)$ for $p \in P$ and $g \in G$ (Theorem 7.2.4). As for any principal bundle, automorphisms α_u act on $\hat{\Gamma}(A)$ via left translation. For the case of a homogeneous principal bundle $P \to P/G$, this action is known as the representation of S induced by the representation ρ of G, denoted $\lambda = \mathrm{ind}_G^S \rho$:

$$(\lambda(u)f)(p) = f(u^{-1}p). \tag{9.4}$$

It is easy to verify that $\lambda(u)f \in \hat{\Gamma}(A)$ for $u \in S$ and $f \in \hat{\Gamma}(A)$, and that $\lambda(uu') = \lambda(u)\lambda(u')$, so λ is indeed a representation of S in $\hat{\Gamma}(A)$. Since $\hat{\Gamma}(A)$ and $\Gamma(A)$ are isomorphic, we immediately obtain an equivalent representation in $\Gamma(A)$, but we defer discussion of non-lifted forms of the induced representation until Section 9.5.3.

9.2.4 Definition of Homogeneous Representation Space

As per Definition 8.1.1, a G-CNN representation is defined in terms of a principal G-bundle P, a group of symmetries S acting on P via automorphisms, and a linear representation (ρ, V) of G. In a homogeneous G-CNN, we choose these as follows. For the principal bundle P, we choose a homogeneous principal bundle $P \to P/G$ (Definition 9.2.1). For the group of symmetries, we choose $S = P$, acting by automorphisms on the principal bundle P (Section 9.2.2). The representation ρ of G, which determines the type of field, may be chosen freely.

Together these data determine the associated bundle $A = P \times_\rho V$, the space $\Gamma(A)$ (ρ-fields over P/G), and the action of S on $\Gamma(A)$ (the induced representation; Section 9.2.3).

9.3 Homogeneous Linear Layers

In this section we study linear maps between homogeneous representation spaces. We consider those maps that can be written as an integral transform using a smooth kernel – so called smoothing operators. We show that the kernel of this transform should be interpreted as a section of a certain associated bundle, called the bundle of two-argument kernels. We do not yet introduce an equivariance constraint; that is left to Section 9.4.

9.3.1 Setup, scope and assumptions

For this and following sections, we work with the following setup. We are given two homogeneous representation spaces $\Gamma(A_n)$, corresponding to the input ($n = 1$) and output ($n = 2$) of a linear layer (see Def. 8.1.1 & 8.1.2). We denote the principal bundle, structure group, base space and representation corresponding to these spaces by P_n, G_n, $B_n = P_n/G_n$, and (ρ_n, V_n), respectively. The two representation spaces have symmetries S_n, but these do not play a role in this section. We will sometimes suppress the index n when it does not matter, and simply write P, G, etc. In Section 9.4, we will further assume that $P_1 = P_2 = S$.

We assume that G_n is a closed Lie subgroup of P_n, so that P_n/G_n is a smooth manifold by the quotient manifold theorem (Lee, 2003). In some theorems, where we need to integrate over G_n, we also assume that G_n is compact (this is required when working with lifted sections, but these theorems can usually be proven in non-lifted form, even when G_n is not compact).

Compactly supported feature maps

Recall that a feature map is modelled as a section $\sigma \in \Gamma(A)$, and that such sections can be described in three equivalent ways:

1. As a section, i.e. a map $\sigma : B \to A$ satisfying $\pi \circ \sigma = \mathrm{id}_B$.

2. As a lifted section, i.e. a map $f : P \to V$ satsifying $f(pg) = \rho(g)^{-1} f(p)$ for $p \in P, g \in G$. This map is defined as $f = \Lambda\sigma$ (see Theorem 7.2.4).

3. As a local section, i.e. a map $f_s : U \to V$ defined by $f_s = f \circ s$, where $s : U \to P$ is a local section of P defined on $U \subset B$.

In our study of linear operators between representation spaces, we will make use of the lifted and local representations.

In practice, feature maps (as well as filters/kernels) always have compact support, so we will henceforth assume that this is the case. This ensures that the integrals we consider are always well defined. Thus, from hereon, we take $\Gamma(A)$ to be the space of *compactly supported* sections of the associated vector bundle A. Likewise, $\hat{\Gamma}(A) = \mathrm{Hom}_\rho(P, V)$ is the corresponding space of lifts of compactly supported sections. Finally, we denote by $\Gamma_s(A)$ the space of local sections defined on $U \subset B$ relative to a gauge $s : U \to P$, for a compact U. In other words, $\Gamma_s(A)$ is the space of smooth maps $f_s : U \to V$.

Invariant integration & unimodularity

As mentioned in the last section, the groups $P = H$ and G are assumed to be Lie groups and hence locally compact. It follows that there exists a left invariant integration measure μ as well as a right invariant measure ν, not necessarily equal unless the group is unimodular, and both of which are unique up to a scale factor (Nachbin, 1965; Rudolph and Schmidt, 2012). For matrix Lie groups, unimodularity simply means that the Jacobian of the map $p \mapsto up$ (for $u, p \in P$) has determinant 1.

For simplicity, we will assume that P and G are unimodular, so that $\mu = \nu$, and there exists an invariant measure on P/G, which we will denote by the same symbol μ. This assumption can easily be loosened by inserting appropriate scaling factors when making substitutions in integrals, but we will not do so in order to reduce clutter.

Smoothing operators & kernels

The linear maps $\Phi : \Gamma(A_1) \to \Gamma(A_2)$ we consider are smoothing operators. These are the linear operators between said spaces that can be written as an integral transform with a smooth kernel. Indeed, such operators are in one-to-one correspondence with smooth kernels (Hörmander, 2003).

It is possible to generalize beyond smoothing operators. That is, one can use the Schwartz kernel theorem (Hörmander, 2003) to justify writing more general linear operators between function or section spaces as an integral transform, but this theorem applies to generalized functions / distributions, which is more than we need. Almost all existing G-CNN implementations discretize the base space via some grid of pixels $\{x_i\}_{i=1}^n \subset B$, with a function or kernel k on B stored as a set of values $k(x_i)$ associated to this grid. It does not seem reasonable to assume that such finite grids can well approximate a singular distribution such as a dirac delta, or implement a differential operator (which can be described as an integral transform with a non-smooth kernel), so we will restrict our attention to *smooth kernels*.

We note that some authors justify the use of smooth kernels via other assumptions such as linearity and boundedness (Duits and Burgeth, 2007; Kondor and Trivedi, 2018; Bekkers, 2019) but in our view it is justified by the application to assume that linear network layers are smoothing operators.

9.3.2 Integral transforms and kernels: trivial case

Before we study the general case, let us consider a simple special case: planar G-CNNs. In this setting, a feature map is just a smooth map $f_i : \mathbb{R}^2 \to \mathbb{R}^{C_n}$, where C_n is the number of channels (for input and output space $n = 1, 2$) and we use the shorthand $f_i = f_{s_i}$. We will discuss the role of the gauge s_i shortly; for now, consider f_i an arbitrary smooth map.

A linear layer Φ takes such a feature map f_i and produces an output feature map $(\Phi f)_j$. By hypothesis, Φ can be written as an integral transform:

$$(\Phi f)_j(x) = \int_{\mathbb{R}^2} k_{ji}(x, y) f_i(y) dy, \tag{9.5}$$

where $k_{ji} : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}^{C_2 \times C_1}$ is a smooth matrix-valued function called the kernel of the integral transform.

The point we wish to make in this section is that, as indicated by the subscripts, f_i, $(\Phi f)_j$ and k_{ji} should really be viewed as "local" representations of underlying fields f, Φf and k relative to gauges $s_i, s_j : B \to P$. It is tempting to

forget about the subscripts and identify k_{ji}, f_i and $(\Phi f)_j$ with their underlying fields because in this simple example all computations can be done with k_{ji}, f_i and $(\Phi f)_j$, the gauges can be defined globally on all of $\mathbb{R}^2$, and there is not really a need to consider changing the gauge (and we can use the same gauge $s_i = s_j$ for the input and output). Nevertheless, it is important to distinguish the field and its representation relative to a gauge in order to understand how the present example fits into the general framework.

The present example is formalized in the general framework as follows. For the representation spaces of the input and output we choose $P = \mathrm{SE}(2)$, $G = \mathrm{SO}(2)$, $B = P/G = \mathbb{R}^2$. The input and output space may use different representations $(\rho_1, \mathbb{R}^{C_1})$ and $(\rho_2, \mathbb{R}^{C_2})$ of $\mathrm{SO}(2)$.

The principal bundle $P = \mathrm{SE}(2)$ is trivial, for it is isomorphic to $\mathbb{R}^2 \times \mathrm{SO}(2)$ as a topological space (though not as a group, since $\mathrm{SE}(2) = \mathbb{R}^2 \rtimes \mathrm{SO}(2)$). So we can choose a global (and in fact canonical) gauge $s : \mathbb{R}^2 \to \mathrm{SE}(2)$ which is defined as $s(x) = (x, e)$ (where e is the identity of $\mathrm{SO}(2)$). With respect to this canonical gauge, a section of $A = P \times_\rho \mathbb{R}^C$ (i.e. feature map) is expressed as a map $f_s : \mathbb{R}^2 \to \mathbb{R}^C$, as we had simply assumed before (recall Sections 7.1.7, 7.2.3 and Theorem 7.2.4). Formally, we have an isomorphism $\Gamma_s(A) \simeq \Gamma(A) \simeq \hat{\Gamma}(A)$ between maps $f_s : \mathbb{R}^2 \to \mathbb{R}^C$, sections $\sigma : \mathbb{R}^2 \to A$, and lifted sections $f : \mathrm{SE}(2) \to \mathbb{R}^C$. In the general case we would only have this isomorphism among spaces of fields restricted to the domain $U \subset B$ of s, over which the bundles are trivial.

From our discussion in Chapter 7 we know that if we change the gauge from s_i to s_k then there exists a function $g_{ki} : \mathbb{R}^2 \to \mathrm{SO}(2)$ (called the gauge transformation) such that $s_k(x) = s_i(x) g_{ki}(x)$. Furthermore, the representation of the input feature map f relative to the new gauge is

$$f_k(x) = f(s_k(x)) = f(s_i(x) g_{ki}(x)) = \rho_1(g_{ki}(x))^{-1} f(s_i(x)) = \rho_1(g_{ki}(x))^{-1} f_i(x). \quad (9.6)$$

Similarly, if we change the gauge for the output feature map from s_j to s_l, it should transform like

$$(\Phi f)_l(x) = \rho_2(g_{lj}(x))^{-1} (\Phi f)_j(x). \quad (9.7)$$

Furthermore, when the input and output are expressed relative to the gauges s_k and s_l respectively, we can express the linear map Φ using a kernel k_{lk}:

$$(\Phi f)_l(x) = \int_{\mathbb{R}^2} k_{lk}(x, y) f_k(y) dy. \quad (9.8)$$

To figure out how k_{lk} and k_{ji} are related, we rewrite $(\Phi f)_l$ and f_k in Eq. 9.8

using Eq. 9.6 and 9.7:

$$\rho_2(g_{lj}(x))^{-1}(\Phi f)_j(x) = \int_{\mathbb{R}^2} k_{lk}(x,y)\rho_1(g_{ki}(y))^{-1} f_i(y)dy. \tag{9.9}$$

Multiplying by $\rho_2(g_{lj}(x))$ on the left and comparing to Eq. 9.5, we find that:

$$k_{ji}(x,y) = \rho_2(g_{lj}(x))\; k_{lk}(x,y)\; \rho_1(g_{ki}(y))^{-1}. \tag{9.10}$$

Since k_{ji} and k_{lk} completely determine each other, we should think of them as different ways to represent the same kernel, relative to gauges s_i, s_j resp. s_l, s_k. The kernel itself is an invariant object that exists independent of a choice of gauge. Like the feature maps, this object is in fact a section of a certain associated vector bundle, which we define in the next section.

9.3.3 The bundle of two-argument kernels

Kernels corresponding to smoothing operators $\Phi : \Gamma(A_1) \to \Gamma(A_2)$ are sections of an associated vector bundle called the bundle of two-argument kernels. We will define this bundle in this section for general, not necessarily homogeneous principal bundles. The bundle of two-argument kernels is associated to the following principal bundle:

Definition 9.3.1 (Product principal bundle)**.** *Let $\pi_1 : P_1 \to B_1$ and $\pi_2 : P_2 \to B_2$ be principal bundles with base spaces B_1 and B_2 and structure groups G_1 and G_2, respectively. Then $P_2 \times P_1$ is a principal bundle with projection $\pi_2 \times \pi_1 : P_2 \times P_1 \to B_2 \times B_1$ defined by $(\pi_2 \times \pi_1)(p_2, p_1) = (\pi_2(p_2), \pi_1(p_1))$. The right action of $G_2 \times G_1$ on $(p_2, p_1) \in P_2 \times P_1$ is given by $(p_2, p_1)(g_2, g_1) = (p_2 g_2, p_1 g_1)$.*

One can verify that the action of $G_2 \times G_1$ on $P_2 \times P_1$ is transitive and free as it should be for $P_2 \times P_1$ to be a principal bundle.

In order to define an associated vector bundle, we need a representation of the structure group. In the case of the product principal bundle $P_2 \times P_1$, that means a representation of $G_2 \times G_1$. As suggested by the discussion in Section 9.3.2 and particularly Eq. 9.10, this product group should act on the space of linear maps (corresponding to the value of the kernel at points x, y) by changing the basis on the input and output side. This is formalized as follows.

Definition 9.3.2 (External tensor product representation)**.** *Let G_1 and G_2 be groups and $\rho_1 : G_1 \to \mathrm{GL}(V_1)$ and $\rho_2 : G_2 \to \mathrm{GL}(V_2)$ be representations. The external tensor product $\rho_2 \boxtimes \rho_1$ is a representation of $G_2 \times G_1$ on $V_2 \otimes V_1 \simeq \mathrm{Hom}(V_1, V_2)$*

(the space of linear maps from V_1 to V_2). It is defined (for $M \in \mathrm{Hom}(V_1, V_2)$ and $g_i \in G_i$) as

$$(\rho_2 \boxtimes \rho_1)(g_2, g_1)M = \rho_2(g_2)\, M\, \rho_1(g_1)^{-1}. \tag{9.11}$$

Now we are ready to define the bundle of two-argument kernels:

Definition 9.3.3 (Bundle of two-argument kernels)**.** *Let two representation spaces (P_1, G_1, ρ_1, V_1) and (P_2, G_2, ρ_2, V_2) be given. We have a product principal bundle $P_2 \times P_1$ with structure group $G_2 \times G_1$, and a representation $\rho_2 \boxtimes \rho_1$ of $G_2 \times G_1$. We can thus define the associated bundle $K = (P_2 \times P_1) \times_{\rho_2 \boxtimes \rho_1} \mathrm{Hom}(V_1, V_2)$, by quotienting $(P_2 \times P_1) \times \mathrm{Hom}(V_1, V_2)$ by the equivalence relation*

$$((p_2, p_1), M) \sim ((p_2 g_2,\ p_1 g_1),\ (\rho_2 \boxtimes \rho_1)(g_2,\ g_1)^{-1} M), \tag{9.12}$$

for any $(g_2, g_1) \in G_2 \times G_1$. K will be called the bundle of two-argument kernels.

We can now define what a two-argument kernel is:

Definition 9.3.4 (Two-argument kernel)**.** *A section $\kappa \in \Gamma(K)$ of the bundle of two-argument kernels $K = (P_2 \times P_1)_{\rho_2 \boxtimes \rho_1} \mathrm{Hom}(V_1, V_2)$ is called a two-argument kernel. As is the case for any section of an associated bundle, we can represent a kernel in three different ways:*

1. *As a section $\kappa \in \Gamma(K)$, i.e. a map $\kappa : B_2 \times B_1 \to K$ satisfying $\pi_K \circ \kappa = \mathrm{id}_{B_2 \times B_1}$.*
2. *As a lifted section $k \in \hat{\Gamma}(K)$, i.e. a map $k : P_2 \times P_1 \to \mathrm{Hom}(V_1, V_2)$ satisfying:*

$$\begin{aligned} k(p_2 g_2, p_1 g_1) &= (\rho_2 \boxtimes \rho_1)(g_2, g_1)^{-1} k(p_2, p_1) \\ &= \rho_2(g_2)^{-1} k(p_2, p_1) \rho(g_1). \end{aligned} \tag{9.13}$$

3. *As a local section $k_{ji}(x, y) = k(s_j(x), s_i(y))$, expressed relative to the two gauges $s_i : U_i \to \pi_1^{-1}(U_i)$ and $s_j : U_j \to \pi_2^{-1}(U_j)$ defined on trivializing neighbourhoods $U_i \subset B_1$ and $U_j \subset B_2$.*

In case where the principal bundles P_1 and P_2 (and hence $P_2 \times P_1$) are homogeneous there is one more interesting and important observation to make. Namely, in this case the bundle $K = (P_2 \times P_1) \times_{\rho_2 \boxtimes \rho_1} \mathrm{Hom}(V_1, V_2)$ is a homogeneous vector bundle, and as discussed in Section 9.2.3 the space of sections $\Gamma(K)$ of such a bundle defines a representation called the induced representation. Specifically, this is a representation of the group $P_2 \times P_1$ acting on $\Gamma(K)$. In

the lifted form, it is defined as follows. Denoting the induced representation by $\lambda = \text{ind}_{G_2 \times G_1}^{P_2 \times P_1} \rho_2 \boxtimes \rho_1$ and for $k \in \hat{\Gamma}(K)$,

$$(\lambda(p,q)k)(p',q') = k(p^{-1}p', q^{-1}q'). \tag{9.14}$$

We will come back to this in Section 9.4 when we study equivariant smoothing operators.

As a bibliographical note, Ceccherini-Silberstein et al. (2009) define the space of lifted sections of K directly, though without defining K itself and without using the language of fiber bundles, and only for the case where P and G are finite groups. Similarly Folland (1995) defines the space of lifted sections of vector bundles associated to a homogeneous principal bundle but without defining the underlying bundles. We have not been able to locate a source covering the bundle K in general as we do here, though we would be surprised if it has not been described before.

9.3.4 Relations between local kernels

In Section 9.3.2 we saw that for a trivial bundle, the kernels corresponding to different gauges are related by a certain kind of gauge transformation (Equation 9.10). We derived this from a constraint on how the input and output feature maps should transform. Having defined the bundle of two-argument kernels K, we can now see that this transformation behaviour of the kernel is simply a consequence of it being a section of K.

Let $k \in \hat{\Gamma}(K)$ be a kernel represented as a lifted section. If we have gauges $s_i : U_i \to \pi_1^{-1}(U_i)$ and $s_j : U_j \to \pi_2^{-1}(U_j)$ expressed on trivializing neighbourhoods $U_i \subset B_1$ and $U_j \subset B_2$ then we can express the kernel on the base space as $k_{ji}(x,y) = k(s_j(x), s_i(y))$. Choosing different gauges s_k, s_l defined on overlapping neighbourhoods, we have $k_{lk}(x,y) = k(s_l(x), s_k(y))$. This gives:

$$\begin{aligned} k_{lk}(x,y) &= k(s_l(x), s_k(y)) \\ &= k(s_j(x) g_{lj}(x), s_i(y) g_{ki}(y)) \\ &= \rho_2(g_{lj}(x))^{-1}\; k(s_j(x), s_i(y))\; \rho_1(g_{ki}(y)) \\ &= \rho_2(g_{lj}(x))^{-1}\; k_{ji}(x,y)\; \rho_1(g_{ki}(y)). \end{aligned} \tag{9.15}$$

This relation between k_{lk} and k_{ji} is in agreement with the one found in Eq. 9.10 for globally defined gauges of a trivial bundle. This confirms that defining the kernel as a section of K is appropriate.

9.3.5 Linear maps as integral transforms (local)

Given a feature map f and kernel k, how should we compute an output feature map? In the general case we cannot represent a feature map globally as a smooth map $f_s : B \to V$ on the whole base space as we did for trivial kernels (Section 9.3.2), but we can still represent it as a collection of maps $f_{s_i} : U_i \to V$ where U_i cover B. Assuming that the kernel has a small support that is contained on one chart U_i, we can then compute the output feature map as before in the case of a trivial bundle (Section 9.3.2).

More specifically, we assume that for every $x \in B_2$, the support of the kernel $\kappa(x, -)$ is contained in some trivializing neighbourhood $U_i \subset B_1$. We say that such kernels have small enough support. Thus, using appropriate gauges s_i, s_j of B_2 and B_1 respectively, we can write

$$(\Phi f)_j(x) = \int_{U_i} k_{ji}(x, y) f_i(y) d\mu(y). \tag{9.16}$$

This expression is similar to Eq. 9.5, only now we are integrating over $U_i \subset B_1$ rather than $\mathbb{R}^2$.

In most CNNs, the kernel is indeed very small and so the assumption of small enough support of the kernel is realistic. For kernels whose support is not small enough to fit inside a single chart, we can still compute the integral transform by integrating over a number of charts modulated by a partition of unity, and summing the results (Tu, 2010).

We note that every choice of smooth kernel $k_{ji} : B_2 \times B_1 \to \mathrm{Hom}(V_1, V_2)$ with small enough support defines a smoothing operator via the above equation. Conversely, smoothing operators with small enough support correspond to a unique kernel. In the next section we will prove this for kernels with arbitrarily large compact support, using the lifted representation of sections.

9.3.6 Linear maps as integral transform (lifted)

So far, we have studied linear maps between local section spaces $\Gamma_s(A)$. This is a gauge-dependent, local, and non-redundant representation that is (after discretization) suitable for computer implementation. It is however also possible to perform a study of linear maps between spaces of lifted sections $\hat{\Gamma}(A)$, which gives a gauge-independent and global, but redundant representation. This representation is most suitable for stating and proving the main theorems in a clean way.

We will show that the space of smoothing operators $\Phi : \hat{\Gamma}(A_1) \to \hat{\Gamma}(A_2)$ between lifted section spaces is isomorphic to the space of lifted kernels $k \in \hat{\Gamma}(K)$ (assuming compact support for both sections and spaces, as stated before). In other words, each kernel corresponds to a unique operator Φ and vice versa (in a natural and linear manner). More formally, we show that the following spaces are isomorphic as vector spaces.

$$\begin{aligned} \mathcal{K} &= \mathrm{Hom}_{\rho_2 \boxtimes \rho_1}(P_2 \times P_1, \mathrm{Hom}(V_1, V_2)) = \hat{\Gamma}(K) \\ \mathcal{H} &= \mathrm{Hom}(\mathrm{Hom}_{\rho_1}(P_1, V_1), \mathrm{Hom}_{\rho_2}(P_2, V_2)) = \mathrm{Hom}(\hat{\Gamma}(A_1), \hat{\Gamma}(A_2)). \end{aligned} \tag{9.17}$$

The notation $\mathrm{Hom}_\rho(X, Y)$ indicates a space of maps from X to Y that are equivariant with respect to the representation ρ of some group G, as in Eq. 9.13. The space $\mathcal{K}$ is the space of two-argument kernels (Definition 9.3.4) in lifted form, while $\mathcal{H}$ is the space of smoothing operators between spaces of lifted sections.

We note that both $\mathcal{K}$ and $\mathcal{H}$ are representations of $P_2 \times P_1$. Specifically, $\mathcal{K}$ is the representation of $P_2 \times P_1$ induced by the representation $\rho_2 \boxtimes \rho_1$ of $G_2 \times G_1$ (see Section 9.2.3). This representation is defined as

$$\mathrm{ind}_{G_2 \times G_1}^{P_2 \times P_1} (\rho_2 \boxtimes \rho_1)(u_2, u_1)\, k(p_2, p_1) = k(u_2^{-1} p_2, u_1^{-1} p_1), \tag{9.18}$$

for $u_1, p_1 \in P_1$ and $u_2, p_2 \in P_2$ and $k \in \mathcal{K}$.

To make $\mathcal{H}$ into a representation of $P_2 \times P_1$, we note that P_n acts on $\hat{\Gamma}(A_n)$ via automorphisms $(\alpha_u \cdot f)(p) = f(u^{-1} p)$ for $u, p \in P_n$ and $f \in \hat{\Gamma}(A_n)$ (see 9.2.2 and 7.3.5), i.e. on the input and output of an operator $\Phi \in \mathcal{H}$. Thus we can define

$$\lambda(u_2, u_1)\Phi = \alpha_{u_2} \circ \Phi \circ \alpha_{u_1}^{-1} \tag{9.19}$$

As we will show, with these definitions $\mathcal{K} \simeq \mathcal{H}$ not just as vector spaces but also as representations.

To prove $\mathcal{K} \simeq \mathcal{H}$, we define the spaces $\widetilde{\mathcal{K}}$ and $\widetilde{\mathcal{H}}$ of which $\mathcal{K}$ and $\mathcal{H}$ are subspaces, show that they are isomorphic, and then show that the isomorphism restricts to an isomorphism between $\mathcal{K}$ and $\mathcal{H}$. The spaces $\widetilde{\mathcal{K}}$ and $\widetilde{\mathcal{H}}$ are defined by removing the equivariance constraints in 9.17:

$$\begin{aligned} \widetilde{\mathcal{K}} &= \mathrm{Hom}(P_2 \times P_1, \mathrm{Hom}(V_1, V_2)) \\ \widetilde{\mathcal{H}} &= \mathrm{Hom}(\mathrm{Hom}(P_1, V_1), \mathrm{Hom}(P_2, V_2)). \end{aligned} \tag{9.20}$$

That is, $\widetilde{\mathcal{K}}$ is the space of smooth maps (kernels) $\tilde{k} : P_2 \times P_1 \to \mathrm{Hom}(V_1, V2)$. The definition is similar to that of $\mathcal{K}$, except that kernels in $\widetilde{\mathcal{K}}$ do not need to

be equivariant with respect to $\rho_2 \boxtimes \rho_1$. $\widetilde{H}$ is the space of smoothing operators between function spaces $\mathrm{Hom}(P_1, V_1)$ and $\mathrm{Hom}(P_2, V_2)$ (again, compactly suported). Although we have suppressed this in the notation, we emphasize that $\widetilde{\mathcal{H}}$ only contains smoothing operators. It is clear that $\mathcal{K}$ is a linear subspace of $\widetilde{\mathcal{K}}$ and likewise that $\mathcal{H}$ is a linear subspace of $\widetilde{\mathcal{H}}$.

We have the following operator $\widetilde{\mathcal{A}} : \widetilde{\mathcal{K}} \to \widetilde{\mathcal{H}}$,

$$\widetilde{\mathcal{A}}(\tilde{k}) f(p) = \int_{P_1} \tilde{k}(p,q) f(q) d\mu(q). \tag{9.21}$$

So $\widetilde{\mathcal{A}}$ maps unconstrained kernels to linear operators between function spaces. It follows from the Schwartz Kernel theorem that under the stated regularity conditions, this operator $\widetilde{\mathcal{A}}$ is an isomorphism of vector spaces, i.e. we have a (linear) one-to-one correspondence between kernels $\tilde{k} \in \widetilde{\mathcal{K}}$ and smoothing operators $\tilde{\Phi} \in \widetilde{\mathcal{H}}$ (see e.g. Hörmander (2003), theorems 5.2.1 and 5.2.6).

The following theorem shows that $\widetilde{\mathcal{A}}$ restricts to an operator $\mathcal{A} : \mathcal{K} \to \mathcal{H}$ that maps two argument kernels $k \in \hat{\Gamma}(K)$ to operators $\Phi : \hat{\Gamma}(A_1) \to \hat{\Gamma}(A_2)$.

Theorem 9.3.1. *For $n = 1, 2$, let P_n be a Lie group with compact Lie subgroup G_n defining a homogeneous principal bundle $\pi_n : P_n \to P_n/G_n$. Let (ρ_n, V_n) be a representation of G_n. These data determine the associated bundle $A_n = P_n \times_{\rho_n} V_n$ and lifted homogeneous representation space $\hat{\Gamma}(A_n)$, as well as the bundle of two argument kernels $K = (P_2 \times P_1) \times_{\rho_2 \boxtimes \rho_1} \mathrm{Hom}(V_1, V_2)$. Let $\widetilde{\mathcal{K}}, \widetilde{\mathcal{H}}, \mathcal{K}$ and $\mathcal{H}$ be defined as in eqs. 9.20 and 9.17.*

Then the isomorphism $\widetilde{\mathcal{A}} : \widetilde{\mathcal{K}} \to \widetilde{\mathcal{H}}$ restricts to an isomorphism $\mathcal{A} : \mathcal{K} \to \mathcal{H}$. That is, the space of smoothing operators $\mathcal{H}$ is isomorphic to the space of two argument kernels $\mathcal{K}$ as a vector space.

Proof. Let $k \in \mathcal{K} = \hat{\Gamma}(K)$ be a lifted kernel. Since k satisfies $k(pg, q) = \rho_2(g)^{-1} k(p, q)$ for $g \in G_2$, we have for any $f \in \hat{\Gamma}(A_1)$,

$$\begin{aligned} \mathcal{A}(k) f(pg) &= \int_{P_1} k(pg, q) f(q) d\mu(q) \\ &= \rho_2(g)^{-1} \int_{P_1} k(p, q) f(q) d\mu(q) \\ &= \rho_2(g)^{-1} (\mathcal{A}(k) f)(p). \end{aligned} \tag{9.22}$$

So indeed $\mathcal{A}(k) f \in \hat{\Gamma}(A_2)$ as claimed. Since $\widetilde{\mathcal{A}}$ is a vector space isomorphism, and $\mathcal{A}$ is a restriction of $\widetilde{\mathcal{A}}$ to a linear subspace, it too is an isomorphism. ∎

An explicit inverse of $\mathcal{A}$ can be obtained as follows. Given an operator $\Phi \in \mathcal{H}$, i.e. a map $\Phi : \hat{\Gamma}(A_1) \to \hat{\Gamma}(A_2)$, we can extend it arbitrarily to a smoothing operator $\tilde{\Phi} \in \widetilde{\mathcal{H}}$ that agrees with Φ on $\mathcal{H} \subset \widetilde{\mathcal{H}}$. We obtain a kernel for $\tilde{\Phi}$ by $\tilde{k} = \widetilde{\mathcal{A}}^{-1}(\tilde{\Phi})$. Depending on how we chose $\tilde{\Phi}$, this kernel $\tilde{k}$ may not be in $\mathcal{K} \subset \widetilde{\mathcal{K}}$. However we can project it to $\mathcal{K}$ using the operator $\Pi_K : \widetilde{\mathcal{K}} \to \mathcal{K}$ defined as:

$$\begin{aligned} k(p,q) &\equiv (\Pi_K \tilde{k})(p,q) \\ &= \int_{G_2} \int_{G_1} \rho_2(g_2) \tilde{k}(pg_2, qg_1) \rho_1(g_1^{-1}) dg_1 dg_2 \end{aligned} \tag{9.23}$$

Importantly, this projection $\tilde{k} \mapsto k$ preserves the effect on sections $f \in \hat{\Gamma}(A_1)$, so we may take $k = \Pi_K \tilde{k}$ as the kernel of Φ.

Moreover, if there is another kernel $\tilde{k}'$ that implements Φ, then we have $\Phi = \mathcal{A}(\Pi_K(\tilde{k})) = \mathcal{A}(k)$ and also $\Phi = \mathcal{A}(\Pi_K(\tilde{k}')) = \mathcal{A}(k')$. Since $\mathcal{A}$ is invertible, we have $\Pi_K(\tilde{k}) = \Pi_K(\tilde{k}')$, i.e. both $\tilde{k}$ and $\tilde{k}'$ map to the same kernel $k = k' \in \hat{\Gamma}(K)$. So k is the unique kernel of Φ in $\hat{\Gamma}(K)$.

Theorem 9.3.2. *$\mathcal{A} : \mathcal{K} \to \mathcal{H}$ is in an isomorphism of representations of $P_2 \times P_1$.*

Proof. To show that $\mathcal{A}$ is an isomorphism of *representations* $\mathcal{K}$ and $\mathcal{H}$, it is sufficient to show that $\mathcal{A}$ is an isomorphism of vector spaces (Theorem 9.3.1), and that $\mathcal{A}$ intertwines (commutes with) the representations. Let π denote the induced representation $\mathrm{ind}_{G_2 \times G_1}^{P_2 \times P_1} \rho_2 \boxtimes \rho_1$ (Eq. 9.18), and let λ denote the representation in $\mathcal{H}$ defined in Eq. 9.19. The proof is a matter of some algebraic manipulation:

$$\begin{aligned} (\mathcal{A}(\pi(u_2,u_1)k)f)(p_2) &= \int_{P_1} k(u_2^{-1}p_2, u_1^{-1}p_1) f(p_1) d\mu(p_1) \\ &= \int_{P_1} k(u_2^{-1}p_2, p_1) f(u_1 p_1) d\mu(p_1) \\ &= \int_{P_1} k(u_2^{-1}p_2, p_1) (\alpha_{u_1}^{-1} \cdot f)(p_1) d\mu(p_1) \\ &= ((\mathcal{A}(k) \circ \alpha_{u_1}^{-1}) f)(u_2^{-1}p_2) \\ &= ((\alpha_{u_2} \circ \mathcal{A}(k) \circ \alpha_{u_1}^{-1}) f)(p_2) \\ &= (\lambda(u_2,u_1)\mathcal{A}(k)) f(p_2). \end{aligned} \tag{9.24}$$

So $\mathcal{A}\pi(u_2,u_1) = \lambda(u_2,u_1)\mathcal{A}$. ∎

9.4 Equivariant Homogeneous Linear Layers

We have seen that the space of smoothing operators between representation spaces $\mathcal{H}$ is isomorphic to the space of two-argument kernels $\mathcal{K}$. Here we show that the space of *equivariant* linear operators between representation spaces is isomorphic to the space of invariant two-argument kernels as well as the space of one-argument kernels. Concretely, this means that such equivariant linear operators can be written as a cross-correlation or convolution using such a one-argument kernel.

We again use the setup described in Section 9.3.1, with the additional assumption that $P_1 = P_2 = S$, i.e. that the input and output share the same total space for the principal bundle, and that this is also the group of symmetries.

9.4.1 Invariant Two-Argument Kernels

As noted before (Eq. 9.18), the space of two argument kernels $\mathcal{K} = \hat{\Gamma}(K)$ can be made into an induced representation of $P_2 \times P_1$, namely $\lambda = \mathrm{ind}_{G_2 \times G_1}^{P_2 \times P_1} \rho_2 \boxtimes \rho_1$. If we take $P_1 = P_2 = S$, the symmetry group of the representation space, we may restrict the induced representation to S acting diagonally, yielding the representation $\lambda_S = \mathrm{res}_S \mathrm{ind}_{G_2 \times G_1}^{S \times S} \rho_2 \boxtimes \rho_1$:

$$[\lambda_S(s)k](p,q) = k(s^{-1}p, s^{-1}q). \tag{9.25}$$

Definition 9.4.1 (Invariant Two-argument Kernel)**.** *A kernel $k \in \mathcal{K} = \hat{\Gamma}(K)$ is called invariant if $\lambda_S(s)k = k$ for $s \in S$, where $\lambda_S = \mathrm{res}_S \mathrm{ind}_{G_2 \times G_1}^{S \times S} \rho_2 \boxtimes \rho_1$, i.e. when $k(s^{-1}p, s^{-1}q) = k(p,q)$ for all $s, p, q \in S$. The space of invariant kernels is called $\mathcal{K}_S$.*

Definition 9.4.2 (Equivariant Linear Layer)**.** *An operator $\Phi : \hat{\Gamma}(A_1) \to \hat{\Gamma}(A_2)$ in $\mathcal{H}$ is called equivariant if for all $s \in S$, we have $\Phi \circ \alpha_s = \alpha_s \circ \Phi$. The space of equivariant operators will be denoted $\mathcal{H}_S$.*

Theorem 9.4.1. *Let (P, G_n, ρ_n) for $n = 1, 2$ define homogeneous representation spaces, i.e. P is a Lie group with compact Lie subgroup G_n and representation (ρ_n, V_n) of G_n, associated vector bundle $A_n = P \times_{\rho_n} V_n$ and representation space $\hat{\Gamma}(A_n)$. Assume the symmetry group $S = P$.*

Then the space of invariant two-argument kernels $\mathcal{K}_S$ (Def. 9.4.1) and equivariant linear layers $\mathcal{H}_S$ (Def. 9.4.2) are isomorphic as representations of S.

Proof. One way to see this is that by definition, $\mathcal{K}_S$ is the trivial component (isotypic subspace) of $\mathcal{K}$ (i.e. $\lambda_S(s)k = k$ for $s \in S$ and $k \in \mathcal{K}_S$ so the elements

of $\mathcal{K}_S$ transform according to a trivial representation), and similarly $\mathcal{H}_S$ is the trivial component of $\mathcal{H}$ (since for $\Phi \in \mathcal{H}_S$ we have $\alpha_s \circ \Phi \circ \alpha_s^{-1} = \Phi$). Since $\mathcal{H} \simeq \mathcal{K}$ it follows $\mathcal{H}_S \simeq \mathcal{K}_S$ as well.

Alternatively, one can show this as follows. Let $k \in \mathcal{K}$, and assume that $\mathcal{A}(k) \in \mathcal{H}_S$ i.e. that $\alpha_s \circ \mathcal{A}(k) \circ \alpha_s^{-1} = \mathcal{A}(k)$. We expand this expression:

$$\begin{aligned}
\alpha_s \circ \mathcal{A}(k) \circ \alpha_s^{-1}(f)(p) &= \mathcal{A}(k) \circ \alpha_s^{-1}(f)(\alpha_s^{-1}(p)) \\
&= \int_P k(\alpha_s^{-1}(p), q)(\alpha_s^{-1} \cdot f)(q) d\mu(q) \\
&= \int_P k(\alpha_s^{-1}(p), q) f(\alpha_s(q)) d\mu(q) \\
&= \int_P k(\alpha_s^{-1}(p), \alpha_s^{-1}(q)) f(q) d\mu(q).
\end{aligned} \tag{9.26}$$

Since this must equal $\int_P k(p,q) f(q) d\mu(q)$ for all f, we conclude that

$$k(\alpha_s^{-1}(p), \alpha_s^{-1}(q)) = k(p,q), \tag{9.27}$$

i.e. that $\lambda(s)k = k$. Conversely if $\lambda(s)k = k$ then it follows from the above that $\mathcal{A}(k) \in \mathcal{H}_S$. ∎

Next we will show that invariant two-argument kernels (and hence equivariant linear maps) are in one to one correspondence with one-argument convolution kernels. These are once again sections of a certain associated vector bundle, which we will now define.

9.4.2 The bundle of double cosets

In Section 9.2.1 we showed how a group P can be viewed as a principal bundle with structure group $G \subset P$, base space P/G, and cosets hG as fibers. Here we show another way to view P as a bundle, this time with product structure group $G_2 \times G_1$ (where both G_1 and G_2 are compact subgroups of P), base space $G_2 \backslash P / G_1$ and fibers the double cosets $G_2 h G_1$. This is however not a *principal* bundle, because as we will see the action of $G_2 \times G_1$ is not free. Whereas the principal bundle $P \to P/G$ defined in Section 9.2.1 is used to define representation spaces in a homogeneous G-CNN, the bundle we will define in this section is used to define spaces of cross-correlation kernels, which will be shown to be in one-to-one correspondence with the invariant two-argument kernels discussed in the previous section.

Definition 9.4.3 (Double Coset Bundle)**.** *Let P be a Lie group and G_1, G_2 two compact subgroups of P. Then we have a $G_2 \times G_1$-bundle with total space P, base space $G_2\backslash P/G_1$, and projection defined by:*

$$\begin{aligned} P &\xrightarrow{\pi} G_2\backslash P/G_1 \\ h &\mapsto G_2 h G_1. \end{aligned} \tag{9.28}$$

The set $G_2 h G_1$ is called a double coset, and $G_2\backslash P/G_1$ denotes the space of double cosets. Thus, the fibers of the projection are double cosets.

The right action of $G_2 \times G_1$ on the total space P is given by

$$h \cdot (g_2, g_1) = g_2^{-1} h g_1. \tag{9.29}$$

To verify that this is a $G_2 \times G_1$-bundle, we check that:

$$\pi(h \cdot (g_2, g_1)) = G_2 g_2^{-1} h g_1 G_1 = G_2 h G_1 = \pi(h).$$

The action of $G_2 \times G_1$ on double cosets is transitive. However, the action is not free. Consider for example the trivial case where $P = G_1 = G_2$. In this case we have $h \cdot (g_2, g_1) = h$ for any element $g_2 = g_1$ that commutes with h, not just the identity $(g_2, g_1) = (e, e)$. Thus, h is a fixed point of (g_2, g_1), and the action is not (fixed-point) free.

9.4.3 The bundle of one-argument kernels

Definition 9.4.4 (Bundle of one-argument kernels)**.** *Let P be a Lie group, and (P, G_1, ρ_1, V_1) and (P, G_2, ρ_2, V_2) be two homogeneous representation spaces. These data determine the double coset bundle $P \to G_2\backslash P/G_1$ with structure group $G_2 \times G_1$ (Definition 9.4.3), and a representation $\rho_2 \boxtimes \rho_1$ of $G_2 \times G_1$ (Definition 9.3.2). We can thus define an associated bundle (Definition 7.2.17) $K_D = P \times_{\rho_2 \boxtimes \rho_1} \mathrm{Hom}(V_1, V_2)$, by quotienting $P \times \mathrm{Hom}(V_1, V_2)$ by the equivalence relation*

$$(p, M) \sim (p \cdot (g_2, g_1), [\rho_2 \boxtimes \rho_1](g_2, g_1)^{-1} M) = (g_2^{-1} p g_1, \rho_2(g_2) M \rho_1(g_1)^{-1}). \tag{9.30}$$

for any $(g_2, g_1) \in G_2 \times G_1$. This bundle will be called the bundle of one-argument kernels or the bundle of cross-correlation kernels.

A one argument kernel is just a section of the bundle just defined.

Definition 9.4.5 (One-argument kernel)**.** *A one-argument / cross-correlation kernel is a section of the bundle $K_D = P \times_{\rho_2 \boxtimes \rho_1} \mathrm{Hom}(V_1, V_2)$, and may be represented in four different ways:*

1. *As a section* $\kappa \in \Gamma(K_D)$*, i.e. a map* $\kappa : G_2\backslash H/G_1 \to K_D$ *satisfying*

$$\pi_{K_D} \circ \kappa = \mathrm{id}_{G_2\backslash P/G_1}. \tag{9.31}$$

2. *As a lifted section* $k \in \hat{\Gamma}(K_D)$*, i.e. a map* $P \to \mathrm{Hom}(V_1, V_2)$ *satisfying:*

$$k(p \cdot (g_2, g_1)) = [\rho_2 \boxtimes \rho_1](g_2, g_1)^{-1} k(p), \tag{9.32}$$

or equivalently,

$$k(g_2^{-1} p g_1) = \rho_2(g_2)^{-1} k(p) \rho_1(g_1), \tag{9.33}$$

3. *As a collection of maps* $k_s : P/G_1 \to \mathrm{Hom}(V_1, V_2)$*, defined as:*

$$k_s(x) = k(s(x)), \tag{9.34}$$

where s *is a section of the* principal *bundle* $P \to P/G_1$.

4. *As a collection of maps* $k_\gamma : G_2\backslash P/G_1 \to \mathrm{Hom}(V_1, V_2)$ *defined as:*

$$k_\gamma(x) = k(\gamma(x)), \tag{9.35}$$

where $\gamma : G_2\backslash P/G_1 \to P$ *is a section of the* $G_2 \times G_1$*- bundle of double cosets* $P \to G_2\backslash P/G_1$.

We note that here we have defined local kernels on P/G_1 and $G_2\backslash P/G_1$ in terms of a global kernel $k \in \hat{\Gamma}(K_D)$. Later in Section 9.6 we will answer the question: when is a collection of functions k_s the local representation of a global kernel? First we will show that one-argument kernels are in one-to-one correspondence with invariant two-argument kernels, and hence with equivariant linear maps.

9.4.4 Convolution is all you need

Theorem 9.4.2. *The space of invariant two-argument kernels* $\mathcal{K}_S$ *(Def. 9.4.1) is isomorphic to the space of one-argument convolution kernels* $\hat{\Gamma}(K_D)$ *by the following vector space isomorphism:*

$$\mathcal{B}(k)(p) \equiv k(e, p) = k(p^{-1}, e), \tag{9.36}$$

with inverse

$$\mathcal{B}^{-1}(k)(p, q) \equiv k(p^{-1} q). \tag{9.37}$$

Proof. Let $k \in \mathcal{K}_S$ be an invariant two-argument kernel. Then

$$\begin{aligned}\mathcal{B}(k)(g_2^{-1}pg_1) &= k(e, g_2^{-1}pg_1)\\ &= k(g_2, pg_1)\\ &= \rho_2(g_2)^{-1}k(e,p)\rho_1(g_1)\\ &= \rho_2(g_2)^{-1}\mathcal{B}(k)(p)\rho_1(g_1).\end{aligned} \tag{9.38}$$

Hence, $\mathcal{B}(k) \in \hat{\Gamma}(K_D)$, i.e. it is a lifted one-argument kernel.

Conversely, if we have a one-argument kernel $k \in \hat{\Gamma}(K_D)$, then

$$\begin{aligned}\mathcal{B}^{-1}(k)(pg_2, qg_1) &= k(g_2^{-1}p^{-1}qg_1)\\ &= \rho_2(g_2)^{-1}k(p^{-1}q)\rho_1(g_1)\\ &= \rho_2(g_2)^{-1}\mathcal{B}(k)(p,q)\rho_1(g_1).\end{aligned} \tag{9.39}$$

So $\mathcal{B}^{-1}(k) \in \mathcal{K}$, i.e. $\mathcal{B}(k)$ is a two argument kernel. It is also invariant,

$$\begin{aligned}\mathcal{B}^{-1}(k)(sp, sq) &= k(p^{-1}s^{-1}sq)\\ &= k(p^{-1}q)\\ &= \mathcal{B}(k)(p,q),\end{aligned} \tag{9.40}$$

so $\mathcal{B}^{-1}(k) \in \mathcal{K}_S$.

Furthermore, $\mathcal{B}$ and $\mathcal{B}^{-1}$ are indeed inverses:

$$\begin{aligned}\mathcal{B}(\mathcal{B}^{-1}(k))(p) &= \mathcal{B}^{-1}(k)(e,p) = k(e^{-1}p) = k(p).\\ \mathcal{B}^{-1}(\mathcal{B}(k))(p,q) &= \mathcal{B}(k)(p^{-1}q) = k(e, p^{-1}q) = k(p,q).\end{aligned} \tag{9.41}$$

■

Theorem 9.4.3 (Convolution is all you need)**.** *Any equivariant linear map $\Phi \in \mathcal{H}_S$ can be written as a convolution with a one-argument kernel $k \in \hat{\Gamma}(K_D)$.*

Proof. We know from Theorem 9.3.1 that any smoothing operator $\Phi \in \mathcal{H}$ (and hence any equivariant $\Phi \in \mathcal{H}_S$) can be written as an integral transform $\Phi f(p) = k \cdot f(p) = \int_P k(p,q)f(q)d\mu(q)$, where $k \in \hat{\Gamma}(K)$ is a unique two-argument kernel (Definition 9.3.4). Moreover, we know from Theorem 9.4.1 that Φ is equivariant if and only if k is invariant, i.e. $\mathcal{H}_S \simeq \mathcal{K}_S$. Using the equivalence between

invariant two-argument kernels and one-argument convolution kernels (Theorem 9.4.2), we can write Φ as a convolution-like integral[3].

$$\mathcal{A}(k)f(p) = \int_P k(p,q)f(q)d\mu(q) = \int_H k(p^{-1}q)f(q)d\mu(q) = k \star f(p). \tag{9.42}$$

■

[3]Technically this may be called a cross-correlation, but it customary in the deep learning literature to refer to this as a convolution as well.

9.5 Local description

In this section we study the local form of the induced representation and equivariant smoothing operators / convolutions. We begin by studying local sections and trivializations of a homogeneous principal bundle $P \to P/G$. This allows us to give a local expression for the left action of the Lie group P on itself, and the induced representation which is just the action of P on the space of sections of the associated vector bundle. Finally, we obtain an expression for the convolution / cross-correlation on the base space, in terms of local sections of the principal bundles.

9.5.1 Local section and trivialization of $P \to P/G$

Once again, let P be a Lie group and G a closed Lie subgroup, so that $P \xrightarrow{\pi} P/G$ is a homogeneous principal bundle. Let us choose a local section (gauge) of $P \to P/G$, i.e. a map $s : U \to P$ such that $\pi \circ s = \mathrm{id}_U$ (for a neighbourhood $U \subset P/G$). Since P is a principal bundle, a global section will only exist of P is trivial, i.e. if $P \simeq P/G \times G$ as a topological space (Thm. 7.2.3).

Using this section, we can define a local trivialization (see Thm. 7.2.2),

$$\begin{aligned} &\varphi : \pi^{-1}(U) \to U \times G \\ &\qquad \varphi(p) = (pG, s(pG)^{-1}p) \equiv (\pi(p), g_s(p)) \\ &\varphi^{-1}(pG, g) = s(pG)g. \end{aligned} \tag{9.43}$$

One can show that φ and φ^{-1} are indeed inverses. Notice that defining φ^{-1} as above can be done for any principal bundle (see Thm. 7.2.2), but usually there is no simple expression for $g_s : \pi^{-1}(U) \to G$ because the inverse of $s(x) \in P$ is not defined when P is not a group.

We verify that $g_s(p) \in G$ as follows

$$g_s(p)G = s(pG)^{-1}pG = s(pG)^{-1}s(pG)G = G. \tag{9.44}$$

Here we used the fact that since s is a section, $pG = \pi(s(pG)) = s(pG)G$. So we can think of g_s as returning "the G-part of $p \in P$". This is not an intrinsic notion, which is to say, g_s depends on a choice of gauge s or trivialization φ.

For φ to be a morphism of G-bundles, we must have that $g_s(pg) = g_s(p)g$ for $p \in P$ and $g \in G$. Indeed this is the case:

$$g_s(pg) = s(pgG)^{-1}pg = s(pG)^{-1}pg = g_s(p)g \tag{9.45}$$

It follows that $\varphi(pg) = \varphi(p)g$ and hence φ is a P/G-isomorphism between the trivial principal G-bundles $\pi^{-1}(U)$ and $U \times G$ (i.e. it is a local trivialization).

9.5.2 Local form of the left action

We noted in Section 9.2.2 that the left action of P on itself, $p \mapsto up$ for $u, p \in P$, is a principal bundle automorphism: $\alpha_u(p) = up$ and $\beta_u(pG) = upG$. Furthermore, we know from Proposition 8 that locally, any G-bundle automorphism can be written as $(x, g) \mapsto (\beta(x), g^\alpha(x)g)$. Indeed if we choose sections s_i, s_j and corresponding trivializations φ_i, φ_j with appropriate domains, we may write

$$\begin{aligned} \varphi_j \circ \alpha_u \circ \varphi_i^{-1}(x, g) &= \varphi_j \circ \alpha_u(s_i(x)g) \\ &= \varphi_j(us_i(x)g) \\ &= (\pi(us_i(x)g), s_j(us_i(x)gG)^{-1} us_i(x)g) \\ &= (ux, s_j(ux)^{-1} us_i(x)g) \end{aligned} \tag{9.46}$$

So the local representation of α_u relative to s_i, s_j is

$$g_{ij}^u(x) = s_j(ux)^{-1} us_i(x). \tag{9.47}$$

This leads to the following useful expression that we will use later:

$$us_i(x) = s_j(ux) g_{ij}^u(x). \tag{9.48}$$

We verify that $g_{ij}^u(x) \in G$:

$$g_{ij}^u(x)G = s_j(ux)^{-1} us_i(x)G = s_j(ux)^{-1} ux = s_j(ux)^{-1} s_j(ux)G = G. \tag{9.49}$$

We note that $g_{ij}^e(x)$ is equal to $g_{ij}(x)$, the gauge transformation between s_i and s_j, defined as $g_{ij}(x) = s_j(x)^{-1} s_i(x)$. Moreover, we have $g_{ij}(\pi(p)) = g_{s_j}(p) g_{s_i}(p)^{-1}$, where $g_{s_i}(p) = s_i(pG)^{-1} p$ (and similarly for g_{s_j}) are as defined in the previous section (Eq. 9.43).

9.5.3 Local form of the induced representation

Given a section $s_i : U_i \to P$ of the principal bundle P defined on $U_i \subset P/G$, we can pull back $f \in \hat{\Gamma}(A)$ (a lifted section of A) to U_i:

$$f_i(x) = f(s_i(x)) \tag{9.50}$$

We can pull back $f' = \lambda(u)f$ as well, where λ is the induced representation $\lambda = \mathrm{ind}_G^P \rho$ (Section 9.2.3), yielding $f_i'(x) = f(u^{-1}s_i(x))$. In order to write f_i' in terms of another local representation of f, we make use of Eq. 9.48 and the equivariance of lifted sections (Sec. 7.2.4):

$$\begin{aligned} f_i'(x) &= f(u^{-1}s_i(x)) \\ &= f(s_j(u^{-1}x)\, g_{ij}^{u^{-1}}(x)) \\ &= \rho(g_{ij}^{u^{-1}}(x))^{-1}\ f(s_j(u^{-1}x)) \\ &= \rho(g_{ij}^{u^{-1}}(x))^{-1}\ f_j(u^{-1}x) \end{aligned} \tag{9.51}$$

Thus, if we represent a field as a collection of local functions f_i rather than a single (redundant) lifted section f, we can apply the induced representation $\lambda(u)$ using this formula.

9.5.4 Kernel & correlation on coset space

We now return to the setting introduced in Section 9.3.1, i.e. two ($n = 1, 2$) homogeneous principal bundles $P \to P/G_n$ with the same total space P.

Given a section $s_i : U_i \to P$ of the principal bundle $P \to P/G_1$, we can write a one-argument convolution kernel $k \in \hat{\Gamma}(K_D)$ (Definition 9.4.5) in local form:

$$k_i(x) = k(s_i(x)). \tag{9.52}$$

If we have another section s_j, we get

$$k_j(x) = k(s_j(x)) = k(s_i(x)g_{ij}(x)) = k_i(x)\rho_1(g_{ij}(x)) \tag{9.53}$$

In addition to $s_i : U_i \to P$ (with $U_i \subset P/G_1$) also choose a section $s_j : U_j \to P$ of the other bundle, i.e. with $U_j \subset P/G_2$. We find the local form of the cross-correlation (Eq. 9.42) as follows

$$\begin{aligned} (k \star f)_{s_j}(x) &= k \star f(s_j(x)) \\ &= \int_P k(s_j(x)^{-1}q) f(q) d\mu(q) \\ &= \int_{P/G_1} \int_{G_1} k(s_j(x)^{-1}s_i(y)g) f(s_i(y)g) d\mu(g) d\mu(y) \\ &= \int_{P/G_1} k(s_j(x)^{-1}s_i(y)) f_{s_i}(y) d\mu(y) \end{aligned} \tag{9.54}$$

(We remind the reader that we use the same symbol μ to refer to the invariant measures on P as well as the induced measures on G_1 and P/G_1)

If we choose another section s_k of P/G_1 with appropriate domain, we can use Eq. 9.48 and the shorthand notation $g_{ijk}(x,y) = g_{ik}^{s_j(x)^{-1}}(y)$. to write

$$s_j(x)^{-1} s_i(y) = s_k(s_j(x)^{-1} y) g_{ijk}(x,y) \tag{9.55}$$

Assuming the support of the kernel is contained in the domain of s_k, the correlation is written as

$$\begin{aligned} k_{s_k} \star f_{s_i}(x) &= \int_{P/G_1} k(s_j(x)^{-1} s_i(y)) f_{s_i}(y) d\mu(y) \\ &= \int_{P/G_1} k(s_k(s_j(x)^{-1} y) g_{ijk}(x,y)) f_{s_i}(y) dy \\ &= \int_{P/G_1} k_{s_k}(s_j(x)^{-1} y)\, \rho_1(g_{ijk}(x,y))\, f_{s_i}(y) dy, \end{aligned} \tag{9.56}$$

where we used the fact that $g_{ijk}(x,y) \in G_1$ and the equivariance property of lifted kernels k. We have thus expressed the convolution entirely in terms of local data.

9.6 Characterization of local kernels

A one-argument (convolution) kernel is most easily defined (9.4.5) in lifted form as a map $k : P \to \mathrm{Hom}(V_1, V_2)$ satisfying $k(g_2^{-1} p g_1) = \rho_2(g_2)^{-1} k(p) \rho_1(g_1)$ for $p \in P$, $g_1 \in G_1$, and $g_2 \in G_2$. Then, given sections / gauges $s_i : U_i \to P$ (defined on neighbourhoods $U_i \subset P/G_1$ covering P/G_1), one can represent the kernel as a collection of local maps $k_i(x) = k(s_i(x))$. Similarly one can express the kernel as a collection of maps on local neighbourhoods of $G_2 \backslash P / G_1$ using sections $\gamma : U \to P$ for $U \subset G_2 \backslash P / G_1$. The question we consider in this section is: what constraints should a collection of maps k_i satisfy in order to be the local representation of some globally defined kernel k? This question is relevant because in a practical implementation of an equivariant linear map / convolution, one would typically store the kernel as a collection of functions on local neighbourhoods (or just one neighbourhood if the kernel has small enough support), without ever explicitly representing the lifted kernel k, because the latter is a highly redundant representation which contains the value of the kernel in any gauge. Hence it is important to know what constraints such a local kernel should satisfy.

9.6.1 Characterization of kernels on coset space

Theorem 9.6.1. *Let $s_i : U_i \to P$ be a collection of gauges with U_i covering P/G_1. Let $\pi(p) = pG_1$ be the projection of the principal bundle $P \to P/G_1$, and $g_i(p) = g_{s_i}(p) = s_i(\pi(p))^{-1} p$ be the G_1-part of p relative to s_i (Eq. 9.43).*

A collection of local kernels $k_i : U_i \to \mathrm{Hom}(V_1, V_2)$ uniquely defines a lifted one-argument kernel $k : P \to \mathrm{Hom}(V_1, V_2)$ (Definition 9.4.5) via

$$k(p) \equiv k_i(\pi(p)) \rho_1(g_i(p)), \tag{9.57}$$

if and only if the following conditions are satisfied:

1. *for all $x \in U_i \cap U_j$,*
$$k_i(x) = k_j(x) \rho_1(g_{ij}(x)), \tag{9.58}$$
where $g_{ij}(x) = s_j(x)^{-1} s_i(x)$,
2. *and for all $g_2 \in G_2$ and $x \in U_i$,*
$$k_j(g_2 x) = \rho_2(g_2) k_i(x) \rho(g_{ij}^{g_2}(x))^{-1}. \tag{9.59}$$
where $g_{ij}^{g_2}(x) = s_j(g_2 x)^{-1} g_2 s_i(x)$ as defined in Eq. 9.47.

Proof. Given a lifted kernel $k \in \hat{\Gamma}(K_D)$, the local kernels $k_i(x) = k(s_i(x))$ satisfy conditions 1 and 2, for

$$k_i(x) = k(s_i(x)) = k(s_j(x)g_{ij}(x)) = k(s_j(x))\rho_1(g_{ij}(x)) = k_j(x)\rho_1(g_{ij}(x)), \tag{9.60}$$

and

$$\begin{aligned} k_j(g_2x) &= k(s_j(g_2x)) \\ &= k(s_j(g_2x)g_{ij}^{g_2}(x)g_{ij}^{g_2}(x)^{-1}) \\ &= k(g_2s_i(x)g_{ij}^{g_2}(x)^{-1}) \\ &= \rho_2(g_2)k_i(x)\rho_1(g_{ij}^{g_2}(x))^{-1}. \end{aligned} \tag{9.61}$$

Conversely, if we are given a collection of local kernels k_i satisfying the two constraints, we may define $k(p) = k_i(\pi(p))\rho_1(g_i(p))$. We must verify that this definition is consistent when $\pi(p) \in U_i \cap U_j$, and that k satisfies the equivariance constraint $k(g_2pg_1) = \rho_2(g_2)k(p)\rho_1(g_1)$.

Let $p \in P$ be such that $\pi(p) \in U_i \cap U_j$, then both $k(p) = k_i(\pi(p))\rho_1(g_i(p))$ and $k(p) = k_j(\pi(p))\rho_1(g_j(p))$ should hold. Indeed this is the case:

$$\begin{aligned} k_i(\pi(p))\rho_1(g_i(p)) &= k_j(\pi(p))\rho_1(g_{ij}(\pi(p))g_i(p)) \\ &= k_j(\pi(p))\rho_1(g_j(p)g_i(p)^{-1}g_i(p)) \\ &= k_j(\pi(p))\rho_1(g_j(p)). \end{aligned} \tag{9.62}$$

Here we used Eq. 9.58 and the fact that $g_{ij}(\pi(p)) = g_j(p)g_i(p)^{-1}$.

For the equivariance constraint, we have

$$\begin{aligned} k(pg_1) &= k_i(\pi(pg_1))\rho_1(g_i(pg_1)) \\ &= k_i(\pi(p))\rho_1(g_i(p))\rho_1(g_1) \\ &= k(p)\rho_1(g_1). \end{aligned} \tag{9.63}$$

Also,

$$\begin{aligned} k(g_2p) &= k_j(\pi(g_2p))\rho_1(g_j(g_2p)) \\ &= k_j(g_2\pi(p))\rho_1(g_j(g_2p)) \\ &= \rho_2(g_2)k_j(\pi(p))\rho_1(g_{ij}^{g_2}(\pi(p))^{-1}g_j(g_2p)) \\ &= \rho_2(g_2)k_j(\pi(p))\rho_1(g_i(p)) \\ &= \rho_2(g_2)k(p). \end{aligned} \tag{9.64}$$

■

9.6.2 Characterization of kernels on double coset space

We repeat the analysis for kernels on the double coset space $G_2\backslash P/G_1$ (see Eq. 9.35). Before we can state a theorem analogous to Theorem 9.6.1, we need to study the double coset decomposition.

Ambiguity in the double coset decomposition

A section γ of the double coset bundle $P \to G_2\backslash P/G_1$ (Def. 9.4.3) is a map satisfying the equation $G_2\gamma(x)G_1 = x$ for all $x \in U \subset G_2\backslash P/G_1$. In other words, γ picks for each double coset x a representative $\gamma(x)$ in that double coset.

Any element $p \in P$ can be decomposed as $p = g_2\gamma(x)g_1$ for some $x \in G_2\backslash P/G_1$, $g_1 \in G_1$ and $g_2 \in G_2$. Although $x = G_2pG_1$ is uniquely defined, there is a potential ambiguity in the choice of g_1 and g_2. Indeed, assume that $p = g_2\gamma(x)g_1 = g_2'\gamma(x)g_1'$. Then $\gamma(x) = g_2^{-1}g_2'\gamma(x)g_1'g_1^{-1}$. Multiplying by G_1, we have $\gamma(x)G_1 = g_2^{-1}g_2'\gamma(x)G_1$, i.e. $g_2^{-1}g_2'$ (an element of G_2) is in the stabilizer subgroup of $\gamma(x)G_1 \in P/G_1$. We denote this stabilizer as $G_2^{\gamma(x)G_1}$ and can be expressed as:

$$\begin{aligned} G_2^{\gamma(x)G_1} &= \{g \in G_2 \mid g\gamma(x)G_1 = \gamma(x)G_1\} \\ &= \{g \in G_2 \mid g \in \gamma(x)G_1\gamma(x)^{-1}\} \\ &= G_2 \cap \gamma(x)G_1\gamma(x)^{-1}. \end{aligned} \tag{9.65}$$

Similarly the element $g_1'g_1^{-1}$ is in the stabilizer

$$\begin{aligned} G_1^{G_2\gamma(x)} &= \{g \in G_1 \mid G_2\gamma(x)g = G_2\gamma(x)\} \\ &= G_1 \cap \gamma(x)^{-1}G_1\gamma(x). \end{aligned} \tag{9.66}$$

From the last expression it is easy to see that these groups are conjugate:

$$G_1^{G_2\gamma(x)} = \gamma(x)^{-1}G_2^{\gamma(x)G_1}\gamma(x). \tag{9.67}$$

So $g_2^{-1}g_2'$ and $g_1'g_1^{-1}$ are elements of conjugate subgroups, and indeed these elements themselves are conjugate: $\gamma(x)^{-1}g_2^{-1}g_2'\gamma(x) = (g_1'g_1^{-1})^{-1}$.

Conversely, for any $g_2 \in G_2^{\gamma(x)G_1}$ and conjugate $g_1 = \gamma(x)^{-1}g_2\gamma(x) \in G_1^{G_2\gamma(x)}$ we have

$$g_2\gamma(x)g_1^{-1} = \gamma(x) \tag{9.68}$$

Thus whenever we have $p = g_2\gamma(x)g_1$ then also $p = g_2\tilde{g}_2\gamma(x)\tilde{g_1}^{-1}g_1$, for any $\tilde{g}_2 \in G_2^{\gamma(x)G_1}$ and $\tilde{g_1} = \gamma(x)^{-1}g_2\gamma(x) \in G_1^{G_2\gamma(x)}$. That is, these stabilizers completely capture the non-uniqueness of the double coset decomposition of p.

Characterization

Theorem 9.6.2. *Let P be a Lie group with compact subgroups G_1, G_2, and $P \to G_2\backslash P/G_1$ be the (G_2, G_1)-bundle defined by the projection $p \mapsto G_2 p G_1$. Let $\gamma_i : U_i \to P$ be a collection of sections of P with U_i covering $G_2\backslash P/G_1$.*

A collection of local kernels $k_i : U_i \to \mathrm{Hom}(V_1, V_2)$ uniquely defines a lifted one-argument kernel $k : P \to \mathrm{Hom}(V_1, V_2)$ (Definition 9.4.5) via

$$k(p) = k(g_2\gamma_i(x)g_1) \equiv \rho_2(g_2)\, k_i(x)\, \rho_1(g_1), \tag{9.69}$$

(where $p = g_2\gamma_i(x)g_1$ is a general element of P that decomposes into $g_1 \in G_1$, $g_2 \in G_2$, and $x \in G_2\backslash P/G_1$), if and only if the local kernels satisfy:

1. *for all $x \in U_i \cap U_j$,*

$$k_i(x) = \rho_2(g^2_{ij}(x))k_j(x)\rho_1(g^1_{ij}(x)), \tag{9.70}$$

where $g^1_{ij} : U_i \cap U_j \to G_1$ and $g^2_{ij} : U_i \cap U_j \to G_2$ satisfy $\gamma_i(x) = g^2_{ij}(x)\gamma_j(x)g^1_{ij}(x)$.

2. *and for all $g_2 \in G_2^{\gamma_i(x)G_1}$ and $g_1 = \gamma_i(x)^{-1} g_2 \gamma_i(x) \in G_1^{G_2\gamma_i(x)}$,*

$$k_i(x) = \rho_2(g_2)k_i(x)\rho_1(g_1)^{-1}. \tag{9.71}$$

Proof. Given a lifted kernel $k \in \hat{\Gamma}(K_D)$, the local kernels $k_i(x) = k(\gamma_i(x))$ satisfy conditions 1 and 2, for

$$\begin{aligned} k_i(x) &= k(\gamma_i(x)) \\ &= k(g^2_{ij}(x)\gamma_j(x)g^1_{ij}(x)) \\ &= \rho_2(g^2_{ij}(x))k(\gamma_j(x))\rho_1(g^1_{ij}(x)) \\ &= \rho_2(g^2_{ij}(x))k_j(x)\rho_1(g^1_{ij}(x)) \end{aligned} \tag{9.72}$$

and

$$\begin{aligned} \rho_2(g_2)k_i(x)\rho_1(g_1)^{-1} &= \rho_2(g_2)k_i(x)\rho_1(\gamma_i(x)^{-1}g_2\gamma_i(x))^{-1} \\ &= k(g_2\gamma_i(x)\gamma_i(x)^{-1}g_2^{-1}\gamma_i(x)) \\ &= k(\gamma_i(x)) \\ &= k_i(x). \end{aligned} \tag{9.73}$$

Conversely, if we are given a collection of local kernels k_i satisfying the two constraints, we may define

$$k(p) = k(g_2\gamma_i(x)g_1) = \rho_1(g_2)k_i(x)\rho_1(g_1). \tag{9.74}$$

We must verify that this definition is consistent when $x \in U_i \cap U_j$, that k satisfies the equivariance constraint $k(g_2 p g_1) = \rho_2(g_2)k(p)\rho_1(g_1)$, and that the definition does not depend on the choice of g_2, g_1 in the double coset decomposition $p = g_2\gamma_i(x)g_1$.

The lifted kernel satisfies the equivariance constraint, for:

$$\begin{aligned} k(g_2 p g_1) &= k(g_2 g_2' \gamma_i(x) g_1' g_1) \\ &= \rho_2(g_2 g_2')k_i(x)\rho_1(g_1' g_1) \\ &= \rho_2(g_2)\rho_2(g_2')k_i(x)\rho_1(g_1')\rho_1(g_1) \\ &= \rho_2(g_2)k(p)\rho_1(g_1). \end{aligned} \tag{9.75}$$

We verify that the definition of k does not depend on the choice of section. Because k is equivariant, it is sufficient to check this for $p = \gamma_i(x)$. So let $p = \gamma_i(x) = g_{ij}^2(x)\gamma_j(x)g_{ij}^1(x)$. Then

$$\begin{aligned} k(\gamma_i(x)) &= k(g_{ij}^2(x)\gamma_j(x)g_{ij}^1(x)) \\ &= \rho_2(g_{ij}^2(x))k_j(x)\rho_1(g_{ij}^1(x)) \\ &= k_i(x) \end{aligned} \tag{9.76}$$

So the definitions of k based on γ_i and γ_j agree.

The lifted kernel does not depend on how we decompose p. Let $p = g_2\gamma_i(x)g_1 = g_2'\gamma_i(x)g_1'$. Then $g_2^{-1}g_2' \in G_2^{\gamma_i(x)G_1}$ and $g_1' g_1^{-1} \in G_1^{G_2\gamma_i(x)}$, and these elements are conjugate: $\gamma_i(x)^{-1}g_2^{-1}g_2'\gamma_i(x) = (g_1' g_1^{-1})^{-1}$. From the two expressions for p we get two values for $k(p)$, namely $k(p) = \rho_2(g_2)k_i(x)\rho_1(g_1)$ and $\rho_2(g_2')k_i(x)\rho_1(g_1')$. By the invariance of the kernel k_i, these are equal:

$$\begin{aligned} \rho_2(g_2)k_i(x)\rho_1(g_1) &= \rho_2(g_2)\rho_2(g_2^{-1}g_2')k_i(x)\rho_1(g_1' g_1^{-1})\rho_1(g_1) \\ &= \rho_2(g_2')k_i(x)\rho_1(g_1') \end{aligned} \tag{9.77}$$

■

9.7 Examples

In this section we study to examples of homogeneous G-CNNs: spherical CNNs and SE(3) CNNs. The examples are based on (Cohen et al., 2018a).

9.7.1 The rotation group SO(3) and spherical CNNs

The group of 3D rotations $P = \mathrm{SO}(3)$ is a three-dimensional manifold that can be parameterized by ZYZ Euler angles $\alpha \in [0, 2\pi)$, $\beta \in [0, \pi]$ and $\gamma \in [0, 2\pi)$, i.e. $p = Z(\alpha)Y(\beta)Z(\gamma) \in \mathrm{SO}(3)$, (where Z and Y denote rotations around the Z and Y axes). For this example we choose $G = G_1 = G_2 = \mathrm{SO}(2) = \{Z(\alpha) \,|\, \alpha \in [0, 2\pi)\}$ as the group of rotations around the Z-axis, i.e. the stabilizer subgroup of the north pole of the sphere. A left G-coset is then a subset of SO(3) of the form

$$pG = \{Z(\alpha)Y(\beta)Z(\gamma)Z(\alpha') \,|\, \alpha' \in [0, 2\pi)\} = \{Z(\alpha)Y(\beta)Z(\alpha') \,|\, \alpha' \in [0, 2\pi)\}.$$

Thus, the coset space $B = P/G$ is the sphere S^2, parameterized by spherical coordinates α and β. The stabilizer G_x of a point $x \in S^2$ is the set of rotations around the axis through x, which (as expected) is isomorphic to $G = \mathrm{SO}(2)$ for all x.

What about the double coset space? The orbit of a point $x(\alpha, \beta) \in S^2$ under G is a circle around the Z axis at lattitude β, so the double coset space $G\backslash P/G$, which indexes these orbits, is the segment $[0, \pi]$ (see Fig. 9.1).

A section $s = s_i : P/G \to P$ may be defined (almost everywhere) as $s(\alpha, \beta) = Z(\alpha)Y(\beta) \in \mathrm{SO}(3)$. For the double coset space, we may define a global section $\gamma(\beta) = \gamma_j(\beta) = Y(\beta) \in \mathrm{SO}(3)$. Then the stabilizer $G_2^{\gamma(\beta)G_1}$ (for $\beta \in G\backslash P/G = [0, \pi]$) that was defined in Section 9.6.2 is the set of Z-axis rotations that leave the point $\gamma(\beta)G_1 = (0, \beta) \in S^2$ invariant. For the poles ($\beta = 0$ or $\beta = \pi$), this stabilizer is all of $G = \mathrm{SO}(2)$, but for other points it is the trivial subgroup $\{e\}$.

Thus, according to Theorem 9.6.2, the equivariant kernels are matrix-valued functions on the segment $[0, \pi]$, that are mostly unconstrained (except at the poles, where they should satisfy Eq. 9.71). As functions on P/G_1 (Theorem 9.6.1), they are matrix-valued functions $k_i : P/G_1 \to \mathrm{Hom}(V_1, V_2)$ satisfying $k_i(gx) = \rho_2(g)k_i(x)\rho_1(s(gx)^{-1}gs(x))^{-1}$ for $g \in \mathrm{SO}(2)$ and $x \in S^2$ (see Eq. 9.59). This says that as a function on the sphere k_i is determined on SO(2)-orbits $\{gx \,|\, g \in \mathrm{SO}(2)\}$ (lattitudinal circles around the Z axis) by its value on one point of the orbit. Indeed, if $\rho_1(g) = \rho_2(g) = \rho(g) = 1$ is the trivial representation, we see that k_i is constant on these orbits, in agreement with (Esteves et al., 2018) who use isotropic filters. For for ρ_1 a trivial or regular representation and ρ_2 a regular representation of SO(2), we recover the non-isotropic method of (Cohen et al., 2018c) (Chapter 4). For segmentation tasks, one can use a trivial representation for ρ_2 in the output layer to obtain a scalar feature map on S^2, analogous to (Winkens et al., 2018). Other choices, such as ρ the standard 2D representation of SO(2), would make it possible to build spherical CNNs that can process vector fields (Esteves et al., 2020).

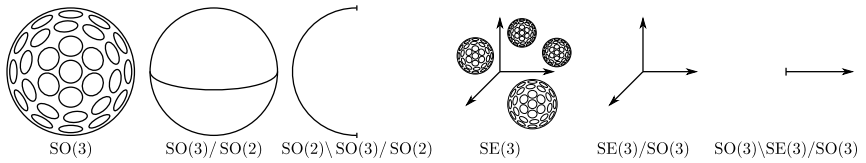


Figure 9.1: *Quotients of* SO(3) *and* SE(3).

9.7.2 The roto-translation group SE(3) and 3D Steerable CNNs

The group of rigid body motions $P = \mathrm{SE}(3)$ is a 6D manifold $\mathbb{R}^3 \rtimes \mathrm{SO}(3)$. In this example, we choose $G = G_1 = G_2 = \mathrm{SO}(3)$ (rotations around the origin). A left G-coset is a set of the form $pG = trG = \{trr' | r' \in \mathrm{SO}(3)\} = \{tr | r \in \mathrm{SO}(3)\}$ where t is the translation component of p. Thus, the coset space P/G is $\mathbb{R}^3$. The stabilizer G_x of a point $x \in \mathbb{R}^3$ is the set of rotations around x, which is isomorphic to SO(3). The orbit of a point $x \in \mathbb{R}^3$ is a spherical shell of radius $\|x\|$, so the double coset space $G\backslash P/G$, which indexes these orbits, is the set of radii $[0, \infty)$.

Since SE(3) is a trivial principal SO(3) bundle, we can choose a global section $s = s_i : P/G \to P$ by taking $s(x)$ to be the translation by x. As double coset representatives we can choose $\gamma(\|x\|) = \gamma_j(\|x\|)$ to be the translation by $(0, 0, \|x\|)$. Then the stabilizer $G_2^{\gamma(\|x\|)G_1}$ for $\|x\| \in G\backslash P/G$ is the set of rotations around Z, i.e. SO(2), except for $\|x\| = 0$, where it is SO(3).

For any representations ρ_1, ρ_2, the equivariant maps between sections of the associated vector bundle are given by convolutions with matrix-valued kernels on $\mathbb{R}^3$ that satisfy $k(rx) = \rho_2(r)k(x)\rho_1(r^{-1})$ for $r \in \mathrm{SO}(3)$ and $x \in \mathbb{R}^3$. This follows from Theorem 9.6.1 (Eq. 9.59) with the simplification $g_{ij}^r(x) = r$ for all $r \in G$, because SE(3) is a semidirect product. Alternatively, we can define the kernel as a function on the double coset space $G\backslash P/G = [0, \infty)$. According to theorem 9.6.2, this kernel $k_j : [0, \infty) \to \mathrm{Hom}(V_1, V_2)$ should satisfy $k_j(\|x\|) = \rho_2(r)k_j(x)\rho_1(r)$ for $r \in \mathrm{SO}(2)$ and $\|x\| > 0$ (and $r \in \mathrm{SO}(3)$ for $\|x\| = 0$). This is in agreement with the results obtained by Weiler et al., 2018a.

10

Gauge CNNs on Manifolds

In Chapter 8 we gave a general definition of G-CNN representation spaces as spaces of sections of an associated vector bundle (i.e. fields on a manifold), and network layers as linear and non-linear maps between such spaces. In the previous chapter we studied the special case of linear equivariant maps between *homogeneous* representation spaces, and showed that they correspond to generalized convolutions. Here we perform a similar analysis for general representation spaces (fields on a general manifold) with gauge symmetry. In the abstract, the present situation is very similar in that we have two representation spaces, a group of symmetries acting by principal bundle automorphisms, and a space of linear equivariant maps between the representation spaces. However, the group of gauge transformations is infinite dimensional and we will find that naively expressing a linear map between representation spaces as an integral transform and requiring gauge equivariance as we did before leads to an overly restrictive constraint on the kernel. This motivates the introduction of a *connection* that determines parallel transport and can be used to alleviate the kernel constraint. Finally, we also show how spatial weight sharing can be motivated in the case of a general Riemannian manifold by considering radial isometries as symmetries.

10.1 Gauge Transformations

As discussed in Chapter 7, gauge transformations can be defined in two ways. Firstly, as local gauge transformations, i.e. functions $g_{ij} : U \to G$ that relate two sections (i.e. gauges) s_i, s_j of a principal bundle P via $s_i(x) = s_j(x)g_{ij}(x)$. And secondly, as principal bundle (B, G)-automorphisms, i.e. diffeomorphisms $\alpha : P \to P$ satisfying $\alpha(pg) = \alpha(p)g$ and $\pi \circ \alpha = \pi$. A (B, G)-automorphism maps each fiber of P onto itself (i.e. fixes base space B) by an automorphism of the fiber (i.e. fixes fiber G as a principal homogeneous space). That these correspond to the same concept is revealed by the local representation of (B, G)-automorphisms as maps $(x, g) \mapsto (x, g^{\alpha}(x)g)$ (Proposition 8).

The set of (B, G)-automorphisms of P, denoted $\mathrm{Aut}_{B,G}(P)$ forms a group: we have $\mathrm{id}_P \in \mathrm{Aut}_{B,G}(P)$, for all $\alpha, \alpha' \in \mathrm{Aut}_{B,G}(P)$ we have $\alpha^{-1} \in \mathrm{Aut}_{B,G}(P)$ and $\alpha \circ \alpha' \in \mathrm{Aut}_{B,G}(P)$. This is the group of gauge transformations of P.

Depending on the nature of P, gauge transformations can be classed as internal or external symmetries, or a combination of both (Rudolph and Schmidt, 2017, Ch. 7). If P is the frame bundle of the tangent space of B, then the frames are related to the directions in the base space, and one speaks of external symmetries. Otherwise, if there is no such relation, one speaks of internal symmetries. An example of an internal symmetry in computer vision could be an independent change of basis of the RGB color space at each pixel. On the other hand if we consider a vector field (e.g. optical flow), a gauge transformation corresponds to a change of basis of the tangent space at each point, so this constitutes an external gauge symmetry (see Section 8.1.2 for detailed examples).

10.2 Gauge Invariant Kernels

In Chapter 9, Theorem 9.4.1, we proved that the two argument kernel k corresponding to a layer Φ is invariant to global symmetries S if and only if Φ is equivariant. The proof of this theorem relied only on the fact that symmetries act via principal bundle automorphisms and on the invariance of the integration measure. Gauge transformations also act by principal bundle automorphisms, so provided that we have an invariant measure on P, one could prove a similar theorem. In this section we study the meaning of gauge invariance and show that a gauge invariant kernel is one that has the same local representation regardless of the gauge used. Moreover we show that, at least for the kind of two-argument kernels we have studied so far, such gauge invariant kernels are necessarily trivial.

For now we assume, as in Chapter 9, that the two representation spaces $(P_n, B_n, G_n, \rho_n, V_n)$ for $n = 1, 2$ (Def. 8.1.1) satisfy $P_1 = P_2 = P$, $\eta = \mathrm{id}_P$, $\chi = \mathrm{id}_S$ and that G is compact. Only now P is a smooth principal bundle (e.g. the orthogonal frame bundle of a manifold) that is not necessarily homogeneous, and the symmetry group $S = \mathrm{Aut}_{B,G}(P)$ is the group of gauge transformations. As before, we consider maps $\Phi : \hat{\Gamma}(A_1) \to \hat{\Gamma}(A_2)$ of the form

$$\Phi f(p) = \int_{P_1} k(p,q) f(q) d\mu(q), \tag{10.1}$$

where $k : P \times P \to \mathrm{Hom}(V_1, V_2)$ is a lifted two-argument kernel, i.e. satisfies $k(p_2 g_2, p_1 g_1) = \rho_2(g_2)^{-1} k(p_2, p_1) \rho_1(g_1)$ (the actual bundle of kernels can be defined analogously to Def. 9.3.3, replacing the homogeneous principal bundle by a general one). We assume that the measure μ is a product of some measure on B and the invariant measure on G.

If we require Φ to be equivariant to gauge transformations $\alpha \in \mathrm{Aut}_{B,G}(P)$, we may reason as in the proof of Thm 9.4.1 to obtain a constraint of the form:

$$k(\alpha(p), \alpha(q)) = k(p,q). \tag{10.2}$$

Although this invariance constraint looks similar to the one found in the previous chapter, the fact that the symmetry group is now infinite dimensional means that this constraint is actually much stronger. To see this more clearly, we will represent the kernel locally on a piece U of the base space B using a section s of P, via $k_s(x,y) = k(s(x), s(y))$. By Proposition 8, a gauge transformation $\alpha \in \mathrm{Aut}_{B,G}(P)$ is locally represented relative to s by a map $g^{\alpha,s} : U \to G$ which we will denote by $g(x) = g^{\alpha,s}(x)$:

$$\alpha(s(x)g) = s(x)g(x)g. \tag{10.3}$$

This allows us to write down the local form of the transformed kernel:

$$\begin{aligned} k(\alpha(s(x)), \alpha(s(y))) &= k(s(x)g(x), s(y)g(y)) \\ &= \rho_2(g(x))^{-1} k(s(x), s(y)) \rho_1(g(y)) \\ &= \rho_2(g(x))^{-1} k_s(x,y) \rho_1(g(y)) \end{aligned} \tag{10.4}$$

By invariance of k, the left hand side of this equation equals $k(s(x), s(y)) = k_s(x,y)$, so the local form of the kernel constraint is

$$\rho_2(g(x))^{-1} k_s(x,y) \rho_1(g(y)) = k_s(x,y) \tag{10.5}$$

Note further that if we have another section s' of P, we can express k as

$$k_{s'}(x,y) = k(s'(x), s'(y)) = k(s(x)g(x), s(y)g(y)) = \rho_2(g(x))^{-1} k_s(x,y)\rho_1(g(y)). \tag{10.6}$$

This then provides us with a fundamental understanding of the meaning of gauge invariance and the relation between principal (B,G)-automorphisms and changing sections: a lifted kernel k is invariant to principal bundle (B,G)-automorphisms (gauge transformations) if and only if the local representations of k relative to different gauges s, s' are the same, i.e. $k_s = k_{s'}$ for all sections $s, s' : U \to P$. Any kernel can be expressed in any gauge, but only a gauge invariant one is represented by *the same* function $U \to \mathrm{Hom}(V_1, V_2)$ regardless of the gauge $s : U \to P$ we choose. Conceptually this means that the kernel treats all frames equivalently, i.e. respects the gauge symmetry.

Returning now to the invariance constraint Eq. 10.5, note that for $x \neq y$, $g(x)$ and $g(y)$ can vary independently. This makes for a very strong constraint on k_s. Indeed, the following theorem says that a kernel of this kind can only make use of scalar fields (i.e. trivial representations of G).

Theorem 10.2.1. *Let (ρ, V) and (ρ', V') be representations of G and $T \in \mathrm{Hom}(V, V')$. Then T satisfies $\rho'(g')^{-1} T \rho(g) = T$ for all $g, g' \in G$ if and only if T maps the trivial subspace of (ρ, V) to the trivial subspace of (ρ', V'), and everything else to zero.*

Proof. It suffices to study the irreducible case, so let (ρ_i, V_i) and (ρ'_j, V'_j) be irreducible subrepresentations of ρ and ρ' respectively, and let T_{ij} be the corresponding map from V_i to V'_j. If both ρ_i and ρ'_j are trivial, then T_{ij} satisfies $\rho'_j(g')T_{ij}\rho_i(g) = T_{ij}$. Now consider the case where at least one of ρ_i and ρ'_j is non-trivial. If T_{ij} satisfies the constraint, then we have in particular that $T_{ij}\rho_i(g) = T_{ij}$ and $\rho'_j(g)T_{ij} = T_{ij}$. But this can only be true if $T = 0$, for else we would have detected a trivial subrepresentation, which contradicts the assumptions of irreducibility and non-triviality. ∎

10.3 Gauge Invariant Kernels with Connection

In order to obtain a less restrictive kernel constraint, we need to introduce a new element, namely a connection. This allows us to define parallel transport along a curve (e.g. a geodesic), which as we will see can be used to alleviate the overly strong kernel constraint. We will briefly discuss the notion of parallel transport below, but for a detailed introduction see (Rudolph and Schmidt, 2012; Tu, 2017; Kobayashi and Nomizu, 1963).

We consider the following setting. We have two representation spaces denoted $(P_n, B_n, G_n, \rho_n, V_n)$ (for $n = 1, 2$) consisting of a smooth principal bundle, base space and structure group, and a representation of that structure group, which together determine $\Gamma(P_n \times_{\rho_n} V_n)$ i.e. the representation space proper. We further assume that we have a morphism $\eta : P_2 \to P_1$ which establishes a correspondence between positions and frames in P_2 and positions and frames in P_1 (see Definition 8.1.3). This morphism satisfies $\eta(pg) = \eta(p)\theta(g)$ for all $p \in P_2$ and $g \in G_2$, where $\theta : G_2 \to G_1$ is a group homomorphism. As discussed in Chapter 8, this affords the possibility of reducing the structure group, e.g. from $\mathrm{O}(d)$ to $\mathrm{SO}(d)$. A general morphism η also satisfies $\pi_1 \circ \eta = \pi_2 \circ \beta$ for some map $\beta : B_2 \to B_1$. We will assume that β is a diffeomorphism[1].

As mentioned, we assume also that we have a principal connection on P_1[2] that determines a parallel transport operation. On the principal bundle $P = P_1$ itself, parallel transport takes the form of an equivariant map $\Pi^\gamma : P_x \to P_y$ where $\gamma : [0, 1] \to B$ is a curve and $x = \gamma(0)$ and $y = \gamma(1)$. Locally on the base space, parallel transport is represented relative to a section $s : B \to P$ by a group element $g^s_\gamma \in G$ that maps a geometric object (e.g. vector) at x to one at y. When we change the section from s to s' (with $s'(x) = s(x)g(x)$) then the parallel transporter changes to $g^{s'}_\gamma = g(x)^{-1} g^s_\gamma g(y)$.

Using the connection we may also define *geodesics* as straightest curves between two points x and y in B. For close enough x, y, the geodesic is unique. We denote parallel transport along the unique geodesic connecting x and y by the shorthand $g^s_{x\to y}$.

Given the principal bundle morphism $\eta : P_2 \to P_1$ over $\beta : B_2 \to B_1$ and a section $s_2 : U_2 \to P_2$ (with $U_2 \subset B$) we may define a section $s_1 : U_1 \to P_1$ by setting $s_1 = \eta \circ s_2 \circ \beta^{-1}$. This is indeed a section for $\pi \circ \eta \circ s_2 \circ \beta^{-1} = \beta \circ \pi \circ s_2 \circ \beta^{-1} = \beta \circ \beta^{-1} = \mathrm{id}_{B_1}$. If we change s_2 to $s'_2(x) = s_2(x)g(x)$ then s_1 changes to $s'_1 = \eta \circ s'_2 \circ \beta^{-1}$ which is related to s_1 by $s'_1(x) = s_1(x)\theta(g(\beta^{-1}(x)))$.

Using this machinery we may define a linear operator

$$(\Phi f)_{s_2}(x) = \int_{U_1} k_{s_2 s_1}(x, y)\rho_1(g^{s_1}_{y\to\beta(x)}) f_{s_1}(y) d\mu(y). \tag{10.7}$$

Here $f_{s_1} : U_1 \to V_1$ is a local section expressed relative to s_1, and similarly for $(\Phi f)_{s_2}$. Similarly $k_{s_2 s_1} : U_1 \times U_2 \to \mathrm{Hom}(V_1, V_2)$ is a map which we will later interpret as the local representation of a section of some bundle of kernels. We

[1] If we assume local support of the kernel and input fields, it suffices to assume that β is injective with large enough range.

[2] Note that under some circumstances η induces a pullback connection on P_2.

assume that for every $x \in B_2$, the kernel $k_{s_2 s_1}(x, -)$ has support that is contained in U_1, so there is no need to integrate over all of B_1 by integrating over multiple charts and using a partition of unity. This assumption is realistic because the filters in actual CNNs are typically small. Finally we note that the introduction of the factor $\rho_1(g^{s_1}_{y \to \beta(x)})$ does not lead to a loss in generality because it is smooth and invertible, and so can be absorbed into the kernel (for a fixed gauge).

If we change the section to $s_2'(x) = s_2(x)g(x)$ (and thus change the other section s_1 to $s_1'(x) = s_1(x)\theta(g(\beta^{-1}(x)))$) we have

$$\begin{aligned} g^{s_1'}_{y \to \beta(x)} &= \theta(g(x))^{-1} g^{s_1}_{y \to \beta(x)} \theta(g(\beta^{-1}(y))) \\ f_{s_1'}(y) &= \theta(g(\beta^{-1}(y)))^{-1} f_{s_1}(y). \end{aligned} \tag{10.8}$$

Substituting these into Eq. 10.7, we find that if the kernel satisfies

$$k_{s_2' s_1'}(x, y) = \rho_2(g(x))^{-1} k_{s_2 s_1}(x, y) \rho_1(\theta(g(x))), \tag{10.9}$$

then we have $(\Phi f)_{s_2'}(x) = \rho_2(g(x))^{-1}(\Phi f)_{s_2}(x)$ as required for Φf to be well defined as a field existing independently of a choice of gauge.

The requirement that under a change of gauge, the kernel transforms as in Eq. 10.9 can be formalized by saying that it is a section of the following bundle:

Definition 10.3.1 (Bundle of two-argument linked kernels)**.** *Let $(P_i, B_i, G_i, \rho_i, V_i)$ (for $i = 1, 2$) define two representation spaces linked by a principal bundle morphism $\eta : P_2 \to P_1$ over $\theta : G_2 \to G_1$. The bundle of two-argument linked kernels is defined as*

$$K_\eta = (P_2 \times P_1) \times_{\tilde{\rho}} \mathrm{Hom}(V_1, V_2), \tag{10.10}$$

where $\tilde{\rho}$ is the following representation of $G_2 \times G_1$:

$$\tilde{\rho}(g_2, g_1)A = \rho_2(g_2)^{-1} A \rho_1(\theta(g_2)). \tag{10.11}$$

Definition 10.3.2 (Linked two-argument kernel)**.** *A linked two-argument kernel is a section of the bundle K_η. Such a kernel can be represented in three equivalent ways:*

1. *As a section $\kappa \in \Gamma(K_\eta)$, that is a map $\kappa : B_2 \times B_1 \to K_\eta$ satisfying $\pi_{K_\eta} \circ \kappa = \mathrm{id}_{B_2 \times B_1}$.*
2. *As a lifted section $k \in \hat{\Gamma}(K_\eta)$, that is a map $P_2 \times P_1 \to \mathrm{Hom}(V_1, V_2)$ satisfying for $p \in P_2$, $q \in P_1$, $g_1 \in G_1$ and $g_2 \in G_2$:*

$$k(pg_1, qg_2) = \tilde{\rho}(g_2, g_1)^{-1} k(p, q) = \rho_2(g_2) k(p, q) \rho_1(\theta(g_2))^{-1}. \tag{10.12}$$

3. *As a collection of maps* $k_{s_2 s_1} : B_2 \times B_1 \to \mathrm{Hom}(V_1, V_2)$ *defined as:*

$$k_{s_2 s_1}(x, y) = k(s_2(x), s_1(y)), \tag{10.13}$$

where s_1, s_2 *are sections of* P_1, P_2 *respectively, and* $k \in \hat{\Gamma}(K_\eta)$.

We have shown that the kernel for any operator of the form Eq. 10.7 should be a section of K_η, so that it has the right transformation behaviour under a change of gauge. However, we have not considered these gauge transformations as symmetries yet. For that we choose as symmetries $S_n = \mathrm{Aut}_{B_n, G_n}(P_n)$ the groups of gauge transformations for the input and output space $n = 1, 2$. As shown in Section 8.2.7, the morphism $\eta : P_2 \to P_1$ induces a compatible group homomorphism $\chi : S_2 \to S_1$ that satisfies $\eta \circ \alpha = \chi(\alpha) \circ \eta$ as required by the definition of linked layer (Def. 8.1.3). Hence there is no need to choose a χ and we simply use the one induced by η.

As discussed in Section 10.2 gauge invariance means that local manifestations of the kernel do not depend on the gauge. That is, for any choice of s_1, s_1', s_2, s_2' we have $k_{s_2 s_1} = k_{s_2' s_1'}$ or equivalently

$$k_{s_2 s_1}(x, y) = \rho_2(g_2) k_{s_2 s_1}(x, y) \rho_1(\theta(g_2))^{-1} \tag{10.14}$$

for all $g_2 \in G_2$.

By the same logic as used in Section 10.2 we can show that this is the local manifestation of $\mathrm{Aut}_{B,G}(P)$-invariance of the lifted kernel k. Consider a lifted kernel $k \in \hat{\Gamma}(K_\eta)$ that satisfies $k(\alpha(p), \chi_\alpha(q)) = k(p, q)$ for all $\alpha \in \mathrm{Aut}_{B_2, G_2}(P_2)$. The local form of the transformed kernel is:

$$\begin{aligned} k(\alpha(s_2(x)), \chi_\alpha(s_1(y))) &= k(s_2(x) g_2(x), s_1(y) g_1(y)) \\ &= \rho_2(g_2(x))^{-1} k(s_2(x), s_1(y)) \rho_1(\theta(g_2(x))) \\ &= \rho_2(g_2(x))^{-1} k_{s_2 s_1}(x, y) \rho_1(\theta(g_2(x))), \end{aligned} \tag{10.15}$$

where $g_2 = g^{\alpha, s_2}$ and $g_1 = g^{\chi_\alpha, s_1}$ are the local representations of α and χ_α (Proposition 8). By invariance of k, the left hand side equals $k(s_2(x), s_1(y)) = k_{s_2 s_1}(x, y)$, which yields once again the local form of the kernel constraint:

$$k_{s_2 s_1}(x, y) = \rho_2(g_2(x))^{-1} k_{s_2 s_1}(x, y) \rho_1(\theta(g_2(x))) = k_{s_2' s_1'}(x, y). \tag{10.16}$$

We have established that like before, for kernels in K_η it is also the case that $\mathrm{Aut}_{B,G}(P)$-invariance of k is the same thing as the local kernels $k_{s_2 s_1}$ being independent of the gauge and indeed this is true in general. However, the

constraint we have thus obtained is much less restrictive, because $k_{s_2 s_1}$ is now acted upon by the same group element $g_s(x)$ on both the left and the right, leading to a non-trivial kernel.

Although in this section we have worked locally, we note that it is also be possible to obtain the corresponding results in global form (at least if the structure group G_1 and principal bundle P_1 are compact so that the required integrals are well defined). In this case the lifted form of Φ is written as:

$$\Phi f(p) = \int_{P_1} k(p,q) f(\Pi^{x \to y}(\eta(p))) d\mu(q), \tag{10.17}$$

where $f : P_1 \to V_1$ is a lifted section and $k : P_2 \times P_1 \to \mathrm{Hom}(V_1, V_2)$ is a lifted kernel in $\hat{\Gamma}(K_\eta)$. One can then once again show under appropriate assumptions that $\alpha \circ \Phi = \Phi \circ \chi_\alpha$ if and only if $k(\alpha(p), \chi_\alpha(q)) = k(p,q)$.

10.4 Radial Isometries & Spatial weight sharing

In the first publication on gauge CNNs (Chapter 5; Cohen et al. (2019)) we not only assumed gauge invariance of the kernel, but also used weight sharing between different positions, resulting in a convolution-like operation with the kernel parameterized by a tangent vector. The same idea has been applied in other works on manifold CNNs (Masci et al., 2015; Boscaini et al., 2015). In general, our philosophy is that weight sharing should always be motivated by a symmetry, but whereas the global symmetries considered in Chapter 9 act transitively on the base space, gauge symmetries only constrain the kernel at a given pair of points $(x, y) \in B_2 \times B_1$, leading to a locally connected rather than convolutional layer. We thus need to introduce further symmetries to justify spatial weight sharing, and for this reason we introduce *radial isometries*.

The notion of an isometry is defined in terms of a metric, so in this section we assume that the base space is equiped with a metric, i.e. that B is a Riemannian manifold. For the principal bundle we will take $P = F_{\mathrm{SO}}(B)$, the bundle of oriented orthogonal frames of the tangent space of B. We will assume familiarity with Riemmanian geometry. Some good references are (Tu, 2017; Lee, 1997).

10.4.1 Isometries

A Riemannian manifold is a smooth manifold equiped with a metric $\langle \cdot, \cdot \rangle_x : T_x B \times T_x B \to \mathbb{R}$ on the tangent space $T_x B$ that varies smoothly with $x \in B$. One

can show that a choice of metric is equivalent to a reduction of the frame bundle to $\mathrm{O}(d)$ (see Section 7.2.15). The metric can be used to define a distance function $d : B \times B \to \mathbb{R}$.

Definition 10.4.1. *We say that a map $\beta : B \to B'$ preserves the metric if:*

$$\langle \beta_{*x} v, \beta_{*x} w \rangle_{\beta(x)} = \langle v, w \rangle_x, \tag{10.18}$$

where $\beta_{*x} : T_x B \to T_{\beta(x)} B'$ is the differential of β at x, and $v, w \in T_x B$.

Definition 10.4.2. *An isometry is a metric-preserving diffeomorphism. The set of isometries of a Riemmanian manifold B forms a group denoted* $\mathrm{Isom}(B) \subset \mathrm{Diff}(B)$.

Proposition 11. *isometries preserve distances:*

$$d(\beta(x), \beta(y)) = d(x, y). \tag{10.19}$$

The fact that isometries are a special kind of diffeomorphism tells us that like diffeomorphisms, isometries can be lifted to automorphisms of the frame bundle $F_{\mathrm{GL}}(TB)$ in a unique way. Indeed it is clear from the above definitions that unlike general diffeomorphisms, they can be lifted to the orthogonal frame bundle $F_{\mathrm{O}}(TB)$ or even $F_{\mathrm{SO}}(TB)$ if it is orientation-preserving.

Isometries are very tractable (finite dimensional, compact structure group), but many manifolds of interest in applications do not admit a transitive group of isometries (or any non-trivial isometries at all). For this reason, we define in the next section a weaker notion of *radial isometry*. This concept preserves the following important property of isometries.

Theorem 10.4.1. *Let $\beta : B \to B'$ be an isometry, and $x \in B$. Assume that the exponential maps $\exp_x$ and $\exp_{\beta(x)}$ on B resp. B' are defined in neighbourhoods $V \subset T_x B$ and $U \subset T_{\beta(x)} B'$, and assume that β_{*x} maps V into U. Then the following diagram commutes:*

$$\begin{array}{ccc} V & \xrightarrow{\beta_{*x}} & U \\ {\scriptstyle \exp_x}\downarrow & & \downarrow{\scriptstyle \exp_{\beta(x)}} \\ B & \xrightarrow{\beta} & B' \end{array} \tag{10.20}$$

In other words, on these domains we have $\beta \circ \exp_x = \exp_{\beta(x)} \circ \beta_{*x}$.

Proof. See Tu (2017), Theorem 15.2. ∎

10.4.2 Radial Isometries

In general, convolutional weight sharing arises from symmetry principles. If we have a transitive symmetry, then we will have weight sharing between filters at every point in the manifold. However, general Riemmanian manifolds B may not have a transitive group of isometries or even any non-trivial ones. In order to still derive a weight sharing scheme from symmetry principles in these cases, we define in this section a weaker notion, called radial isometry.

The basic idea is as follows. Consider a point $x \in B$ with a neighbourhood $U = \{x' \in B \mid d(x, x') < r\}$. We call such a neighbourhood (x, U) a geodesic ball of radius r, centered on x. Now if we have another such point with neighbourhood (y, V), we can ask if there is an isometry that maps x to y and U to V. In general this is not possible. For instance if we consider a disk in the plane and a curved disk on the sphere. However, for small enough radii, we can still always find a map that sends x to y and U to V, such that distances of points $x' \in U$ to the center point x are preserved. Such maps we call radial isometries.

To weaken the notion of isometry, we let go of the requirement that β is metric-preserving, requiring only that it preserve the metric at the center point of a geodesic neighbourhood. In addition, we require that β still commutes with the exponential map.

Definition 10.4.3 (Radial Isometry)**.** *Let $x, x' \in B$ and $U, U' \subset B$ be geodesic balls of radius r centered on x and x', respectively. Assume that r is small enough so that the exponential map defines a bijection between a region of the tangent spaces and the neighbourhoods.*

A radial isometry $(x, U) \mapsto (x', U')$ is a diffeomorphism $\beta : U \to U'$ with $\beta(x) = x'$ and

$$\begin{aligned} \beta \circ \exp_x &= \exp_{\beta(x)} \circ \beta_{*x} \\ \langle \beta_{*x} v, \beta_{*x} w \rangle_{x'} &= \langle v, w \rangle_x. \end{aligned} \tag{10.21}$$

The first equation is valid for tangent vectors within the injective radius of the exponential map.

That is, a radial isometry is a diffeomorphism that commutes with the exponential map and induces a linear isometry between the tangent spaces $T_x B$ and $T_{\beta(x)} B$. Notice that we only require β to preserve the metric in the tangent space at the center points x, x', so a radial isometry is in general not an isometry (which preserves the metric at all points).

Definition 10.4.4 (Composable radial isometries)**.** *We will say that two radial isometries $\beta^1 : (x, U) \to (x', U')$ and $\beta^2 : (y, V) \to (y', V')$ are composable if $\beta^1(x) =$*

y, and $U' = V$. The composite is written $\beta^2 \circ \beta^1 : (x, U) \to (y', V)$.

One could define a more general notion by requiring that $U' \subseteq V$ instead of $U' = V$, but we will not need this. We will show that the composition of radial isometries is a radial isometry, but first we will prove the following elementary result about diffeomorphisms.

Proposition 12. *For a diffeomorphism $\beta : M \to N$, we have $(\beta_{*x})^{-1} = (\beta^{-1})_{*x'}$, where $x' = \beta(x)$.*

Proof. By the chain rule, we have

$$\begin{aligned} (\beta^{-1} \circ \beta)_{*x} &= (\beta^{-1})_{*x'} \circ \beta_{*x} = \mathrm{id}_{T_x M} \\ (\beta \circ \beta^{-1})_{*x'} &= \beta_{*x} \circ (\beta^{-1})_{*x'} = \mathrm{id}_{T_{x'} N} \end{aligned} \tag{10.22}$$

Since $(\beta^{-1})_{*x'}$ is both a right and left inverse of β_{*x}, it is equal to the unique inverse $(\beta_{*x})^{-1}$. ∎

Proposition 13. *The inverse of a radial isometry is also a radial isometry, and the composition of two composable radial isometries $(x, U) \mapsto (x', U')$ and $(x', U') \mapsto (x'', U'')$ is again a radial isometry $(x, U) \mapsto (x'', U'')$.*

Proof. Inverse: Let $\beta : (x, U) \to (x', U')$ be a radial isometry. Since β is a diffeomorphism, it has an inverse $\beta^{-1} : U' \to U$ that is also a diffeomorphism. The inverse satisfies $\beta^{-1}(x') = x$.

Next, we show that the inverse commutes with the exponential map. Since β commutes with the exponential map, we have $\beta_{*x} = \exp_{x'}^{-1} \circ \beta \circ \exp_x$. Inverting this linear isometry, we find that $\exp_x^{-1} \circ \beta^{-1} \circ \exp_{x'} = (\beta_{*x})^{-1} = (\beta^{-1})_{*x'}$ by Proposition 12. Composing with $\exp_x$ on the left we find that indeed β^{-1} commutes with the exponential map:

$$\beta^{-1} \circ \exp_{x'} = \exp_x \circ (\beta^{-1})_{*x'}. \tag{10.23}$$

The inverse β^{-1} induces a linear isometry $(\beta^{-1})_{*x'} : T_{x'} B \to T_x B$:

$$\begin{aligned} \langle (\beta^{-1})_{*x'} v, (\beta^{-1})_{*x'} w \rangle_x &= \langle (\beta_{*x})^{-1} v, (\beta_{*x})^{-1} w \rangle_x \\ &= \langle v, w \rangle_{x'}. \end{aligned} \tag{10.24}$$

The first step uses Proposition 12 and the second step uses the fact that β_{*x} is a linear isometry and hence its inverse is a linear isometry as well. The above two equations show that, β^{-1} is a radial isometry from (x', U') to (x, U).

Composition: Let $\beta^1 : (x, U) \to (x', U')$ and $\beta^2 : (x', U') \to (x'', U'')$ be two radial isometries. Then $\beta = \beta^2 \circ \beta^1$ is a diffeomorphism from $U \to U''$ that satisfies $\beta(x) = x''$. It commutes with the exponential map:

$$\begin{aligned} \beta^2 \circ \beta^1 \circ \exp_x &= \beta^2 \circ \exp_{x'} \circ \beta^1_{*x} \\ &= \exp_{x''} \circ \beta^2_{*x'} \circ \beta^1_{*x} \\ &= \exp_{x''} \circ (\beta^2 \circ \beta^1)_{*x}, \end{aligned} \tag{10.25}$$

and induces a linear isometry $\beta_{*x} : T_x B \to T_{x''} B$:

$$\begin{aligned} \langle (\beta^2 \circ \beta^1)_{*x} v, (\beta^2 \circ \beta^1)_{*x} w \rangle_{x''} &= \langle \beta^2_{*x'} \circ \beta^1_{*x} v, \beta^2_{*x'} \circ \beta^1_{*x} w \rangle_{x''} \\ &= \langle \beta^1_{*x} v, \beta^1_{*x} w \rangle_{x'} \\ &= \langle v, w \rangle_x. \end{aligned} \tag{10.26}$$

■

Remark. *Proposition 13 tells us that any collection of radial isometries that is closed under composition and inverses forms a* groupoid. *A groupoid is defined as a category where every morphism is an isomorphism. A radial isometry groupoid (RI Groupoid) has as objects patches with basepoint (x, U) and as morphisms the radial isometries between them. Groupoid composition is given by composition of radial isometries. We leave this here merely as a remark and will not use groupoid theory in the sequel.*

Note that there are many ways to make a RI groupoid, so we cannot speak of *the* RI groupoid. For one thing, we can vary the radius r of neighbourhoods, include only patches of one radius or multiple, and if we loosen the definition of the neighbourhoods being geodesic balls, even vary their shape. In practice, it will be most convenient to work with the RI groupoid that contains all RIs between geodesic balls of one given radius r.

Proposition 14. *Radial isometries $\beta : (x, U) \to (x', U')$ between radius-r geodesic balls are in one-to-one correspondence with linear isometries $A : T_x B \to T_{x'} B$.*

Proof. Given a radial isometry β, we get a linear isometry $A = \beta_{*x} : T_x B \to T_{x'} B$. Conversely, given a linear isometry $A : T_x B \to T_{x'} B$, we can define the map

$$\beta = \exp_{x'} \circ A \circ \exp_x^{-1}. \tag{10.27}$$

The differential of this map is

$$\beta_{*x} = (\exp_{x'})_{*0} \circ A_{*0} \circ (\exp_x^{-1})_{*x} = A \tag{10.28}$$

Here we used the chain rule (with the fact that $\exp_x^{-1}(x) = 0$, and A is linear so $A0 = 0$), and the fact that the differential of $\exp_x : T_xB \to B$ at $0 \in T_xB$ is the identity (and similarly the differential of $\exp_x^{-1}$ at x is the identity).

Since $\beta_{*x} = A$ and A is a linear isometry, β_{*x} is a linear isometry as required for β to be a radial isometry. It is also clear that β commutes with $\exp_x$, because $\beta \circ \exp_x = \exp_{x'} \circ A = \exp_{x'} \circ \beta_{*x}$. Thus, β is a radial isometry. ∎

Proposition 15. *There exists a radial isometry between any two geodesic balls of radius r, provided that r is less than the injective radius of the exponential map.*

Proof. Let (x, U) and (x', U') be geodesic balls of radius r centered at $x \in B$ and $x' \in B$ respectively. The tangent spaces T_xB and $T_{x'}B$ are inner product spaces of the same dimension, and as such there exists a linear isometry $A : T_xB \to T_{x'}B$ between them. By Proposition 14, this means that there exists a radial isometry between (x, U) and (x', U). ∎

Proposition 16. *In Riemannian normal coordinates, a radial isometry corresponds to a rotation matrix.*

Proof. Let $w_x : \mathbb{R}^d \to T_xB$ and $w_{x'} : \mathbb{R}^d \to T_{x'}B$ be *orthonormal* frames. Riemannian normal coordinates for geodesic neighbourhoods are defined using charts $\varphi : \mathbb{R}^d \to U$ as $\varphi_U(v) = \exp_x w_x v$ and $\varphi_{U'}(v) = \exp_{x'} w_{x'} v$. Expressing $\beta = \exp_{x'} \circ A \circ \exp_x^{-1}$ in these coordinates, we find:

$$\begin{aligned} \varphi_{U'}^{-1} \circ \beta \circ \varphi_U(v) &= w_{x'}^{-1} \circ \exp_{x'}^{-1} \circ \exp_{x'} \circ A \circ \exp_x^{-1} \circ \exp_x \circ w_x \\ &= w_{x'}^{-1} \circ A \circ w_x \\ &= [A] \end{aligned} \tag{10.29}$$

where $[A]$ is the matrix representation of A, which is orthogonal because A is a linear isometry and w_x and $w_{x'}$ are orthonormal frames. ∎

Theorem 10.4.2. *A diffeomorphism $\beta : B \to B$ of a Riemannian manifold B is an isometry if and only if for every $x \in B$ there exists a neighbourhood $U \subset B$ of x such that the restriction $\beta|_U : (x, U) \to (y, V)$ is a radial isometry.*

Proof. Assume β is an isometry. Then β_{*x} is a linear isometry, and by Theorem 15.2 of Tu (2017), $\beta \circ \exp_x = \exp_{\beta(x)} \beta_{*x}$. So it restricts to a radial isometry at every x.

Now assume β restricts to a radial isometry $\beta_U : (x, U) \to (y, V)$ at every $x \in B$. This means that for every $x \in B$, we have $\langle \beta_{*x} v, \beta_{*x} w \rangle_y = \langle v, w \rangle_x$ for every $v, w \in T_xB$. A map with this property is called an isometry. ∎

Since isometries are also radial isometries, we deduce that if there exists an isometry between two neighbourhoods (U, x) and (U', x'), then there exists a radial isometry between them. It is not true however that any radial isometry between such neighbourhoods is an isometry.

10.5 Kernel in Normal Coordinates

Consider again the operator Φ defined in Eq. 10.7. For simplicity, we will assume from hereon that $P_2 = P_1 = P$ and $\eta = \mathrm{id}_P$.

If $k_s(x, y) = k(s(x), s(y))$ (for $k \in \hat{\Gamma}(K_\eta)$) is locally supported, i.e. $k_s(x, y) = 0$ whenever $d(x, y) > r$ for some small enough radius r, then we can write y as $y = \exp_x V$ for some $V \in T_x B$. Indeed, if we choose a gauge $w_x : \mathbb{R}^d \to T_x B$, then we can write y using Riemannian normal coordinates v^i as $y(v) = \exp_x w_x v$.

We can thus write the linear operator (Eq. 10.7) as

$$(\Phi f)_s(x) = \int_{\mathbb{R}^d} k_s(x, y(v)) \rho(g^s_{y(v)\to x}) f_s(y(v)) \sqrt{|g|}\, dv \tag{10.30}$$

This operation is similar to the one introduced in chapter 5, except that the kernel depends on position x. Dropping this dependence can be motivated by invariance to radial isometries, as we will show next.

10.6 Invariance to Radial Isometries

We will consider radial isometries between geodesic balls of radius r, where r is chosen small enough so that there exists a radial isometry between any two such geodesic balls for a given manifold B. As a consequence, a kernel that is invariant to all of these radial isometries will be independent of position.

Recall from Theorem 7.2.1 that any diffeomorphism $\beta : B \to B$ lifts uniquely to a principal bundle automorphism $\alpha : F_{\mathrm{GL}}(TB) \to F_{\mathrm{GL}}(TB)$ of the frame bundle of B. The fiber $F_{\mathrm{GL}}(T_x B)$ of the frame bundle at x consists of frames $w_x : \mathbb{R}^n \to T_x B$ where n is the dimension of B. A section $s : U \to F_{\mathrm{GL}}(TB)$ on $U \subset B$ is thus a map that returns a map, i.e. $s(x) : \mathbb{R}^n \to T_x B$. The local representation $g^\alpha_{ij} : U \to \mathrm{GL}(\mathbb{R}^n)$ relative to gauges s_i, s_j is given by $g^\alpha_{ij}(x) = s_j(\beta(x))^{-1} \circ \beta_{*x} \circ s_i(x)$. For s_i, s_j orthogonal frames, $g^\alpha_{ij}(x)$ will be orthogonal.

Consider now two sections $s_i : U_i \to F_{\mathrm{GL}}(TB)$, $s_j : U_j \to F_{\mathrm{GL}}(TB)$, a locally supported kernel $k \in \hat{\Gamma}(K_\eta)$ (for $\eta = \mathrm{id}$; Def. 10.3.2) and a radial isometry

$\beta : (x, U_i) \to (\beta(x), U_j)$ that lifts to $\alpha : F_{\mathrm{GL}}(TU_i) \to F_{\mathrm{GL}}(TU_j)$. We require that $k(\alpha(p), \alpha(q)) = k(p, q)$ for any $p \in \pi^{-1}(x)$ and $q \in F_{\mathrm{GL}}(U_i)$. The local representation of the transformed kernel is

$$\begin{aligned} k(\alpha(s_i(x)), \alpha(s_i(y))) &= k(s_j(\beta(x))g_{ij}^{\alpha}(x), s_j(\beta(y))g_{ij}^{\alpha}(y)) \\ &= \rho_2(g_{ij}^{\alpha}(x))k(s_j(\beta(x)), s_j(\beta(y)))\rho_1(g_{ij}^{\alpha}(x))^{-1} \\ &= \rho_2(g_{ij}^{\alpha}(x))k_{s_j}(\beta(x), \beta(y))\rho_1(g_{ij}^{\alpha}(x))^{-1}. \end{aligned} \tag{10.31}$$

So we obtain the following constraint on local kernels:

$$k_{s_i}(x, y) = \rho_2(g_{ij}^{\alpha}(x))k_{s_j}(\beta(x), \beta(y))\rho_1(g_{ij}^{\alpha}(x))^{-1}. \tag{10.32}$$

We may express y in terms of normal coordinates centered on x, i.e. $y = \exp_x s(x)v$ where $v \in \mathbb{R}^n$. We can define the kernel directly in terms of these variables by $\bar{k}_{s_i}(x, v) = k_{s_i}(x, \exp_x s_i(x)v)$. The kernel constraint the becomes

$$k_{s_i}(x, v) = \rho_2(g_{ij}^{\alpha}(x))k_{s_j}(\beta(x), g_{ij}^{\alpha}(x)^{-1}v)\rho_1(g_{ij}^{\alpha}(x))^{-1}. \tag{10.33}$$

Consider now a radial isometry β for which $g_{ij}^{\alpha}(x) = I$, the identity. Then the constraint reads $k_{s_i}(x, v) = k_{s_j}(\beta(x), v)$. Since we can relate any two points by such a β, we conclude that the kernels are independent of x. Hence we can define $\bar{k}_{s_i}(v) = \bar{k}_{s_i}(x, v)$. Finally, since we may choose the rotation $g_{ij}^{\alpha}(x)$ freely for the center point x when choosing a radial isometry, the final form of the kernel constraint becomes:

$$\tilde{k}_{s_i}(v) = \rho_2(g)\tilde{k}_{s_i}(g^{-1}v)\rho_1(g)^{-1}, \tag{10.34}$$

for $g \in \mathrm{SO}(n)$, in full agreement with the constraint found in Chapter 5.

11

Conclusion

Symmetries provide a strong inductive bias. Physicists have long known this, and indeed the concept of symmetry is considered to be of fundamental importance in modern physics. In pattern recognition, computer vision, machine learning and related fields, symmetry has historically taken a less prominent position. Indeed, although the value of invariant features has long been recognized and much work has been done in that area, in the machine learning and neural networks literature the topic of symmetry and equivariant representations has received relatively little attention until recently.

In this thesis I have presented my attempt at a systematic study of equivariant neural networks, with an emphasis on convolutional neural networks. We began in Chapter 2 by generalizing the classical translation equivariant CNN to a group equivariant CNN by replacing convolutions by group convolutions. In Chapter 3 we generalized to steerable G-CNNs, whose feature spaces can be interpreted as fields over a homogeneous space. Although in these first two chapters we demonstrated the ideas only for discrete plane groups, the mathematics applies equally well to any discrete or continuous group and homogeneous space. We made good on this promise in Chapter 4, where we presented spherical CNNs. The final generalization (for now) — to general manifolds and gauge symmetries — is presented in Chapter 5.

In my work I have often relied on mathematics because it affords precision and generality, and because the math-first way of working has a tendency to produce unique solutions, that is, solutions that are hard to vary without losing some desirable property (e.g. equivariance to symmetries). For instance,

there is really very little to choose when one wishes to correctly implement the Icosahedral CNN presented in Chapter 6. Although future works may well improve upon the method as applied to a particular problem, for example improving computational efficiency by making networks that are only approximately equivariant to certain rotations but not others, the hope is that the canonical version of the method will at least remain useful as a reference and foundation for future specializations, and thus have a longer shelf life.

In contrast to the currently fashionable approach to machine learning research, our methods were obtained not primarily through tinkering or rigorous experimentation but mostly derived from first principles. And, going against the advice of many, I have attempted to explain them in the most elegant rather than the most elementary manner. Despite this, we hope that the reader can see that in most cases, the resulting methods are highly practical, efficient, and effective[1]. This is amply demonstrated by the experiments reported in Chapters 2 – 6, as well as other papers I have co-authored.

In Part II of the thesis, we developed the general theory of G-CNNs on manifolds. Fiber bundles play a principal role in this theory, so we covered the relevant background in Chapter 7. We can then define the notions of G-CNN *representation space* and *network layer*, in Chapter 8. These theoretical idealizations provide a framework for designing and reasoning about equivariant CNNs, and attempt to capture as many geometrically sensible designs as possible, while excluding unsound designs. In Chapters 9 and 10, we use this framework to analyze the space of linear equivariant maps between representation spaces, and show that such maps can be written as an integral transform with a constrained kernel, that in the case of a transitive symmetry becomes a generalized convolution. Thus, such a convolution layer is universal.

11.1 Future work

Much work remains to be done, both on better algorithms and models and addressing new applications, and on branching out and further generalization. Several examples given in Chapter 8, such as a G-CNN for diffusion tensor images on $\mathbb{R}^3$ or vector fields on S^2, have not yet been implemented. Similarly, internal (e.g. color space) gauge symmetries have not been used yet as far as we know. Despite recent progress (Weiler and Cesa, 2019), some challenges

[1] The only exception is the spherical convolution via non-commutative FFT, which is hard to implement efficiently in practice. For now we recommend the far simpler and faster icosahedral CNN or another Platonic CNN for practical applications.

remain regarding the implementation of G-CNNs for continuous groups in a way that is both efficient and numerically accurate. Much work remains to be done on the implementation of gauge equivariant networks for general manifolds and meshes.

One limitation of all of the work presented in this thesis is that we presume that the symmetries and their action on the input space and output space are known a priori, so that equivariance can be hardwired into the network. In fact, as per Chapter 8, we need to specify the group, base manifold, principal and associated bundle for each representation space. It would be interesting to investigate if symmetries and related geometrical structures can be learned, for this could make it possible to exploit symmetries that are either unknown to the researcher, or those which are hard to describe mathematically, or present only approximately.

Ideological wars notwithstanding, it may be interesting to investigate if there is any connection between the convolution kernels studied in Part II and classical kernel methods and the theory of reproducing kernel Hilbert spaces. In this connection the work of Reisert and Burkhardt (2007) on equivariant matrix-valued kernels might provide a good starting point.

As highlighted in Chapter 8, G-CNNs employ much of the same mathematical machinery used in physics to model physical systems. Beyond helping physicists learn about G-CNN, such an analogy is useful because the wish to complete the analogy can lead to ideas flowing in both directions. For instance, the idea of color space symmetries mentioned above arose by asking the question "what is an example of an internal symmetry in computer vision"? A further example, which will likely take much more work to elaborate, is the concept of a *connection* and its curvature. Connections play a pivotal role in both relativity and gauge theory, and although the concept is lurking in the background, it does not seem to play as significant a role in the theory of G-CNNs yet. G-CNNs have matter fields (feature maps) but not force fields (connections), so what is a force field in a G-CNN?

If physics can serve as an inspiration, so too can pure mathematics. Category theory provides a far reaching generalization of equivariance, going by the name naturality. More specifically, a group representation is a certain kind of functor, and an equivariant map is a natural transformation. The concepts of functoriality and naturality provide the very barest notion of relatedness and analogy between structures far more general than G-spaces. By emphasizing the relations between objects, category theory has been very useful for organizing knowledge in mathematics, computer science, and increasingly many other domains (Fong and Spivak, 2018). And if it works for HIs, why not AIs?

BIBLIOGRAPHY

[ACM15] P Agrawal, J Carreira, and J Malik. "Learning to See by Moving". In: International Conference on Computer Vision (ICCV). 2015.

[Ade+17] Tameem Adel et al. "3D Scattering Transforms for Disease Classification in Neuroimaging". In: *NeuroImage : Clinical* 14 (Feb. 10, 2017), pp. 506–517. pmid: 28289601.

[Ama78] S Amari. "Feature spaces which admit and detect invariant signal transformations". In: *Proc. 4th Int. Joint Conf. Pattern Recognition*. 1978, pp. 452–456.

[Ans+14] F Anselmi et al. *Unsupervised Learning of Invariant Representations with Low Sample Complexity: The Magic of Sensory Cortex or a New Framework for Machine Learning?* MIT Center for Brains, Minds and Machines, 2014.

[Arm+17] Iro Armeni et al. "Joint 2D-3D-Semantic Data for Indoor Scene Understanding". In: (2017).

[ARP15] F Anselmi, L Rosasco, and T Poggio. *On Invariance and Selectivity in Representation Learning*. MIT Center for Brains, Minds and Machines, 2015.

[BCB14] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate". Sept. 1, 2014.

[BCV13] Y Bengio, A Courville, and P Vincent. "Representation Learning: A Review and New Perspectives". In: *IEEE Trans. Pattern Anal. Mach. Intell.* (Feb. 28, 2013), pp. 1–30.

[Bek+18] Erik J Bekkers et al. "Roto-Translation Covariant Convolutional Networks for Medical Image Analysis". Apr. 10, 2018.

[Bek19] Erik J. Bekkers. "B-Spline CNNs on Lie Groups". In: (Nov. 20, 2019). arXiv: 1909.12057 [cs, stat].

[BF17] Wouter Boomsma and Jes Frellsen. "Spherical Convolutions and Their Application in Molecular Modelling". In: *Advances in Neural Information Processing Systems 30*. Ed. by I Guyon et al. Curran Associates, Inc., 2017, pp. 3436–3446.

[BG16] David Balduzzi and Muhammad Ghifary. "Strongly-Typed Recurrent Neural Networks". In: *Proceedings of the 33rd International Conference on Machine Learning* 33 (2016).

[Ble81] David Bleecker. *Gauge Theory and Variational Principles*. y First printing edition. Reading, Mass: Addison-Wesley, Nov. 1, 1981. 179 pp.

[BM13] J Bruna and S Mallat. "Invariant Scattering Convolution Networks". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 35.8 (Aug. 2013), pp. 1872–1886.

[Bos+15] D. Boscaini et al. "Learning Class-specific Descriptors for Deformable Shapes Using Localized Spectral Convolutional Networks". In: *Computer Graphics Forum* 34.5 (Aug. 1, 2015), pp. 13–23.

[Bos+16] Davide Boscaini et al. "Learning Shape Correspondence with Anisotropic Convolutional Neural Networks". In: (Nips 2016), pp. 1–9.

[BR09] Lorenz C. Blum and Jean-Louis Reymond. "970 Million Druglike Small Molecules for Virtual Screening in the Chemical Universe Database GDB-13". In: *Journal of the American Chemical Society* 131.25 (July 1, 2009), pp. 8732–8733.

[Bro+16] Michael M Bronstein et al. "Geometric Deep Learning: Going beyond Euclidean Data". In: (Nov. 24, 2016).

[Bru+14] Joan Bruna et al. "Spectral Networks and Locally Connected Networks on Graphs". In: *International Conference on Learning Representations (ICLR)*. 2014. arXiv: 1312.6203.

[BT19] Benjamin Bloem-Reddy and Yee Whye Teh. "Probabilistic Symmetry and Invariant Neural Networks". In: (Jan. 17, 2019). arXiv: 1901.06082 [cs, stat].

[Car09] Nathan Carter. *Visual Group Theory*. 2009.

[Cec+09] T Ceccherini-Silberstein et al. "Induced Representations and Mackey Theory". In: *J. Math. Sci.* 156.1 (Jan. 1, 2009), pp. 11–28.

[CGW18a] Taco Cohen, Mario Geiger, and Maurice Weiler. "A General Theory of Equivariant CNNs on Homogeneous Spaces". In: (Nov. 5, 2018). arXiv: 1811.02017 [cs, stat].

[CGW18b] Taco S Cohen, Mario Geiger, and Maurice Weiler. "Intertwiners between Induced Representations (with Applications to the Theory of Equivariant Neural Networks)". Mar. 28, 2018.

[Cha+] Angel X Chang et al. "ShapeNet: An Information-Rich 3D Model Repository". In: (), p. 11.

[CK01] G S Chirikjian and A B Kyatkin. *Engineering Applications of Noncommutative Harmonic Analysis*. 1st ed. CRC Press, May 2001.

[Coh+17] Taco S Cohen et al. "Convolutional Networks for Spherical Signals". In: ICML Workshop on Principled Approaches to Deep Learning. 2017.

[Coh+18] Taco S Cohen et al. "Spherical CNNs". In: International Conference on Learning Representations (ICLR). 2018.

[Coh+19] Taco S. Cohen et al. "Gauge Equivariant Convolutional Networks and the Icosahedral CNN". In: (May 13, 2019). arXiv: 1902.04615 [cs, stat].

[Coh13] T Cohen. "Learning Transformation Groups and Their Invariants". In: (2013).

[Cra14] Keenan Crane. "Discrete Differential Geometry: An Applied Introduction". In: 30664 (2014), pp. 1–6.

[CUH15] D Clevert, T Unterthiner, and S Hochreiter. "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)". In: *arXiv:1511.07289v3* (2015).

[CW14] T Cohen and M Welling. "Learning the Irreducible Representations of Commutative Lie Groups". In: Proceedings of the 31st International Conference on Machine Learning (ICML). Vol. 31. 2014, pp. 1755–1763.

[CW15a] T S Cohen and M Welling. "Transformation Properties of Learned Visual Representations". In: *International Conference on Learning Representations (ICLR)* (2015).

[CW15b] Taco S Cohen and Max Welling. "Harmonic Exponential Families on Manifolds". In: Proceedings of the 32nd International Conference on Machine Learning (ICML). 2015, pp. 1757–1765.

[CW16] Taco S Cohen and Max Welling. "Group Equivariant Convolutional Networks". In: Proceedings of The 33rd International Conference on Machine Learning (ICML). Vol. 48. 2016, pp. 2990–2999.

[CW17] Taco S Cohen and Max Welling. "Steerable CNNs". In: ICLR. 2017.

[DB07] Remco Duits and Bernhard Burgeth. "Scale Spaces on Lie Groups". In: *Scale Space and Variational Methods in Computer Vision*. Ed. by Fiorella Sgallari, Almerico Murli, and Nikos Paragios. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2007, pp. 300–312.

[DDK16] S Dieleman, J De Fauw, and K Kavukcuoglu. "Exploiting Cyclic Symmetry in Convolutional Neural Networks". In: International Conference on Machine Learning (ICML). 2016.

[De 29] Bruno De Finetti. "Funzione caratteristica di un fenomeno aleatorio". In: *Atti del Congresso Internazionale dei Matematici: Bologna del 3 al 10 de settembre di 1928*. 1929, pp. 179–190.

[DF84] Persi Diaconis and David Freedman. "Partial exchangeability and sufficiency". In: *Statistics: applications and new directions* (1984), pp. 205–236.

[DH94] J R Driscoll and D M Healy. "Computing Fourier Transforms and Convolutions on the 2-Sphere". In: *Adv. Appl. Math.* 15.2 (1994), pp. 202–250.

[Dia88] P Diaconis. *Group Representations in Probability and Statistics*. Hayward, CA: Institute of Mathematical Statistics, 1988. JSTOR: 4355560.

[DW19] Nichita Diaconu and Daniel E. Worrall. "Affine Self Convolution". In: (Nov. 18, 2019). arXiv: 1911.07704 [cs].

[DWA08] John B Drake, Pat Worley, and Eduardo D Azevedo. "Spherical Harmonic Transform Algorithms". In: X (2008), pp. 111–131.

[DWD15] S Dieleman, K W Willett, and J Dambre. "Rotation-Invariant Convolutional Neural Networks for Galaxy Morphology Prediction". In: *Mon. Not. R. Astron. Soc.* 450.2 (2015).

[Eat89] Morris L Eaton. *Group Invariance Applications in Statistics*. Institute of Mathematical Statistics, 1989.

[Eck+17] A. Eck et al. "Interpretation of Microbiota-Based Diagnostics by Explaining Individual Classifier Decisions". In: *BMC Bioinformatics* 18 (Oct. 4, 2017). pmid: 28978318.

[EMD20] Carlos Esteves, Ameesh Makadia, and Kostas Daniilidis. "Spin-Weighted Spherical CNNs". In: (2020).

[Est+18] Carlos Esteves et al. "Learning SO(3) Equivariant Representations with Spherical CNNs". In: 2018. arXiv: 1711.06721 [cs].

[Est20] Carlos Esteves. "Theoretical Aspects of Group Equivariant Neural Networks". In: (Apr. 2020). arXiv: 2004.05154 [cs.LG].

[Eti+11] Pavel Etingof et al. *Introduction to Representation Theory*. Providence, R.I: American Mathematical Society, July 26, 2011. 228 pp.

[FA91] W T Freeman and E H Adelson. "The Design and Use of Steerable Filters". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 13.9 (Sept. 1991), pp. 891–906.

[FM82] Kunihiko Fukushima and Sei Miyake. "Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position". en. In: *Pattern Recognit.* 15.6 (Jan. 1982), pp. 455–469.

[Fol95] G B Folland. *A Course in Abstract Harmonic Analysis*. CRC Press, 1995.

[FS18] Brendan Fong and David I Spivak. *Seven Sketches in Compositionality: An Invitation to Applied Category Theory*. 2018.

[GD14] R Gens and P Domingos. "Deep Symmetry Networks". In: Advances in Neural Information Processing Systems (NIPS). 2014.

[Goo+13] I J Goodfellow et al. "Maxout Networks". In: Proceedings of the 30th International Conference on Machine Learning (ICML). 2013, pp. 1319–1327.

[Gra14] B Graham. "Fractional Max-Pooling". In: *arXiv:1412.6071* (2014).

[Gre+94] H Greenspan et al. "Overcomplete Steerable Pyramid Filters and Rotation Invariance". In: *Proceedings of the Computer Vision and Pattern Recognition (CVPR)* (1994).

[Gro55] Alexandre Grothendieck. "A General Theory of Fibre Spaces with Structure Sheaf". In: (1955).

[Gur92] David Gurarie. *Symmetries and Laplacians: Introduction to Harmonic Analysis, Group Representations and Applications*. Elsevier B.V., 1992.

[Gut+08] B Gutman et al. "Shape Registration with Spherical Cross Correlation". In: *2nd MICCAI workshop* (2008).

[Gut+16] Nicholas Guttenberg et al. "Permutation-Equivariant Neural Networks Applied to Dynamics Prediction". In: (2016).

[GW09] Roe Goodman and Nolan R Wallach. *Symmetry, Representations, and Invariants*. Vol. 255. Graduate Texts in Mathematics. New York, NY: Springer New York, 2009.

[Ham17] Mark Hamilton. *Mathematical Gauge Theory: With Applications to the Standard Model of Particle Physics*. Universitext. Springer International Publishing, 2017.

[Har+18] Jason Hartford et al. "Deep Models of Interactions Across Sets". In: (Mar. 7, 2018). arXiv: 1803.02879 [cs, stat].

[He+16a] K He et al. "Deep Residual Learning for Image Recognition". In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016.

[He+16b] Kaiming He et al. "Identity Mappings in Deep Residual Networks". In: European Conference on Computer Vision (ECCV). 2016.

[Hea+03] D Healy et al. "FFTs for the 2-Sphere – Improvements and Variations". In: *J. Fourier Anal. Appl.* 9.4 (2003), pp. 340–385.

[HFS18] Geoffrey Hinton, Nicholas Frosst, and Sara Sabour. "Matrix Capsules with EM Routing". In: (2018).

[Hig+18] Irina Higgins et al. "Towards a Definition of Disentangled Representations". In: (Dec. 5, 2018). arXiv: 1812.02230 [cs, stat].

[HKW11] G E Hinton, A Krizhevsky, and S D Wang. "Transforming Auto-Encoders". In: *ICANN-11: International Conference on Artificial Neural Networks, Helsinki* (2011).

[HLW16] Gao Huang, Zhuang Liu, and Kilian Q Weinberger. "Densely Connected Convolutional Networks". 2016.

[Hoo+18] Emiel Hoogeboom et al. "HexaConv". In: International Conference on Learning Representations (ICLR). 2018.

[Hör03] Lars Hörmander. *The Analysis of Linear Partial Differential Operators I: Distribution Theory and Fourier Analysis*. 2nd ed. 2003 edition. Berlin ; New York: Springer, Aug. 13, 2003. 440 pp.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (Nov. 1, 1997), pp. 1735–1780.

[Hu+19] Jie Hu et al. "Squeeze-and-Excitation Networks". In: (May 16, 2019). arXiv: 1709.01507 [cs].

[Hus94] Dale Husemöller. *Fibre Bundles*. 3rd ed. Graduate Texts in Mathematics 20. New York: Springer-Verlag, 1994. 353 pp.

[Hy+18] Truong Son Hy et al. "Predicting Molecular Properties with Covariant Compositional Networks". In: *The Journal of Chemical Physics* 148.24 (June 27, 2018), p. 241745.

[IS15] S Ioffe and C Szegedy. "Batch Normalization : Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *arXiv:1502.03167v3* (2015).

[Jac+16] Jorn-Henrik Jacobsen et al. "Structured Receptive Fields in CNNs". In: Computer Vision and Pattern Recognition (CVPR). 2016.

[Jad+15] M Jaderberg et al. "Spatial Transformer Networks". In: Advances in Neural Information Processing Systems 28 (NIPS 2015). 2015.

[Jay+20] Siddhant M. Jayakumar et al. "Multiplicative Interactions and Where to Find Them". In: International Conference on Learning Representations (ICLR). 2020.

[Jia+18] Chiyu "Max" Jiang et al. "Spherical CNNs on Unstructured Grids". In: International Conference on Learning Representations (ICLR). 2018. arXiv: 1901.02039.

[Kan90] Kenichi Kanatani. *Group-Theoretical Methods in Image Understanding*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1990.

[KB15] D Kingma and J Ba. "Adam: A Method for Stochastic Optimization". In: Proceedings of the International Conference on Learning Representations (ICLR). 2015.

[Key91] Michael Keyl. "About the Geometric Structure of Symmetry-breaking". In: *Journal of Mathematical Physics* 32.4 (Apr. 1, 1991), pp. 1065–1071.

[KLT18] Risi Kondor, Zhen Lin, and Shubhendu Trivedi. "Clebsch–Gordan Nets: A Fully Fourier Space Spherical Convolutional Neural Network". In: *Neural Information Processing Systems (NeurIPS)*. 2018.

[KM07] Kai Krajsek and Rudolf Mester. "A Unified Theory for Steerable and Quadrature Filters". In: *Communications in Computer and Information Science* 4 CCIS (2007), pp. 201–214.

[KMS93] Ivan Kolář, Peter W. Michor, and Jan Slovák. *Natural Operations in Differential Geometry*. Berlin ; New York: Springer-Verlag, 1993. 434 pp.

[KN63] Shoshichi Kobayashi and Katsumi Nomizu. *Foundations of Differential Geometry (Volume 1)*. 1963.

[Kon+18] Risi Kondor et al. "Covariant Compositional Networks For Learning Graphs". Jan. 7, 2018.

[Kon07] R Kondor. "A Novel Set of Rotationally and Translationally Invariant Features for Images Based on the Non-Commutative Bispectrum". In: *arXiv:0701127* (2007).

[Kon08] R Kondor. "Group Theoretical Methods in Machine Learning". Columbia University, 2008.

[KP03] Stefan Kunis and Daniel Potts. "Fast Spherical Fourier Algorithms". In: *J. Comput. Appl. Math.* 161 (2003), pp. 75–98.

[KR07] Peter J Kostelec and Daniel N Rockmore. "SOFT: SO(3) Fourier Transforms". 2007.

[KR08] Peter J Kostelec and Daniel N Rockmore. "FFTs on the Rotation Group". In: *J. Fourier Anal. Appl.* 14.2 (2008), pp. 145–179.

[Kri09] Alex Krizhevsky. *Learning Multiple Layers of Features from Tiny Images*. University of Toronto, 2009.

[KSH12] A Krizhevsky, I Sutskever, and G Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Adv. Neural Inf. Process. Syst.* 25 (2012).

[KSJ14] A Kanazawa, A Sharma, and D Jacobs. "Locally Scale-Invariant Convolutional Neural Network". In: *Deep Learning and Representation Learning Workshop: NIPS* (2014), pp. 1–11.

[KT13] Eberhard Kaniuth and Keith F Taylor. *Induced Representations of Locally Compact Groups*. Cambridge University Press, 2013.

[KT18] Risi Kondor and Shubhendu Trivedi. "On the Generalization of Equivariance and Convolution in Neural Networks to the Action of Compact Groups". In: International Conference on Machine Learning (ICML). 2018.

[KV90] J J Koenderink and a J Van Doorn. "Receptive Field Families". In: *Biol. Cybern.* 63.4 (1990), pp. 291–297.

[KW11] Jyri J Kivinen and Christopher K I Williams. "Transformation Equivariant Boltzmann Machines". In: 21st International Conference on Artificial Neural Networks. June 2011.

[KW17] Thomas N Kipf and Max Welling. "Semi-Supervised Classification with Graph Convolutional Networks". In: 2017, pp. 1–14.

[Lar+07] H Larochelle et al. "An Empirical Evaluation of Deep Architectures on Problems with Many Factors of Variation". In: *Proceedings of the 24th International Conference on Machine Learning (ICML)* (2007).

[LB18] Gerton Lunter and Richard Brown. "An Equivariant Bayesian Convolutional Network Predicts Recombination Hotspots and Accurately Resolves Binding Motifs". In: (June 20, 2018).

[LCY14] M Lin, Q Chen, and S Yan. "Network In Network". In: *International Conference on Learning Representations (ICLR)* (2014).

[LeC+90] Yann LeCun et al. "Handwritten Digit Recognition with a Back-Propagation Network". In: *Advances in Neural Information Processing Systems 2*. Ed. by D S Touretzky. Morgan-Kaufmann, 1990, pp. 396–404.

[LeC+98] Y LeCun et al. "Gradient-Based Learning Applied to Document Recognition". In: *Proc. IEEE* 86.11 (1998).

[Lee+15] C Lee et al. "Deeply-Supervised Nets". In: Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics (AISTATS). Vol. 38. 2015, pp. 562–570.

[Lee03] John M. Lee. *Introduction to Smooth Manifolds*. en. Graduate Texts in Mathematics. New York: Springer-Verlag, 2003.

[Lee18] John Lee. *Introduction to Riemannian Manifolds*. 2nd ed. Graduate Texts in Mathematics. Springer International Publishing, 2018.

[Lee97] John M. Lee. *Riemannian Manifolds: An Introduction to Curvature*. Graduate Texts in Mathematics. New York: Springer-Verlag, 1997.

[Len89] Reiner Lenz. "Group-Theoretical Model of Feature Extraction". In: *J. Opt. Soc. Am. A* 6.6 (1989), pp. 827–834.

[Len90] Reiner Lenz. *Group Theoretical Methods in Image Processing*. Springer, Berlin, Heidelberg, 1990.

[LG16] A Lavin and S Gray. "Fast Algorithms for Convolutional Neural Networks". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 4013–4021.

[LGT15] C Lee, P W Gallagher, and Z Tu. "Generalizing Pooling Functions in Convolutional Neural Networks: Mixed, Gated, and Tree". In: *ArXiv:1509.08985* (2015).

[Lin+18] Jasper Linmans et al. "Sample Efficient Semantic Segmentation Using Rotation Equivariant Convolutional Networks". In: *FAIM/ICML Workshop: Towards Learning with Limited Labels: Equivariance, Invariance, and Beyond*. 2018.

[Liu+19] Min Liu et al. "Deep Learning 3D Shapes Using Alt-Az Anisotropic 2-Sphere Convolution". In: International Conference on Learning Representations (ICLR). 2019.

[Low04] D G Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". In: *Int. J. Comput. Vis.* 60.2 (Nov. 2004), pp. 91–110.

[LV15] K Lenc and A Vedaldi. "Understanding Image Representations by Measuring Their Equivariance and Equivalence". In: Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). 2015.

[Mac52] George W Mackey. "Induced Representations of Locally Compact Groups I". In: *Ann. Math.* 55.1 (1952), pp. 101–139.

[Mac53] George W Mackey. "Induced Representations of Locally Compact Groups II. The Frobenius Reciprocity Theorem". In: *Ann. Math.* 58.2 (1953), pp. 193–221.

[Mac68] George W Mackey. *Induced Representations of Groups and Quantum Mechanics*. New York-Amsterdam: W.A. Benjamin Inc., 1968.

[Mal12] Stephane Mallat. "Group Invariant Scattering". In: *Communications in Pure and Applied Mathematics* 65.10 (2012), pp. 1331–1398.

[Man+06] Siddharth Manay et al. "Integral Invariants for Shape Matching". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 28.10 (2006), pp. 1602–1617.

[Mar+17] Diego Marcos et al. "Rotation Equivariant Vector Field Networks". In: International Conference on Computer Vision (ICCV). 2017.

[Mar+19] Haggai Maron et al. "Invariant and Equivariant Graph Networks". In: *International Conference on Learning Representations (ICLR)*. 2019. arXiv: 1812.09902.

[Mas+15] Jonathan Masci et al. "Geodesic Convolutional Neural Networks on Riemannian Manifolds". In: (2015).

[Mas98] David K Maslen. "Efficient Computation of Fourier Transforms on Compact Groups". In: *J. Fourier Anal. Appl.* 4.1 (1998).

[Mat+19] Tambet Matiisen et al. "Teacher-Student Curriculum Learning". In: *IEEE Transactions on Neural Networks and Learning Systems* (2019), pp. 1–9. arXiv: [object0bject].

[MGD07] Ameesh Makadia, Christopher Geyer, and Kostas Daniilidis. "Correspondence-Free Structure from Motion". In: *Int. J. Comput. Vis.* 75.3 (Dec. 1, 2007), pp. 311–327.

[MH07] Roland Memisevic and Geoffrey Hinton. "Unsupervised Learning of Image Transformations". In: *2007 IEEE Conference on Computer Vision and Pattern Recognition* (June 2007), pp. 1–8.

[MHL14] M Mathieu, M Henaff, and Y LeCun. "Fast Training of Convolutional Networks through FFTs". In: International Conference on Learning Representations (ICLR). 2014.

[Mon+12] Grégoire Montavon et al. "Learning Invariant Representations of Molecules for Atomization Energy Prediction". In: *Advances in Neural Information Processing Systems 25*. Ed. by F Pereira et al. Curran Associates, Inc., 2012, pp. 440–448.

[Mon+16] Federico Monti et al. "Geometric Deep Learning on Graphs and Manifolds Using Mixture Model CNNs". In: (Nov. 25, 2016). arXiv: 1611.08402 [cs].

[MP87] Marvin Minsky and Seymour Papert. "Perceptrons: An introduction to computational geometry (expanded edition)". In: *Editorial The MIT Pres. Libro publicado* 28 (1987).

[Mud+17] Mayur Mudigonda et al. "Segmenting and Tracking Extreme Climate Events Using Neural Networks". In: (2017), p. 5.

[MVT16] Diego Marcos, Michele Volpi, and Devis Tuia. "Learning Rotation Invariant Convolutional Filters for Texture Classification". In: (2016), p. 6.

[Nac65] L Nachbin. *The Haar Integral*. 1965.

[Nak03] M Nakahara. *Geometry, Topology, and Physics*. 2003.

[Ngu+19] Thu Nguyen-Phuoc et al. "HoloGAN: Unsupervised Learning of 3D Representations from Natural Images". In: (Oct. 1, 2019). arXiv: 1904.01326 [cs].

[NR84] Lora Nikolova and V. A. Rizov. "Geometrical Approach to the Reduction of Gauge Theories with Spontaneously Broken Symmetry". In: *Reports on Mathematical Physics* 20.3 (Dec. 1, 1984), pp. 287–301.

[Ola14] Chris Olah. *Groups and Group Convolutions*. 2014. URL: https://colah.github.io/posts/2014-12-Groups-Convolution/.

[Ola15] Chris Olah. *Neural Networks, Types, and Functional Programming*. 2015. URL: https://colah.github.io/posts/2015-09-NN-Types-FP/.

[OM15] E Oyallon and S Mallat. "Deep Roto-Translation Scattering for Object Classification". In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015, pp. 2865–2873.

[Per+18] Nathanaël Perraudin et al. "DeepSphere: Efficient Spherical Convolutional Neural Network with HEALPix Sampling for Cosmological Applications". In: (Oct. 29, 2018). arXiv: 1810.12186 [astro-ph].

[Pet17] Jean Petitot. *Elements of Neurogeometry: Functional Architectures of Vision*. Lecture Notes in Morphogenesis. Springer International Publishing, 2017.

[PO18] Adrien Poulenard and Maks Ovsjanikov. "Multi-Directional Geodesic Neural Networks via Equivariant Convolution". In: (Oct. 1, 2018). arXiv: 1810.02303 [cs].

[PPV09] Daniel Potts, J Prestin, and A Vollrath. "A Fast Algorithm for Nonequispaced Fourier Transforms on the Rotation Group". In: *Numer. Algorithms* (2009), pp. 1–28.

[PST98] Daniel Potts, Gabriele Steidl, and Manfred Tasche. "Fast and Stable Algorithms for Discrete Spherical Fourier Transforms". In: *Linear Algebra Appl.* 275 (1998), pp. 433–450.

[Raj+16] Anant Raj et al. "Local Group Invariant Representations via Orbit Embeddings". Dec. 6, 2016.

[RB07] M Reisert and H Burkhardt. "Learning Equivariant Functions with Matrix Valued Kernels". In: *J. Mach. Learn. Res.* 8 (2007), pp. 385–408.

[Ree14] Mark Reeder. "Notes on Representations of Finite Groups". In: (2014).

[Rei08] Marco Reisert. "Group Integration Techniques in Pattern Analysis: A Kernel View". Albert-Ludwigs-University, 2008.

[Rez+16] Danilo Jimenez Rezende et al. "Unsupervised Learning of 3D Structure from Images". In: (2016).

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (2015), pp. 234–241.

[RMC15] Alec Radford, Luke Metz, and Soumith Chintala. "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks". In: (Nov. 19, 2015).

[Roc04] Daniel N Rockmore. "Recent Progress and Applications in Group FFTS". In: *NATO Science Series II: Mathematics, Physics and Chemistry* 136 (2004), pp. 227–254.

[RS12] Gerd Rudolph and Matthias Schmidt. *Differential Geometry and Mathematical Physics: Part I. Manifolds, Lie Groups and Hamiltonian Systems*. 2013 edition. Dordrecht ; New York: Springer, Nov. 10, 2012. 762 pp.

[RS17] Gerd Rudolph and Matthias Schmidt. *Differential Geometry and Mathematical Physics: Part II. Fibre Bundles, Topology and Gauge Fields*. 1st ed. 2017 edition. New York, NY: Springer, Mar. 29, 2017. 830 pp.

[RSP17a] Siamak Ravanbakhsh, Jeff Schneider, and Barnabas Poczos. "Deep Learning with Sets and Point Clouds". In: International Conference on Learning Representations (ICLR) – Workshop Track. 2017.

[RSP17b] Siamak Ravanbakhsh, Jeff Schneider, and Barnabas Poczos. "Equivariance Through Parameter-Sharing". Feb. 27, 2017.

[Rup+12] Matthias Rupp et al. "Fast and Accurate Modeling of Molecular Atomization Energies with Machine Learning". In: *Physical Review Letters* 108.5 (Jan. 31, 2012), p. 058301.

[RVB15] Antti Rasmus, Harri Valpola, and Mathias Berglund. "Semi-Supervised Learning with Ladder Network". In: *arXiv* (2015), pp. 1–17.

[Sav+16] M. Savva et al. "Large-Scale 3D Shape Retrieval from ShapeNet Core55". In: *Proceedings of the Eurographics 2016 Workshop on 3D Object Retrieval*. 3DOR '16. Goslar Germany, Germany: Eurographics Association, 2016, pp. 89–98.

[Sch16] Frederic Schuller. *Lectures on the Geometrical Anatomy of Theoretical Physics*. 2016. URL: https://www.youtube.com/playlist?list=PLPH7f_7ZlzxTi6kS4vCmv4ZKm9u8g5yic.

[SDL18] Stefan C. Schonsheck, Bin Dong, and Rongjie Lai. "Parallel Transport Convolution: A New Tool for Convolutional Neural Networks on Manifolds". In: (2018). arXiv: 1805.07857 [cs, math, stat].

[Sej86] Terrence J. Sejnowski. "Higher-Order Boltzmann Machines". In: *AIP Conference Proceedings*. AIP Conference Proceedings Volume 151. Vol. 151. AIP, 1986, pp. 398–403.

[Ser77] Jean-Pierre Serre. *Linear Representations of Finite Groups*. Springer, 1977.

[SF95] E P Simoncelli and W T Freeman. "The Steerable Pyramid: A Flexible Architecture for Multi-Scale Derivative Computation". In: *Proc. Int. Conf. Image Anal. Process.* 3 (1995), pp. 444–447.

[SG17a] Yu-Chuan Su and Kristen Grauman. "Flat2Sphere: Learning Spherical Convolution for Fast Features from 360{\deg} Imagery". In: (2017).

[SG17b] Y C Su and K Grauman. "Learning Spherical Convolution for Fast Features from 360 Imagery". In: *Adv. Neural Inf. Process. Syst.* (2017).

[SGS15a] R K Srivastava, K Greff, and J Schmidhuber. "Highway Networks". In: 2015.

[SGS15b] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. "Training Very Deep Networks". In: *Adv. Neural Inf. Process. Syst.* (2015).

[Sha+16] Wenling Shang et al. "Understanding and Improving Convolutional Neural Networks via Concatenated Rectified Linear Units". In: International Conference on Machine Learning (ICML). Vol. 48. 2016.

[Sha89] J Shawe-Taylor. "Building symmetries into feedforward networks". In: *1989 First IEE International Conference on Artificial Neural Networks, (Conf. Publ. No. 313)*. Oct. 1989, pp. 158–162.

[Sha93] J Shawe-Taylor. "Symmetries and Discriminability in Feedforward Network Architectures". In: *IEEE Trans. Neural Netw.* (1993), pp. 1–25.

[Sha97] R W Sharpe. *Differential Geometry: Cartan's Generalization of Klein's Erlangen Program*. 1997.

[SKH86] Terrence J Sejnowski, Paul K Kienker, and Geoffrey E Hinton. "Learning symmetry groups with hidden units: Beyond the perceptron". In: *Physica D* 22.1 (Oct. 1986), pp. 260–275.

[Ski13] H Skibbe. "Spherical Tensor Algebra for Biomedical Image Analysis". Albert-Ludwigs-Universitat Freiburg im Breisgau, 2013.

[SL12] K Sohn and H Lee. "Learning Invariant Representations with Local Transformations". In: *Proceedings of the 29th International Conference on Machine Learning (ICML-12)* (2012).

[SM13] Laurent Sifre and Stephane Mallat. "Rotation, Scaling and Deformation Invariant Scattering for Texture Discrimination". In: *IEEE conference on Computer Vision and Pattern Recognition (CVPR)* (2013).

[Spr+15] J T Springenberg et al. "Striving for Simplicity: The All Convolutional Net". In: *Proceedings of the International Conference on Learning Representations (ICLR)* (2015).

[SR12] U Schmidt and S Roth. "Learning Rotation-Aware Features: From Invariant Priors to Equivariant Descriptors". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)* (2012).

[STD14] Samuele Salti, Federico Tombari, and Luigi Di Stefano. "SHOT: Unique Signatures of Histograms for Surface and Texture Description". In: *Computer Vision and Image Understanding* 125 (Aug. 1, 2014), pp. 251–264.

[Ste51] Norman Steenrod. *The Topology of Fibre Bundles*. 1951.

[Sug90] Mitsuo Sugiura. *Unitary Representations and Harmonic Analysis*. 2nd ed. New York, London, Sydney, Toronto: John Wiley & Sons, 1990.

[Tay86] Michael E Taylor. *Noncommutative Harmonic Analysis*. American Mathematical Society, 1986.

[Teo98] Patrick Cheng-San Teo. "Theory and Applications of Steerable Functions". Stanford University, 1998.

[Tho+18] Nathaniel Thomas et al. "Tensor Field Networks: Rotation- and Translation-Equivariant Neural Networks for 3D Point Clouds". Feb. 22, 2018.

[Tie14] Tijmen Tieleman. "Optimizing Neural Networks That Generate Images". In: *PhD thesis* (2014).

[Tok+15] Seiya Tokui et al. "Chainer: A Next-Generation Open Source Framework for Deep Learning". In: *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems (NIPS)* (2015), pp. 1–6.

[Tu10] Loring W. Tu. *An Introduction to Manifolds*. 2nd ed. 2011 edition. New York: Springer, Oct. 6, 2010. 428 pp.

[Tu17] Loring W. Tu. *Differential Geometry: Connections, Curvature, and Characteristic Classes*. New York, NY: Springer Science+Business Media, 2017.

[Vas+15] N Vasilache et al. "Fast Convolutional Nets with Fbfft: A GPU Performance Evaluation". In: International Conference on Learning Representations (ICLR). 2015.

[Vas+17] A Vaswani et al. "Attention Is All You Need". In: *arXiv preprint arXiv* (2017).

[Vee+18] Bastiaan S. Veeling et al. "Rotation Equivariant CNNs for Digital Pathology". In: June 8, 2018. arXiv: 1806.03962 [cs, stat].

[Wan+13] L Wan et al. "Regularization of Neural Networks Using Dropconnect". In: *International Conference on Machine Learning (ICML)* (2013), pp. 109–111.

[WB18] Daniel Worrall and Gabriel Brostow. "CubeNet: Equivariance to 3D Rotation and Translation". Apr. 12, 2018.

[WC18a] Marysia Winkels and Taco Cohen. "3D G-CNNs for Pulmonary Nodule Detection". In: International Conference on Medical Imaging with Deep Learning (MIDL). 2018.

[WC18b] Marysia Winkels and Taco Cohen. "3D Group-Equivariant Neural Networks for Octahedral and Square Prism Symmetry Groups". In: *FAIM/ICML Workshop: Towards Learning with Limited Labels: Equivariance, Invariance, and Beyond*. 2018.

[WC19a] Maurice Weiler and Gabriele Cesa. "General E(2) - Equivariant Steerable CNNs". In: Neural Information Processing Systems (NeurIPS). 2019, p. 12.

[WC19b] Marysia Winkels and Taco Cohen. "Pulmonary Nodule Detection in CT Scans with Equivariant CNNs". In: *Medical Image Analysis (Under Review)* (2019).

[Wea14] James Owen Weatherall. "Fiber Bundles, Yang-Mills Theory, and General Relativity". In: (Nov. 12, 2014). arXiv: 1411.3281 [gr-qc, physics:hep-th, physics:math-ph, physics:physics].

[Wei+18] Maurice Weiler et al. "3D Steerable CNNs: Learning Rotationally Equivariant Features in Volumetric Data". In: July 6, 2018.

[Wen08] Chris Wendl. *Lecture Notes on Bundles and Connections*. 2008.

[Wey39] Hermann Weyl. *The Classical Groups: Their Invariants and Representations*. Princeton University Press, 1939.

[WHS18] Maurice Weiler, Fred A Hamprecht, and Martin Storath. "Learning Steerable Filters for Rotation Equivariant CNNs". In: Computer Vision and Pattern Recognition (CVPR). 2018.

[Win+18] Jim Winkens et al. "Improved Semantic Segmentation for Histopathology Using Rotation Equivariant Convolutional Networks". In: 2018, p. 3.

[Wor+17] Daniel E Worrall et al. "Harmonic Networks: Deep Translation and Rotation Equivariance". In: CVPR. 2017.

[WS96] Jeffrey Wood and John Shawe-Taylor. "Representation Theory and Invariant Neural Networks". In: *Discrete Appl. Math.* 69.1 (Aug. 13, 1996), pp. 33–60.

[Zah+17] Manzil Zaheer et al. "Deep Sets". In: Mar. 10, 2017.

[ZCW16] Luisa M. Zintgraf, Taco S. Cohen, and Max Welling. "A New Method to Visualize Deep Neural Networks". In: (Mar. 8, 2016). arXiv: 1603.02518 [cs].

[Zha+15] C Zhang et al. "Discriminative Template Learning in Group-Convolutional Networks for Invariant Speech Representations". In: *InterSpeech* (2015), pp. 3229–3233.

[Zin+17] Luisa M. Zintgraf et al. "Visualizing Deep Neural Network Decisions: Prediction Difference Analysis". In: (Feb. 15, 2017). arXiv: 1702.04595 [cs].

[ZK16] S Zagoruyko and N Komodakis. "Wide Residual Networks". In: *arXiv:1605.07146* (2016).