Group equivariant learning

March 21, 2023

Dell'Omo Michela

Abstract

CNNs have a significant impact on classification, segmentation, and image detection tasks. They offer good computational power due to weight sharing, and are particularly effective for images that exhibit shift symmetry. However, CNNs have limitations in recognizing other types of symmetries. Group Equivariant Convolutional Neural Networks (GCNNs), proposed in 2016, provide an optimal generalization of CNNs. These architectures are based on the concept of "group equivariance," which enables the recognition of symmetries in datasets and facilitates weight sharing. GCNNs are used in various fields, including physics, biology, and medical image recognition

1. Introduction

This project aims to generalize Convolutional Neural Networks (CNNs) to another architecture that can handle a larger group of symmetries in the input dataset. By introducing the concept of "Group," it is possible to progressively build a filter capable of working with a general group of transformations, in this case, the p4 group, a finite group that represents all rotations of $\frac{\pi}{2}$ and translations. The objective will be to create a G-equivariant layer based on the definition of equivariance:

$$(T_g x) = T'_g \phi(x) \tag{1}$$

Here, T_g and T'_g are group actions for some $g \in G$. Equation [1] means that *transforming an input x by with g and then passing it through the filter ϕ should give the same result as first mapping x through ϕ and then transforming the representation.* The Methods section [2] provides a description of the key points of the theory, including how convolution behaves between layers, how it inherits properties in the case of non-linearity, and how to rewrite its with a simple notation. Using these tools, we apply the new GCNN class to different datasets to study its performance on various dataset characteristics, including Rotated MNIST and CIFAR-10. The results are reported in the Experimental Results section [3]. Additionally, at the end of the project and in the same section as the previous results, we introduce Steerable CNN to generalize G-space from finite to infinity, and study also performance on two other datasets, CIFAR-10+ and Rotated CIFAR-10.

Email: Dell'Omo Michela <dellomo.1699220@studenti.uniroma1.it>.

Deep Learning and Applied AI 2022, Sapienza University of Rome, 2nd semester a.y. 2021/2022.

2. Methods

With respect to the classic CNN, the definition of convolution is equivariant to the shift of the feature maps, as follows $[L_t f] \star \psi = L_t[f \star \psi]$, where L_t is the operator of shift on the feature maps, f is the feature map, and ψ is a filter. It is possible to generalize this convolution to a general group G, to find a G-correlation of the first layer of GCNN such as:

$$f \star \psi(g) = \sum_{h \in G} \sum_{k} f_k(y) \psi_k(g^{-1} y) \tag{2}$$

For the first layer, on the first summation, the index is a function $y \in \mathcal{Z}^2$. In this way, the G-equivariance of this operation is defined, using the substitution $h \to ut$:

$$[L_u f] \star \psi = L_u[f \star \psi] \tag{3}$$

which means that the G-correlation preserves the transformation property from one layer to another, but with non-linearity, it is necessary to add some constraints to inherit the transformation properties from layer to layer. This can be achieved by introducing the *Composition operator* C_ν, which has the property of $C_\nu L_h f = L_h C_\nu f$ where f and h are elements of the group G. With the commutation, a common space is defined in which C_ν and L_h have a common basis. With the aim of having an efficient implementation, we introduce the *split*: it is a plane symmetry group that decomposes a transformation $g \in G$ into a translation $t \in \mathcal{Z}^2$ and a transformation s in the stabilizer of the origin. For

$p4$, we have $g = ts$ where t is a translation and s is a rotation about the origin. Using the split of G, we write the convolution [3] as:

$$f \star \psi(ts) = \sum_{h \in X} \sum_{k} f_k(h) L_t[L_s \psi_k(h)] \tag{4}$$

where $X = \mathcal{Z}^2$ for the first layer and $X = G$ in the others. In practice, to compute the $p4$ correlation $f \star \psi$, it is possible to compute $L_s\psi$, the filter transformation for all four rotations, and then perform a planar correlation on f (Cohen, 2016a). Moreover, it is also studied the application of a steerable CNN (Cohen, 2016b), which is, in simple terms, the generalization of an *infinity* group G, in which instead of storing the value of a feature map on each group element, the model stores the Fourier transform of this feature map, up to a finite number of frequencies (Cesa, a); so we introduce a *feature fields* $\mathcal{R}^c$, where c is the coefficient of the Fourier Transform.

3. Experimental results[1]

To implement a group convolution layer, we first have implemented the $P4$ group of rotation $\frac{\pi}{2}$ and translation. Then, we have defined the filter convolution, group convolution, and GCNN classes. The parameters, loss, and optimization values such as the kernel and stride have been setting according to (Cohen, 2016a). For all datasets, we have used the following classes: **Vanilla CNN**: composed of five convolutional layers with ReLU activation function; **Z2CNN**: following the reference (Cohen, 2016a), it is composed of five convolutional layers. The first two layers use only ReLU activation function, while the remaining layers use dropout, max pooling, and ReLU; **P4CNN**: composed of a convolutional layer, as reported in (Cohen, 2016a). The first layer is made up of lifting convolution, while the remaining layers use group convolution, max pooling, dropout, and ReLU; **P4CNNBatchNorm**: similar to the previous class, but with an added regularization layer, batch normalization.

Rotated MNIST dataset The Rotated MNIST dataset, introduced by (Larochelle et al., 2007), contains 62 randomly rotated handwritten digits. The dataset is split into training, validation, and test sets of size $10k$, $2k$, and $50k$, respectively, as suggested in (Cohen, 2016a). The enumerated classes used in this experiment are identical to those in the dataset; since this dataset exhibits internal symmetry, we expected GCNNs to perform better than CNNs.

CIFAR10 dataset 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 subsets. To decrease the complexity of the system, we removed one convolutional layer, resulting in a constant number of parameters across all classes. We expected CNNs and GCNNs to perform similarly because the dataset doesn't have inner symmetry.

Table 1. Performance comparison.

Classes	RotMNIST Acc[%]	Parametri	Cifar10 Acc[%]	Parametri
CNN	84.14	395k	61.70	200k
Z2CNN	88.26	395k	68.49	201k
P4CNN	92.36	393k	67.65	237k
P4CNNB	95.62	393k	66.54	237k
Steer	95.22		72.67	

At the end of the experiment, the convolutional models have been tested on the CIFAR10+(Cohen, 2016a) and CIFAR10 with rotation datasets to increase the complexity of the dataset. Additionally, a Steerable CNN has been implemented as a generalization of GCNNs: to model the CIFAR10 dataset, the FieldType in the escnn library has to be modified. The results are reported in Table 2.

Table 2. Performance comparison.

Classes	Cifar10+ Acc[%]	Cifar10Rot Acc[%]	Parametri
CNN	60.05	43.06	200k
Z2CNN	64.88	46.52	201k
P4CNN	63.67	51.22	237k
P4CNNB	64.23	48.52	237k
Steer	73.36	64.56	

4. Discussion and conclusions

The MNIST rotated dataset achieved the best performance, demonstrating good performance with both GCNNs and CNNs despite the increased complexity. In terms of the CIFAR10 datasets, CNNs and GCNNs performed comparably due to the lack of some underlying image symmetries, but improvements have been highlighted with steerable convolution. The CIFAR10 dataset with rotations performed worse due to the dataset's complexity, making classification difficult. It's important to note that the accuracy of each model has been improved by the batch normalization layer. This work can be generalized by placing more emphasis on the concept of steerable and understanding how to learn the FieldType(Cohen, 2016b). Another interesting generalization is the consideration of group-equivariant graphs (EGCNN)(Victor Garcia Satorras, 2022), given the growing interest in these objects.

[1] https://github.com/AlehciM00/DLAI-2022

Bibliography. (Esteves, 2020) (Cohen, 2016a) (Knigge) (Cesa, c) (Cohen, 2016b) (Weiler, 2021) (Cesa, b) (Cesa, a)

References

Cesa, G. Gdl - steerable cnns, a. URL https://uvadlc-notebooks.readthedocs.io/en/latest/tutorial_notebooks/DL2/Geometric_deep_learning/tutorial2_steerable_cnns.html#.

Cesa, G. General e(2)-equivariant steerable cnns, b. URL https://quva-lab.github.io/escnn/index.html.

Cesa, G. Tutorial on group convolutional networks - ammi geometric deep learning course, c. URL https://colab.research.google.com/drive/1DfUuk-NZtW5d0toMnL752dYEMSVuNWgM?usp=sharing.

Cohen, T. Group equivariant convolutional networks, 2016a. URL https://tacocohen.files.wordpress.com/2016/06/gcnn.pdf.

Cohen, T. Steerable cnns, 2016b. URL https://arxiv.org/pdf/1612.08498.pdf.

Esteves, C. Theoretical aspects of group equivariant neural networks, 2020. URL https://arxiv.org/pdf/2004.05154.pdf.

Knigge, D. Gdl - regular group convolutions. URL https://uvadlc-notebooks.readthedocs.io/en/latest/tutorial_notebooks/DL2/Geometric_deep_learning/tutorial1_regular_group_convolutions.html.

Victor Garcia Satorras, Emiel Hoogeboom, M. W. E(n) equivariant graph neural networks, 2022.

Weiler, M. General e(2) - equivariant steerable cnns, 2021. URL https://arxiv.org/pdf/1911.08251.pdf.