Toward Geometrical Robustness with Hybrid Deep Learning and Differential Invariants Theory

Pierre-Yves Lagrave, Mathieu Riou
Thales Research and Technology France,
1 avenue Augustin Fresnel
91767 Palaiseau cedex
pierre-yves.lagrave@thalesgroup.com, mathieu.riou@thalesgroup.com

Abstract

Symmetries are ubiquitous in physics problems and these should be taken into account when neural networks are used to approximate their solutions. Embedding symmetries within the neural networks by using equivariant layers has been shown to be efficient from an accuracy standpoint. Building equivariant structures also appears appealing from the robustness standpoint since the use of correct-by-design algorithms alleviates the verification step, which is a prerequisite to any critical applications such as safety and military related tasks. However, generically enforcing equivariance in neural networks requires the use of cumbersome operators such has group-based convolution kernels, for which the outputs may be hard to interpret. In this paper, we introduce `EqPdeNet`, an alternative method in which equivariant partial differential equations are embedded within the first layer of a neural network. This approach provides approximate equivariance with respect to any Lie group action and allows combining several types of equivariance within the same network. Moreover, the structure of the associated partial differential equations can be directly related to the physical nature of the input data, making this approach particularly appealing from an interpretability standpoint when compared to the use of group-based convolution kernels.

Introduction

Symmetries are ubiquitous in physics with finite groups of symmetries such as the hexagonal lattice of the graphene and continuous groups such as Lorentz group in particle physics.

Highlighted by their successes in image and speech recognition (Szegedy et al. 2017), (Xiong et al. 2016), neural networks are now used in various physics fields such as fluid mechanics (Raissi, Perdikaris, and Karniadakis 2019), (Raissi, Yazdani, and Karniadakis 2020), high energy physics (Baldi, Sadowski, and Whiteson 2014) or condensed matter physics (Carrasquilla and Melko 2017), (Van Nieuwenburg, Liu, and Huber 2017).

Convolutional Neural Networks (CNN) (LeCun et al. 1998) have demonstrated the efficiency of embedding translation symmetries into the design of a neural network for image processing taks. More generally, directly encoding the required symmetries into the algorithm design decreases the number of parameters and increases the robustness. Using algorithms in which given properties are enforced through their specification (correct-by-design algorithms) has also the advantage of being more amenable to critical applications such as safety and military related tasks. In regards to the diversity of symmetries occurring in physics, a generic approach for embedding the symmetry groups into the neural networks design is needed. Group-Convolutional Neural Networks (G-CNN), firstly introduced by (Cohen and Welling 2016), have been recently extended to generic groups of symmetries (Finzi et al. 2020) and achieve this purpose. However, they rely on a cumbersome specification and are hard to interpret.

In this paper, we introduce the `EqPdeNet` neural network by leveraging on the differential invariants of Lie group actions. Within `EqPdeNet`, equivariant inner representations of the input data are built through the first layer of the neural network and are then processed by deeper fully connected layers, following the hierarchical structure of the usual CNN. This hybrid approach applies to any Lie group without requiring the group action to act transitively on the input manifold and increases the accuracy and the robustness by achieving approximate equivariance. In addition, the structure of the associated Partial Differential Equations (PDE) can be directly related to the physical nature of the input data, making this approach particularly appealing from an interpretability standpoint when compared to the use of group-based convolution kernels.

Related Work and Contribution

We review in the following the related work by first focusing on G-CNN and then highlight the existing duality between neural networks and PDE.

Group-Convolutional Neural Networks

The success of CNN for image processing task has motivated several works with respect to the generalization of their translation equivariant layers to other type of transforms. In this context, (Cohen and Welling 2016) introduced

the concept of Group-Convolutional Neural Network (G-CNN) by extending the principle of weights sharing to satisfy other symmetries than translations and focuses on discrete groups such as p4 and p4m. Other works were developed focusing on specific groups of symmetry such as the permutations group (Zaheer et al. 2017), some discrete subgroups of the 2-dimensional rotation group $SO(2)$ (Marcos, Volpi, and Tuia 2016), $SO(2)$ itself (Oyallon and Mallat 2015), (Worrall et al. 2017), (Weiler, Hamprecht, and Storath 2018), the 3-dimensional translation and rotation groups (Cohen et al. 2018), (Esteves et al. 2018).

These approaches were later generalized to more general sets of transforms and in particular, to those arising from a transitive action of a Lie group (Gens and Domingos 2014), (Huang et al. 2017), (Bekkers 2019). Recently, a generic approach was proposed (Finzi et al. 2020) without requiring the group action to be transitive.

All these works aim at generalizing the usual CNN structure by building equivariant layers to make the overall network equivariant. Our approach rather aims at achieving approximate equivariance through the use of one PDE layer and does not use group-based convolution kernels.

PDE-Based Neural Networks

Motivated by the universal approximation theorem (Hornik et al. 1989), neural networks have been used to approximate the solutions of PDE. A major work in this area is the introduction of the Physics Informed Neural Network (PINN) approach (Raissi, Perdikaris, and Karniadakis 2019) as an alternative to the usual finite difference methods.

(Chen et al. 2018) emphasizes that a residual neural network can actually be seen as some discretization of an unknown Ordinary Differential Equation (ODE) and they show how to efficiently learn the ODE parameters from the data by using adjoint techniques and classical ODE solvers. Building on similar ideas, (Ruthotto and Haber 2019) uses the ODE formulation of the neural network to introduce inductive bias, such as parabolic or hyperbolic properties to enforce respectively robustness to perturbations and low memory usage. The use of differential equation formulation to embed desired properties into the neural network is a common feature with our work. However the question of symmetry is not considered in this work.

The use of PDE has also appeared useful for building equivariant structures. (Shen et al. 2020) introduces an equivariant kernel to the isometry group SE(2) from differential operators approximated through usual kernel convolutions. In (Smets et al. 2020), a neural network equivariant to a generic transitive Lie group action is proposed by using several layers of equivariant PDE, the training of the algorithm consisting in finding the parameters of the PDE. Less recently, but closer to the present work, (Fang et al. 2017) have used a single PDE to extract equivariant features for a linear classifier by leveraging on differential invariants theory.

The hybrid approach we are proposing is applicable to any Lie group action provided that a generating set of differential invariants can be efficiently computed and it allows for several group actions to be considered simultaneously. Also, thanks to a convolution-based integration of the PDE layer, an end-to-end training can be performed within some automatic differentiation framework such as TensorFlow or PyTorch.

Contribution

The main contributions of this paper are the following:

- We introduce the `EqPdeNet` hybrid architecture featuring a first equivariant PDE layer by leveraging on the differential invariants of Lie group actions, followed by usual fully connected layers. Our approach in particular allows considering several types of equivariance within the first layer PDE layer.
- We give some numerical evidence to support the interest of our approach from both performance and robustness standpoints by performing some comparisons with the behavior of some usual neural networks on the ROTMNIST dataset (Larochelle et al. 2007).
- We provide a numerical integration scheme for arbitrary PDE by using some discrete convolution operators, making the `EqPdeNet` approach compatible with an end-to-end training through back-propagation within an automatic differentiation framework.

Invariance and PDE

By leveraging on the formalism introduced in (Olver 1993), we give in the following some general background about invariance theory for PDE. This will allow us to introduce the notion of differential invariants of a Lie group action, which is central to our work, and to explain how to build equivariant representations of input data by solving a specific type of PDE.

Symmetry Group

Formally, we will see a PDE of order n in p independent variables $x = (x_1, ..., x_p) \in \mathcal{X}$ and one dependent variable $u = u(x_1, ..., x_p) \in \mathcal{U}$ as an equation involving x, u and $u_\alpha = \partial_\alpha u$, for $\alpha \in \mathbb{N}^k$, $k \geq 0$ and $|\alpha| \leq n$. A PDE solution will be of the form $u = f(x)$.

In the following, we denote by $\mathcal{X} = \mathbb{R}^p$, with coordinates $(x_1, ..., x_p)$, the space of the independent variables and by $\mathcal{U} = \mathbb{R}$, with coordinates u, that of the dependent variable. Let's then consider a Lie group G of dimension m acting as $g.(x, u)$ on a sub-manifold $\mathcal{M} \subseteq \mathcal{X} \times \mathcal{U}$, with its Lie algebra $\mathfrak{g}$ generated by the vector fields $\zeta_1, ..., \zeta_m$. For instance, G could be the 2-dimensional rotation group $SO(2)$ acting on $\mathcal{X} \times \mathcal{U} \simeq \mathbb{R}^2$ with the infinitesimal generator $\zeta_1 = -u\partial_x + x\partial_u$.

We can define the transform of a function $u = f(x)$ under the action of G by identifying f with its graph $\Gamma_f = \{(x, f(x)),\ x \in \Omega \subseteq \mathcal{X}\} \subseteq \mathcal{X} \times \mathcal{U}$ and by defining $g.f = f_g$, where the function f_g is the function associated with the transformed graph $g.\Gamma_f$ defined as it follows for $g \in G$:

$$g.\Gamma_f = \{g.(x, f(x)), (x, f(x)) \in \Gamma_f\} = \Gamma_{f_g} \quad (1)$$

These notions of transformed function and group action on functional graphs are illustrated on Figure 1 where the graph

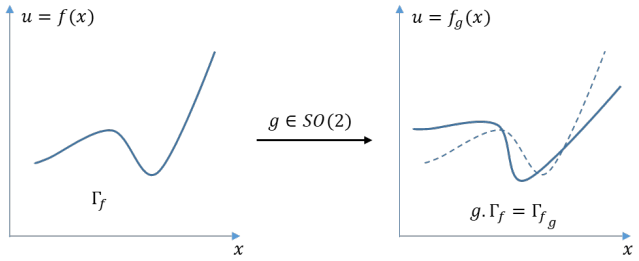

Figure 1: Action of an element g in the 2-dimensional rotation group $SO(2)$ on the graph Γ_f of a function $f : \mathbb{R} \to \mathbb{R}$

of a function $f : \mathbb{R} \to \mathbb{R}$ (left) is transformed according to the action of a group element $g \in SO(2)$ by simple rotation (right).

With this formalism, a symmetry group G of the considered PDE is a group G acting of $\mathcal{M} \subseteq \mathcal{X} \times \mathcal{U}$ in such a way that if f is a solution, then its transformed f_g by the group action is also a solution.

Differential Invariants We call $n-$order jet space $J^{(n)}$ the cartesian product between the space of the independent variables $\mathcal{X}$ and enough copies of the space of the dependent variable $\mathcal{U}$ to include coordinates for each partial derivative of order less or equal than n:

$$J^{(n)} = \mathcal{X} \times \underbrace{\mathcal{U} \times .... \times \mathcal{U}}_{\binom{p+n}{n}} \tag{2}$$

In the above definition, the binomial coefficient $\binom{p+n}{n}$ corresponds to the number of partial derivatives of the function f (assumed to be smooth enough) with order less or equal than n. A function $f : \mathcal{X} \to \mathcal{U}$ represented as $u = f(x)$ can naturally be prolonged to a function $u^{(n)} = f^{(n)}(x)$ from $\mathcal{X}$ to $J^{(n)}$ by evaluating f and the corresponding partial derivative, so that $u^{(n)} = \{u_\alpha, |\alpha| \leq n\}$.

According to the considered formalism, a generic PDE could therefore be written as it follows,

$$\Delta\left(x, u^{(n)}\right) = 0 \tag{3}$$

where Δ is an operator from the $n-$order jet space $J^{(n)}$ to $\mathbb{R}$. We then denote by $\mathrm{pr}^{(n)} G$ the prolongation of the group action of G to $J^{(n)}$ for which a prolonged transform $g^{(n)}$, for $g \in G$, sends the graph $\Gamma_{f^{(n)}}$ onto $\Gamma_{(g.f)^{(n)}}$, and by $\mathrm{pr}^{(n)} \zeta_1, ..., \mathrm{pr}^{(n)} \zeta_m$ the corresponding prolonged vector fields.

In the following, we will be interested in operators Δ associated with the PDE having G as a symmetry group. These operators are called the differential invariants of the action of G and are the algebraic invariants of the prolonged group action $\mathrm{pr}^{(n)} G$, for $n \geq 0$. They can be obtained by leveraging on the infinitesimal invariance criteria $\mathrm{pr}^{(n)} \zeta_i \Delta = 0$ for $i = 1, ..., m$ (Olver 2016) and (Hubert 2009).

A set of differential invariants of order n will be generically denoted by $\partial\phi^G_{u,n}$ in the sequel.

Examples We have chosen to work with image classification to illustrate our approach and we have considered the action of the 2 dimensional special euclidean group $SE(2)$ and that of the group $\Lambda_{\mathbb{R}^*_+}(2)$ of the scaled translations on $\mathcal{X} \subseteq \mathbb{R}^2$ ($p = 2$), which can be seen as actions on $\mathcal{X} \times \mathcal{U}$ by considering a trivial component for the $\mathcal{U}$ part.

Using the infinitesimal invariance criteria allows writing corresponding sets of differential invariants as it follows:

$$\partial\phi^{SE(2)}_{u,2} = \left\{ \begin{array}{c} u, \\ u_x^2 + u_y^2, \\ u_{xx} + u_{yy}, \\ u_x^2 u_{xx} + 2u_x u_y u_{xy} + u_y^2 u_{yy}, \\ u_{xx}^2 + 2u_{xy}^2 + u_{yy}^2 \end{array} \right\} \tag{4}$$

$$\partial\phi^{\Lambda_{\mathbb{R}^*_+}}_{u,2} = \left\{ \frac{u_x^2}{u_{xx}}, \frac{u_x^2}{u_{xy}}, \frac{u_x^2}{u_{yy}}, \frac{u_y^2}{u_{xx}}, \frac{u_y^2}{u_{xy}}, \frac{u_y^2}{u_{yy}} \right\} \tag{5}$$

Equivariant Representations

A map $\psi : \mathcal{A} \to \mathcal{B}$ is said to be equivariant with respect to the action of a group G if $\psi(g.a) = g.\psi(a)$, $\forall a \in \mathcal{A}$ and $\forall g \in G$. Leveraging on the differential invariants theory previously introduced, we build from $d \in \mathcal{I}$ representations which are equivariant to the action of a given group G, where $\mathcal{I}$ refers to the input space.

To do so, we consider that a data point $d \in \mathcal{I}$ can be represented by the graph of a function f_d from $\mathcal{X}$ to $\mathcal{U}$, so that $d = \{(x, f_d(x)),\ x \in \mathcal{X}\}$. With this formalism, a gray scale image such as one of the ROTMNIST samples considered in our numerical experiments can be represented by the graph of the function associating each position to its pixel value.

Following similar ideas to (Fang et al. 2017) and (Smets et al. 2020), we model the representation learning process by the following PDE:

$$\begin{cases} \partial_t u &= F\left(\partial\phi^G_{u,n}\right) \\ u_{t=0} &= f_d \end{cases} \tag{6}$$

F is a function from the set of the differential invariants to $\mathbb{R}$ and $F\left(\partial\phi^G_{u,n}\right)$ is therefore also a differential invariant, any function of the differential invariants being a differential invariant itself.

It therefore means that for $g \in G$, $g.u_T$ will also be a solution so that the learned representation of the data is actually equivariant with respect to the action of G in the sense that $g.u_T(f_d) = u_T(g.f_d)$, where $u_T(f_0)$ corresponds to the solution of (6) with initial condition f_0. Hence, as illustrated in Figure 2 in the case of $SE(2)$, diffusing the PDE (6) allows for extracting similar representations (features maps on the right) from the inputs f_d (upper left) and $g.f_d$ (lower left).

Different functions F of the differential invariants lead to different equivariant representations by diffusing the corresponding PDE. As equivariance only is not enough for a representation to be discriminative (e.g., black areas in the corner of the MNIST samples), we will then use a learning approach to identify the representations conveying some

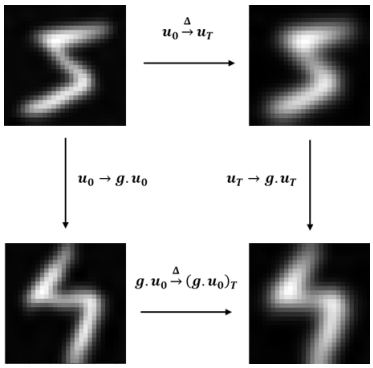

Figure 2: Extraction of an $SE(2)$- equivariant representation of an MNIST sample I_0 with the heat equation $\partial_t u = u_{xx} + u_{yy}$, $u_{t=0} = I_0$. The equivariant property makes the associated diagram commutative, for $g \in SE(2)$.

meaningful information about the input data through the inference of the function F.

More precisely, we will assume that F belongs to a parametric space, so that $F = F_\theta$, for $\theta \in \Theta \subseteq \mathbb{R}^k$. In the following, F will be chosen to be linear as in (Fang et al. 2017) or more generally, as a multivariate polynomial in the differential invariants. The corresponding vectorial parameter θ will be part of the trainables parameters of our approach. In the following, we will denote u_T^θ the representation extracted by solving the PDE (6) with $F = F_\theta$. The explicit reference to the initial condition is made by writing $u_T^\theta(f_d)$ when needed but it will be generally dropped to ease the exposition.

An Hybrid Approach

We introduce in this section a generic hybrid approach combining the previously introduced PDE based equivariant representations learning with some fully connected feed forward layers, following the intuition behind of the hierarchical structure of usual CNN.

`EqPdeNet` Structure

We introduce the `EqPdeNet` structure depicted with 2-dimensional data on Figure 3, in which n_e PDE are used to extract the equivariant representations $u_T^{\theta_1}$,...,$u_T^{\theta_{n_e}}$. A dimension reduction layer (e.g., pooling, linear combination, etc.) is then combined with deeper fully connected layers to produce the outputs. An output $\tilde{y}_d \in \mathcal{Y}$ of `EqPdeNet` corresponding to the input f_d is then computed according to the following formula:

$$\tilde{y}_d = \mathcal{N}_\omega \left(\phi_\delta \left(u_T^{\theta_1}(f_d), .., u_T^{\theta_{n_e}}(f_d) \right) \right) \tag{7}$$

where $\mathcal{N}_\omega$ refers to the prediction function of the fully connected layers with weights ω and δ to the parameters of the dimension reduction layer. Denoting by $L : (\mathcal{Y} \times \mathcal{Y})^{n_t} \to \mathbb{R}$, for $n_t \geq 1$, a relevant loss function for the considered learning task, the training of the algorithm therefore consists in finding an approximate solution to the following minimization problem,

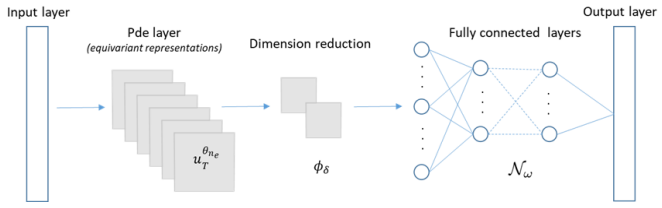

Figure 3: `EqPdeNet` structure combining a first equivariant PDE layer, a dimension reduction layer and deeper fully connected layers

$$\min_{\theta_1,\ldots,\theta_{n_e},\omega,\delta} L\left(\prod_{i=1}^{n_t} \{\tilde{y}_{d_i}, y_{d_i}\} \right) \tag{8}$$

where the $(f_{d_i}, y_{d_i})_{i=1}^{n_t}$ refers to the training samples.

PDE Integration

In order to efficiently find some numerical approximations to the equivariant PDE and to train the entire architecture from end-to-end, we propose an integration method compatible with the backpropagation technique within some automatic differentiation frameworks such as TensorFlow or PyTorch.

Convolution Approach Following (Ruthotto and Haber 2019) and (Long et al. 2018), our approach consists in approximating the PDE integration operator with some usual convolution layers built from well chosen kernels. More precisely, we consider the explicit Euler discretization of (6), which we write as it follows:

$$\begin{cases} u_{\ell+1} & = u_\ell + \Delta t \times F_\theta\left(\partial \phi_{u,n}^G\right) \\ u_0 & = f_d \end{cases} \tag{9}$$

where $u_\ell = u_{\ell \Delta t}$, $\Delta_t > 0$ is a discretization parameter and $0 \leq \ell \leq \ell_T$, with $\ell_T = \left\lfloor \frac{T}{\Delta t} \right\rceil$. We then consider that each iteration of the above Euler scheme corresponds to a layer of a neural network with input u_ℓ and outputs $u_\ell + \Delta t \times F_\theta\left(\partial \phi_u^G\right)$. Our approach called `EulerConv` (Figure 4) then consists in implementing this layer by approximating the differential operator ∂_α required for building the differential invariants $\partial \phi_u^G$ with some appropriate convolution filters.

For each differentiation index α, it is therefore possible to write

$$\partial_\alpha U_\ell = K_\alpha \star U_\ell \tag{10}$$

where U_ℓ is a tensor referring to a discretization of u_ℓ over the domain $\mathcal{X}$, K_α is a constant convolution kernel and $\star$ is the discrete convolution operator (Differential convolution layer on Figure 4). The differential invariants can then be obtained from the values $\partial_\alpha U_\ell$ which correspond to the approximate values of the differential $\partial_\alpha u_\ell$, computed by finite differences through the convolution kernel K_α (Differential invariants layer on Figure 4). The corresponding output (Update layer on Figure 4) is the result of one step of the Euler scheme (9). The unit allowing to perform the entire Euler

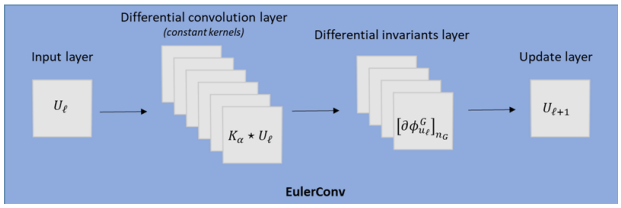

Figure 4: `EulerConv` unit allowing to perform one step in the explicit Euler scheme by approximating the differential operators ∂_α with appropriate convolution kernels

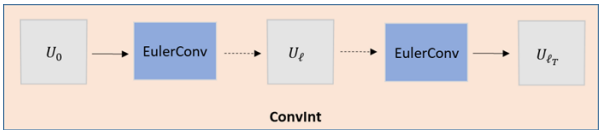

Figure 5: `ConvInt` unit allowing to integrate the PDE using an explicit discretisation scheme using several `EulerConv`units

scheme is referred to as `ConvInt` (Figure 5) and is a concatenation of `EulerConv` layers for the appropriate number of time steps.

About Numerical Accuracy The above convolution approach to the PDE integration can actually be seen as a specific explicit finite difference scheme, therefore raising some natural questions about consistency, stability and convergence. Even for simple choices of the function F_θ, the theoretical analysis of this scheme is not an easy task to perform due to the strong non-linearity introduced by the differential invariants and we therefore defer it to some further work.

It is however possible to comment on some practical tools that can be used to control the numerical accuracy of the discretization scheme. Considering the time dimension only, it holds that $\|u_{\ell\Delta_t} - u_t\| \overset{\Delta t\to 0}{=} o(\Delta t)$ so that we can make the time discretization error arbitrarily small by decreasing the parameter Δt and adding some more `EuleurConv` units accordingly in the `ConvInt` units. With respect to the space dimension, the discretization error can be controlled by increasing the number of sampling points when building U_ℓ from $u_{\ell\Delta t}$.

In some practical situations such as image processing tasks, the input data lies in a discrete manifold so that the smooth functional representation that was previously introduced does not directly apply. In this case, interpolation methods such as the functional convolution can be used to obtain some continuous inputs (Simard et al. 1998).

Numerical Experiments

We provide in this section the results of numerical experiments we have conducted on the 2-dimensional problem of image classification. However, as generically applicable to symmetric learning tasks on smooth functional data, our approach is not specific to image classification and could for instance be instantiated to predict the evolution of a physical system with symmetries (Noether's theorem) as long as a generating set of differential invariants can be efficiently computed for the corresponding symmetry group.

`EqPdeNet`			
#param	Test	`iso`	`sca`
13033	$70.7_{(2.7)}$	$37.3_{(1.6)}$	$23.1_{(1.3)}$
26537	$77.9_{(0.6)}$	$39.6_{(1.3)}$	$24.2_{(1.1)}$
55081	$82.7_{(0.7)}$	$43.8_{(0.9)}$	$25.8_{(1.4)}$
118313	$86.8_{(0.6)}$	$48.8_{(1.1)}$	$28.2_{(1.0)}$
269353	$89.9_{(0.8)}$	$53.0_{(0.9)}$	$30.2_{(1.1)}$

Table 1: Accuracy of the `EqPdeNet` network in several scenarios after training on the original ROTMNIST training samples

Following the line of existing work with respect to the testing of equivariant algorithms, we have structured our numerical from the ROTMNIST dataset and we emphasize here that we did not use any kind of data augmentation technique for the training step. More precisely, the ROTMNIST dataset was built in (Larochelle et al. 2007) from the orignial MNIST digits by applying to the original samples random rotations with angles sampled uniformly in $[0, 2\pi]$. In the following, algorithms have been trained on the 12k training samples and all results have been obtained with a TensorFlow based implementation of our approach running on a GeForce RTX 2080 Nvidia card.

We have used a `EqPdeNet` network with a PDE layer aiming to build equivariant data representations with respect to the translation group, and either the rotation or the scaling group. More precisely, the PDE layer includes two PDE built from the differential invariants $\partial\phi^{SE(2)}_{u,2}$ and two others combining those of $\partial\phi^{\Lambda_{\mathbb{R}^*_+}}_{u,2}$, whose outputs are then linearly combined.

To illustrate the benefits of our approach when compared to corresponding fully connected neural networks (FCNN) from both accuracy and robustness standpoints, we have built several scenarios from the original ROTMNIST testing set, namely

- `iso`: a random isometry, i.e. a combination of a random translation of (t_h, t_v) pixels and a random rotation of θ degrees, where $t_h \sim \mathcal{U}(-2, 2)$, $t_v \sim \mathcal{U}(-2, 2)$, and $\theta \sim \mathcal{U}(-30, 30)$, is applied to each of the original testing samples.
- `sca`: a random scaling transform $x, y \to (\lambda x, \lambda y)$ with parameter $\lambda \sim \mathcal{U}\left(\frac{2}{3}, 1\right)$is applied to each of the original testing samples.

where $\mathcal{U}(a, b)$ refers to the uniform distribution on the interval$[a, b]$.

The accuracy results in each scenario obtained after averaging over 10 instances of testing to smooth out the statistical noise are given in tables 1 and 2, together with the corresponding standard deviation as subscripts.

We see that the accuracy on the testing set is consistently higher with our approach than with the corresponding FCNN, for all the considered numbers of parameters. With

FCNN			
#param	Test	`iso`	`sca`
13002	$65.5_{(1.5)}$	$36.7_{(1.4)}$	$23.5_{(0.6)}$
26506	$73.4_{(0.9)}$	$37.9_{(1.2)}$	$23.4_{(0.7)}$
55050	$80.8_{(0.5)}$	$42.4_{(1.0)}$	$25.0_{(1.2)}$
118282	$85.7_{(0.4)}$	$47.0_{(0.8)}$	$26.9_{(0.9)}$
269322	$87.7_{(0.5)}$	$49.5_{(0.6)}$	$28.3_{(0.8)}$

Table 2: Accuracy of FCNN in several scenarios after training on the original ROTMNIST training samples

respect to robustness, higher accuracies are reached with our approach in the `iso` and `sca` testing scenarios as the number of parameters increases, consistently with the increase of the overfitting risk.

Hence, although less performant than using G-CNN which are able to achieve a testing accuracy of almost 99% with $SE(2)$ equivariance (Finzi et al. 2020) because of a simpler structure and approximate equivariance, our `EqPdeNet` approach does provide material improvements with respect to the usual FCNN, from both accuracy and robustness standpoints.

Conclusions and Further Work

In this paper we proposed an hybrid architecture with a first PDE based layer made equivariant to generic group actions by leveraging on differential invariants theory. This structure allows achieving simultaneous approximate equivariance with respect to several group actions by aggregating the learned inner representations through a dimension reduction layer feeding deeper fully connected layers.

In order to make the approach practical, we have specified an end-to-end training method compatible with the usual automatic differentiation frameworks in which the numerical approximations to the several PDE solutions are obtained through the use of fixed weights convolution operators.

We have performed some numerical testing on the ROTMNIST dataset and have shown the superiority of our approach from both accuracy and robustness standpoints, when compared to fully connected neural networks. Our results are however below those reported for G-CNN as our approach is simpler and does not ensure strict equivariance. However, the PDE built from the differential invariants are easier to interpret than group-based convolution kernels.

Although we believe the approach and our preliminary numerical results to be promising, additional work is needed for deriving rigorous rules with respect to hyperparameters setting. In particular, a theoretical analysis of the convergence of the discretization units for several Lie groups and PDE types would be valuable.

Finally, by leveraging on the interpretability feature of our approach, we plan to conduct analysis of the learned equivariant representations as to refine the choice of the parametric form of the PDE to be considered, to study the opportunity to use partial specification techniques and to discuss some safety and certification aspects. Also, as done in (Finzi et al. 2020) to model convolution kernels, using a small neural network instead of a multivariate polynomial for the parameterization of the differential invariants through the function F_θ may help improving the expressiveness of our approach.

References

Baldi, P.; Sadowski, P.; and Whiteson, D. 2014. Searching for exotic particles in high-energy physics with deep learning. *Nature communications* 5(1): 1–9.

Bekkers, E. J. 2019. B-spline cnns on lie groups. *arXiv preprint arXiv:1909.12057* .

Carrasquilla, J.; and Melko, R. G. 2017. Machine learning phases of matter. *Nature Physics* 13(5): 431–434.

Chen, R. T.; Rubanova, Y.; Bettencourt, J.; and Duvenaud, D. K. 2018. Neural ordinary differential equations. In *Advances in neural information processing systems*, 6571–6583.

Cohen, T.; and Welling, M. 2016. Group equivariant convolutional networks. In *International conference on machine learning*, 2990–2999.

Cohen, T. S.; Geiger, M.; Köhler, J.; and Welling, M. 2018. Spherical cnns. *arXiv preprint arXiv:1801.10130* .

Esteves, C.; Allen-Blanchette, C.; Makadia, A.; and Daniilidis, K. 2018. Learning so (3) equivariant representations with spherical cnns. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 52–68.

Fang, C.; Zhao, Z.; Zhou, P.; and Lin, Z. 2017. Feature learning via partial differential equation with applications to face recognition. *Pattern Recognition* 69: 14–25.

Finzi, M.; Stanton, S.; Izmailov, P.; and Wilson, A. G. 2020. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. *arXiv preprint arXiv:2002.12880* .

Gens, R.; and Domingos, P. M. 2014. Deep symmetry networks. In *Advances in neural information processing systems*, 2537–2545.

Hornik, K.; Stinchcombe, M.; White, H.; et al. 1989. Multilayer feedforward networks are universal approximators. *Neural networks* 2(5): 359–366.

Huang, Z.; Wan, C.; Probst, T.; and Van Gool, L. 2017. Deep learning on lie groups for skeleton-based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 6099–6108.

Hubert, E. 2009. Differential invariants of a Lie group action: Syzygies on a generating set. *Journal of Symbolic Computation* 44(4): 382 – 416. ISSN 0747-7171. doi: https://doi.org/10.1016/j.jsc.2008.08.003. URL http://www.sciencedirect.com/science/article/pii/S0747717108001089.

Larochelle, H.; Erhan, D.; Courville, A.; Bergstra, J.; and Bengio, Y. 2007. An Empirical Evaluation of Deep Architectures on Problems with Many Factors of Variation. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, 473–480. New York, NY, USA: Association for Computing Machinery. ISBN 9781595937933. doi:10.1145/1273496.1273556.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11): 2278–2324.

Long, Z.; Lu, Y.; Ma, X.; and Dong, B. 2018. PDE-Net: Learning PDEs from Data. volume 80 of *Proceedings of Machine Learning Research*, 3208–3216. Stockholmsmässan, Stockholm Sweden: PMLR. URL http://proceedings.mlr.press/v80/long18a.html.

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

Olver, P. 1993. *Applications of Lie Groups to Differential Equations*. New York, NY, USA: Springer-Verlag.

Olver, P. 2016. Equivariant Moving Frames for Euclidean Surfaces.

Oyallon, E.; and Mallat, S. 2015. Deep roto-translation scattering for object classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2865–2873.

Raissi, M.; Perdikaris, P.; and Karniadakis, G. E. 2019. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics* 378: 686–707.

Raissi, M.; Yazdani, A.; and Karniadakis, G. E. 2020. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science* 367(6481): 1026–1030.

Ruthotto, L.; and Haber, E. 2019. Deep Neural Networks Motivated by Partial Differential Equations. *Journal of Mathematical Imaging and Vision* 62: 352–364.

Shen, Z.; He, L.; Lin, Z.; and Ma, J. 2020. PDO-eConvs: Partial Differential Operator Based Equivariant Convolutions. *arXiv preprint arXiv:2007.10408* .

Simard, P.; LeCun, Y.; Denker, J. S.; and Victorri, B. 1998. Transformation Invariance in Pattern Recognition-Tangent Distance and Tangent Propagation. In *Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop*, 239–27. Berlin, Heidelberg: Springer-Verlag. ISBN 3540653112.

Smets, B.; Portegies, J.; Bekkers, E.; and Duits, R. 2020. PDE-based Group Equivariant Convolutional Neural Networks. *arXiv preprint arXiv:2001.09046* .

Szegedy, C.; Ioffe, S.; Vanhoucke, V.; and Alemi, A. A. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*.

Van Nieuwenburg, E. P.; Liu, Y.-H.; and Huber, S. D. 2017. Learning phase transitions by confusion. *Nature Physics* 13(5): 435–439.

Weiler, M.; Hamprecht, F. A.; and Storath, M. 2018. Learning steerable filters for rotation equivariant CNNs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 849–858.

Worrall, D. E.; Garbin, S. J.; Turmukhambetov, D.; and Brostow, G. J. 2017. Harmonic networks: Deep translation and rotation equivariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5028–5037.

Xiong, W.; Droppo, J.; Huang, X.; Seide, F.; Seltzer, M.; Stolcke, A.; Yu, D.; and Zweig, G. 2016. Achieving human parity in conversational speech recognition. *arXiv preprint arXiv:1610.05256* .

Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Poczos, B.; Salakhutdinov, R. R.; and Smola, A. J. 2017. Deep sets. In *Advances in neural information processing systems*, 3391–3401.