Lattice Gauge Symmetry in Neural Networks

Matteo Favoni, Andreas Ipp, David I. Müller[*] and Daniel Schuh

Institute for Theoretical Physics, TU Wien
Wiedner Hauptstrasse 8-10/136, Tower B, 1040 Wien, Austria
E-mail: favoni@hep.itp.tuwien.ac.at, ipp@hep.itp.tuwien.ac.at, dmueller@hep.itp.tuwien.ac.at, schuh@hep.itp.tuwien.ac.at

We review a novel neural network architecture called lattice gauge equivariant convolutional neural networks (L-CNNs), which can be applied to generic machine learning problems in lattice gauge theory while exactly preserving gauge symmetry. We discuss the concept of gauge equivariance which we use to explicitly construct a gauge equivariant convolutional layer and a bilinear layer. The performance of L-CNNs and non-equivariant CNNs is compared using seemingly simple non-linear regression tasks, where L-CNNs demonstrate generalizability and achieve a high degree of accuracy in their predictions compared to their non-equivariant counterparts.

The 38th International Symposium on Lattice Field Theory, LATTICE2021 26th-30th July, 2021
Zoom/Gather@Massachusetts Institute of Technology

[*]Speaker

1. Introduction

The concept of symmetry or equivariance under symmetry transformations is at the theoretical foundation of modern physics, and it is hard to overstate its importance. Noether's first theorem establishes a clear relationship between invariance of Lagrangians under continuous global symmetries and the existence of conserved quantities and conserved currents [1]. Global symmetries, as the name implies, are transformations that are applied the same way at every point in space time. In mechanical systems and field theories, energy and momentum conservation laws follow from invariance under space-time translations, whereas rotational invariance implies the conservation of angular momentum. More generally, global symmetry under the Poincaré group, which includes translations, rotations and boosts, is the foundation of special relativity. In addition, symmetries not associated with isometries of space-time are of particular importance to quantum field theory. For example, global U(1) invariance in field theories of fermions and complex scalars implies the existence of locally conserved particle currents and globally conserved particle numbers. On the other hand, local symmetry, associated with continuous differentiable transformations that are functions on space time, is the foundation of gauge theories. For example, quantum chromodynamics (QCD) is a gauge theory with symmetry group $\mathrm{SU}(3)$. All known fundamental forces are formulated as gauge theories and therefore associated with particular local symmetries or gauge groups.

Machine learning methods can make use of the concept of symmetry in a similar manner. Exploiting the geometry and symmetries of a particular machine learning problem to develop appropriate neural network architectures is the main idea behind geometric deep learning (see recent reviews [2] and [3] for a mathematical introduction). Convolutional neural networks (CNNs) can be understood as special cases of generic neural networks that respect translational symmetry. In the past decades, these types of neural networks have proven to be exceptionally useful in image recognition and computer vision in general. More specifically, CNNs are neural networks consisting of convolutional layers acting on image data or feature maps $f : \mathbb{Z}^2 \to \mathbb{R}^n$, where $\mathbb{Z}^2$ is the base space on which the image is defined and $n \in \mathbb{N}$ denotes the number of channels (e.g. $n = 3$ for an RGB image). Convolutional layers are equivariant in the sense that a spatial translation of the input feature map induces an appropriate translation of the output feature map. Equivariance of CNNs naturally lends itself to applications in lattice field theory, which typically exhibit translational symmetry [4]. Going beyond translations, the framework of group equivariant convolutional neural networks (G-CNNs) [5] generalizes the equivariance property to a general group G with the traditional CNN being a special case when G is identified with the translation group. Apart from their theoretical appeal, G-CNNs with rotational symmetry have shown to be more robust on certain image recognition tasks, where traditional CNNs can fail to make correct predictions when provided with previously unseen rotated images. In the parlance of physics, G-CNNs are neural networks exhibiting global symmetry. A generalization of equivariant neural networks to local symmetries has been proposed in Ref. [6] called gauge equivariant CNNs. In this architecture, the base space on which the data is defined is generalized to a curved manifold, as opposed to a flat base space. In order to retain equivariance in convolutional layers, feature maps must be appropriately parallel transported in order to obtain the correct transformation behavior under coordinate or frame changes. Although the term "gauge equivariant" would, to physicists, imply gauge symmetry in the sense of gauge theories such as electrodynamics or QCD, parallel transport of feature maps in these types of networks is purely due to the curved geometry of the base space and not due to the presence of a gauge field. Recently, several groups have tackled the problem of using machine learning methods in the context of lattice QCD or pure lattice gauge theory, while retaining gauge symmetry in the sense of gauge theories. For example, gauge equivariant normalizing flows [7–9] have been successfully used to train generative models which can produce statistically independent lattice configurations for Monte Carlo simulations. Gauge covariant neural networks that can perform smearing and Wilson flow have also been studied [10]. However, a general framework akin to G-CNNs or gauge equivariant CNNs for lattice gauge theory has been lacking so far.

Lattice gauge theory is an exactly gauge covariant formulation of non-Abelian Yang-Mills theory discretized on a lattice originally proposed by Wilson [11]. The degrees of freedom are gauge link variables $U_{x,\mu} \in \mathrm{SU}(N_c)$ defined on the edges of a hypercubic lattice Λ with lattice spacing a. A gauge link $U_{x,\mu}$ is defined along the edge $(x, x+\mu)$ starting at the lattice site $x \in \Lambda$ and ending at $x+\mu$, which is shorthand for $x + a\hat{e}_\mu$, where $\hat{e}_\mu$ is a Euclidean basis vector. From a geometrical viewpoint, gauge links define parallel transport along the edges of the lattice. Under general gauge transformations, gauge links transform according to

$$U'_{x,\mu} = \Omega_x U_{x,\mu} \Omega^\dagger_{x+\mu}. \tag{1}$$

Reversing the path yields the inverse link which we denote by $U^\dagger_{x,\mu} = U_{x+\mu,-\mu}$. Links can be concatenated to form plaquettes (1×1 loops)

$$U_{x,\mu\nu} = U_{x,\mu} U_{x+\mu,\nu} U_{x+\mu+\nu,-\mu} U_{x+\mu,-\mu}, \qquad U'_{x,\mu\nu} = \Omega_x U_{x,\mu\nu} \Omega^\dagger_x, \tag{2}$$

which transform locally under lattice gauge transformations. The Wilson action is given by

$$S_W[U] = \frac{2}{g^2} \sum_{x \in \Lambda} \sum_{\mu < \nu} \mathrm{Tr}\left[\mathbb{1} - U_{x,\mu\nu}\right], \tag{3}$$

where $g > 0$ is the Yang-Mills coupling constant.

In these proceedings we review lattice gauge equivariant convolutional neural networks (L-CNNs), which is an equivariant neural network architecture proposed by the authors [12] specifically tailored to $\mathrm{SU}(N_c)$ lattice gauge theory. This architecture allows for neural networks that use link variables in the input layer and satisfy equivariance under general lattice gauge transformations. We discuss some aspects of L-CNNs in detail and show the main results of our computational experiments, where we compare traditional (non-equivariant) CNNs to L-CNNs in specific regression tasks.

2. Lattice gauge equivariant convolutional neural networks

The idea behind L-CNNs is to formulate CNNs which can be used to process gauge link configurations $\mathcal{U} = \{U_{x,\mu}\}$, i.e. the set of all gauge links on the lattice, in a gauge equivariant manner. In this section we review a few aspects of L-CNNs; a complete description can be found in our original paper [12]. Adopting a similar notation as in [5], a gauge equivariant function g satisfies the equivariance property

$$T_\Omega g(\mathcal{U}) = g(T_\Omega \mathcal{U}), \tag{4}$$

where T_Ω denotes the application of a general gauge transformation Ω_x. In the case of gauge links we have $T_\Omega U_{x,\mu} = \Omega_x U_{x,\mu}\Omega^\dagger_{x+\mu}$. The transformation properties of $T_\Omega g(\mathcal{U})$ generally do not need to be the same as the one for links. For example, the function $g(\mathcal{U})_x = U_{x,\mu\nu}$, which maps links to plaquettes, transforms locally

$$T_\Omega g(\mathcal{U})_x = g(T_\Omega \mathcal{U})_x = \Omega_x U_{x,\mu\nu}\Omega^\dagger_x = \Omega_x g(\mathcal{U})_x \Omega^\dagger_x. \tag{5}$$

In particular, a function is gauge invariant if $T_\Omega g = g$, i.e. if T_Ω acts as the identity map.

More generally, we consider functions of the form $g(\mathcal{U}, \mathcal{W})$ acting on a tuple $(\mathcal{U}, \mathcal{W})$ consisting of the set of links and locally transforming matrices $\mathcal{W} = \{W_{x,i}\}$ where $W_{x,i} \in \mathbb{C}^{N_c \times N_c}$ are general complex matrices and $i \in \{1, 2, \dots, N_{\mathrm{ch}}\}$ denotes different channels similar to traditional CNNs. We require that $\mathcal{W}$ variables transform locally: $T_\Omega W_{x,i} = \Omega_x W_{x,i} \Omega^\dagger_x$. For example, the set $\mathcal{W}$ may consist of all possible plaquettes on the lattice formed by links, but generally $\mathcal{W}$ variables can be considered independent of the links $\mathcal{U}$. A gauge equivariant function then satisfies

$$T_\Omega g(\mathcal{U}, \mathcal{W}) = g(T_\Omega \mathcal{U}, T_\Omega \mathcal{W}). \tag{6}$$

We formulate L-CNNs as CNNs consisting of individual layers which satisfy gauge equivariance. The idea behind the use of tuples $(\mathcal{U}, \mathcal{W})$ in L-CNNs is to explicitly split the input of the L-CNN into a feature map (or data) $\mathcal{W}$ and link variables $\mathcal{U}$, which encode the geometrical information of how data at different lattice sites can be compared using parallel transport in a manner consistent with gauge equivariance. This approach is comparable to gauge equivariant CNNs [6] where the base manifold is the "stage" on which feature maps (which can be scalar-, vector- or tensor-valued) are defined, and the connection provides the necessary information to compare feature maps at different points. Similarly, the gauge links $\mathcal{U}$ provide the stage for our feature maps $\mathcal{W}$. Figure 1a shows the data in L-CNNs schematically.

We exemplify the L-CNN using two particularly important layers: gauge equivariant convolutions and bilinear layers.

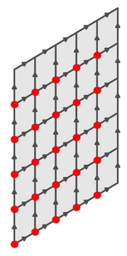
Lattice gauge equivariant convolutions (L-Convs) Convolutional layers in CNNs combine data at different points by computing a sum of data of a feature map, where each term is weighted by a (trainable) kernel. Consider a two-dimensional, single-channel feature map $f : \mathbb{Z}^2 \to \mathbb{R}$, then a convolution can be defined as

$$f'_x = \sum_{y \in \mathbb{Z}^2} \omega_{x-y} f_y, \tag{7}$$

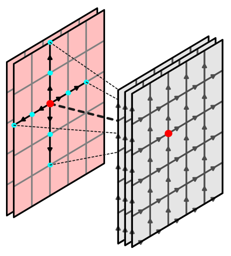
where $\omega_{x-y} \in \mathbb{R}$ are the kernel weights and f' is the new feature map after the convolution. The largest distance considered in a convolution defines the size $K \in \mathbb{N}$ of the kernel, i.e. we assume the kernel to be compact. Computing a standard convolution of $\mathcal{W}$ variables would violate the requirement of gauge equivariance, unless one accounts for parallel transport:

$$W'_{x,i} = \sum_y \omega_{x-y} U_{x\leftarrow y} W_{y,i} U^\dagger_{x\leftarrow y}, \tag{8}$$

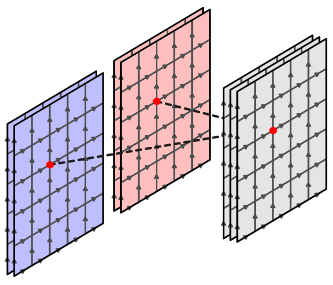
where $U_{x\leftarrow y}$ is a Wilson line on the lattice tracing a path from x to y. However, the choice of the path on the lattice is not unique. Whereas in the continuum one may only consider straight-line



(a) Lattice data $(\mathcal{U}, \mathcal{W})$



(b) L-Conv layer



(c) L-Bilin layer

Figure 1: Schematic overview of different L-CNN layers. (a) The data $(\mathcal{U}, \mathcal{W})$ considered in L-CNNs is a tuple consisting of links $\mathcal{U}$ (gray edges) and locally transforming matrices $\mathcal{W}$ (red dots). (b) A lattice gauge equivariant convolution computes a weighted sum of parallel transported data (cyan points) along straight lines (black lines) up to a certain kernel size (here, $K = 2$). (c) A lattice gauge equivariant bilinear layer multiplies data at a common lattice site (red dot), combining different channels in the process (blue and red planes).

paths (or more generally geodesics), there are multiple shortest paths connecting x and y on the lattice. A possible solution is to only take straight paths along the lattice axes into account. Thus, we define the lattice gauge equivariant convolution (L-Conv) as

$$W'_{x,i} = \sum_{j,\mu,k} \omega_{i,j,\mu,k} U_{x,k\cdot\mu} W_{x+k\cdot\mu,j} U^\dagger_{x,k\cdot\mu}, \tag{9}$$

where $\omega_{i,j,\mu,k} \in \mathbb{C}$ is a trainable kernel with $1 \leq i \leq N_{\mathrm{ch,out}}$, $1 \leq j \leq N_{\mathrm{ch,in}}$, $0 \leq \mu \leq D$ and $-K \leq k \leq K$, where K is the kernel size. The parallel transport from x to $x + k \cdot \mu$ is given by

$$U_{x,k\cdot\mu} = \prod_{i=0}^{k-1} U_{x+i\cdot\mu,\mu} = U_{x,\mu} U_{x+\mu,\mu} U_{x+2\cdot\mu,\mu} \dots U_{x+(k-1)\cdot\mu,\mu}. \tag{10}$$

We allow for possible mixing of channels, and the number of channels can change from $N_{\mathrm{ch,in}}$ to $N_{\mathrm{ch,out}}$. Additionally, one may enlarge the channels of $\mathcal{W}$ by unit matrices prior to computing the convolution via

$$(W_{x,1}, W_{x,2}, \dots, W_{x,N_{\mathrm{ch,in}}}) \to (\mathbb{1}, W_{x,1}, W_{x,2}, \dots, W_{x,N_{\mathrm{ch,in}}}), \tag{11}$$

to allow for a bias term. Even more expressivity is gained by also including hermitian conjugates

$$(W_{x,1}, W_{x,2}, \dots, W_{x,N_{\mathrm{ch,in}}}) \to (\mathbb{1}, W_{x,1}, W_{x,2}, \dots, W_{x,N_{\mathrm{ch,in}}}, W^\dagger_{x,1}, W^\dagger_{x,2}, \dots, W^\dagger_{x,N_{\mathrm{ch,in}}}). \tag{12}$$

We note that L-Convs, similar to any convolutional layer, are equivariant under translations of the lattice. Figure 1b visualizes the L-Conv layer.

Lattice gauge equivariant bilinear layers (L-Bilin) Parallel transport is required when comparing data at different lattice sites, but one can easily define operations acting only on single points without violating gauge equivariance. An example for such a layer in the L-CNN is the lattice gauge

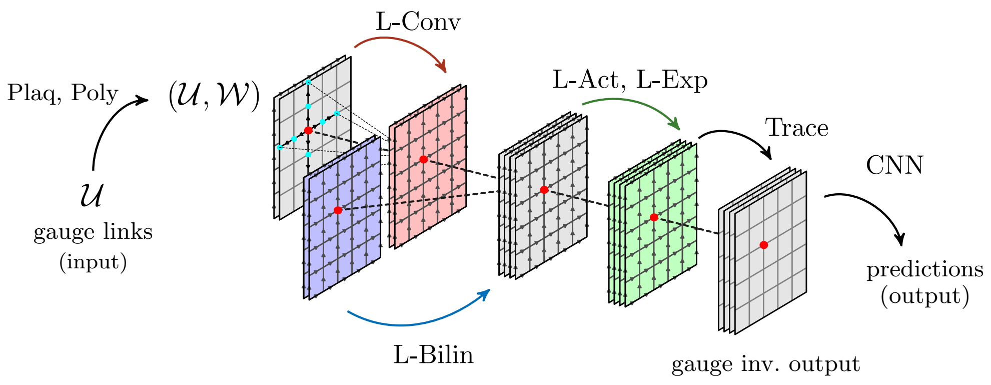


Figure 2: Schematic of a feed-forward L-CNN consisting of multiple layers. Links are first pre-processed to generate plaquettes (and/or Polyakov loops) before being fed to L-Conv and L-Bilin layers. Activation functions and exponentiation layers may also be used. For gauge invariant predictions, a Trace layer is required. Figure adapted from [12].

equivariant bilinear layer (L-Bilin) which combines two tuples $(\mathcal{U}, \mathcal{W})$ and $(\mathcal{U}, \mathcal{W}')$ into a single tuple $(\mathcal{U}, \mathcal{W}'')$ in a bilinear manner. We define L-Bilin operations via

$$W''_{x,i} = \sum_{j,k} \alpha_{i,j,k} W_{x,j} W'_{x,k}, \tag{13}$$

where $\alpha_{i,j,k} \in \mathbb{C}$ are trainable weights with $1 \leq i \leq N_{\text{out}}$, $1 \leq j \leq N_{\text{in},1}$ and $1 \leq k \leq N_{\text{in},2}$. The product of the two matrices $W_{x,j}$ and $W'_{x,k}$ yields a matrix that transforms locally at x and is thus compatible with gauge equivariance. L-Bilin is linear in each of the two arguments $\mathcal{W}$ and $\mathcal{W}'$, however one may also choose $\mathcal{W}' = \mathcal{W}$ yielding a quadratic operation. Similar to L-Convs, the channels of $\mathcal{W}$ can be extended by unit matrices and hermitian conjugates. This way the bilinear layer contains a bias term and linear terms as well (through multiplication with unit elements of the other argument).

Other layers and the expressivity of L-CNNs Our paper [12] proposes more types of layers, which we only briefly summarize here. For example, one may combine an L-Conv and an L-Bilin layer into a single operation, which we found to be easier to train than separate convolutions and bilinear layers. Other types are lattice gauge equivariant activation functions (L-Act), which locally multiply $\mathcal{W}$ variables with gauge invariant scalars, and lattice gauge equivariant exponentiation layers (L-Exp), which can modify the set of gauge links $\mathcal{U}$. Another important layer is the Trace layer, which maps $(\mathcal{U}, \mathcal{W})$ to gauge invariant variables $\mathcal{T}_{x,i}$ via

$$\mathcal{T}_{x,i} = \mathrm{Tr}\left[W_{x,i}\right]. \tag{14}$$

This layer is used when the output of the L-CNN is required to be invariant, e.g. if the L-CNN is used to fit invariant observables. To keep the discussion as general as possible, we have not yet specified what information the $\mathcal{W}$ variables should contain, except requiring them to transform locally. As a pre-processing step to any L-CNN, one may generate $\mathcal{W}$ from the set of gauge links $\mathcal{U}$ by computing all plaquettes (which we refer to as a Plaq layer) and Polyakov loops (Poly layer).

Generic L-CNNs are comprised of multiple layers, as shown in Fig. 2, similar to standard CNNs. It is well known that deep CNNs can describe arbitrary continuous functions (see e.g. universality

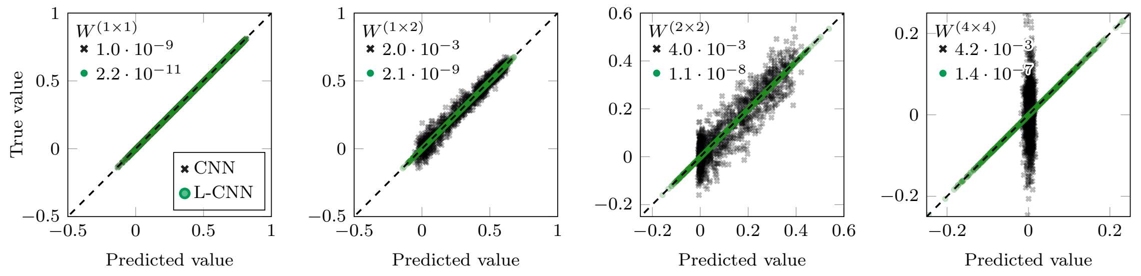


Figure 3: Scatter plot of our best L-CNNs (green circles) vs. our best CNNs (black crosses) on 8×8 test data for various sizes of the Wilson loop. The dashed line indicates perfect agreement between prediction and true value. MSEs are stated in the top left corner of each panel. Figure from [12].

theorems in [13]). The question then arises as to what class of functions can be represented using L-CNNs. In [12] we show that by stacking multiple L-Conv and L-Bilin layers and using plaquettes as input $\mathcal{W}$ variables, L-CNNs can generate contractible Wilson loops of arbitrary shape and size. Non-contractible Wilson loops can also be generated by including Polyakov loops in the input layer. Similar to universality theorems for deep CNNs, the use of L-Act layers allows arbitrary non-linear gauge equivariant functions to be expressed as L-CNNs.

3. Computational experiments

In order to test the L-CNN framework in practice, we study a series of seemingly simple regression problems. Restricting the base space to a quadratic two dimensional lattice (8×8 up to 64×64) and focusing on SU(2), we generate gauge field configurations from a Markov Chain Monte Carlo simulation at various values of the coupling constant. These configurations are split into separate training, validation and test datasets. For each individual configuration, we compute the real value of traced Wilson loops of various sizes:

$$W^{(m\times n)}_{x,\mu\nu} = \frac{1}{N_c} \mathrm{Re}\,\mathrm{Tr}\left[U^{(m\times n)}_{x,\mu\nu}\right], \tag{15}$$

where $n, m \geq 1$ define the width and height of the loop. We generate loops of size 1×1 (plaquettes), 1×2, 2×2 and 4×4. We then formulate a regression problem where the lattice configurations should be mapped to the value of a particular traced loop using different L-CNN and non-equivariant CNN architectures, i.e. the input is given by the links $\mathcal{U}$ and the desired output (labels) are $W^{(m\times n)}_{x,\mu\nu}$. L-CNN architectures consist of multiple L-Conv+L-Bilin layers and a Trace layer at the end, similar to Fig. 2. For comparison, we study a wide range of different non-equivariant CNN architectures of various widths, depths and employing different activation functions. The CNNs are provided with the same input as the L-CNN after pre-processing, namely links and plaquettes. Both types of architectures are implemented and trained using the *PyTorch* framework. Networks are optimized by minimizing the mean squared error (MSE) with respect to trainable weight parameters. Training and validation are performed on the smallest lattice size (8×8), but models are tested also on larger lattice sizes.

Figure 3 shows one of the main results of this study, where the predictions of an L-CNN and a CNN are plotted against the true values (labels) of the test dataset. As we have trained multiple networks of both types (100 L-CNNs, 2680 CNNs), we only show the best models encountered in our study, which are selected using MSE on the validation dataset. We observe that L-CNN architectures are able to make sensible predictions in all cases. On the other hand, the performance of non-equivariant CNNs starts to deteriorate with larger loop size. In the case of the largest loops, CNNs seem to find no correlation between input and output at all, which is indicated by an almost constant prediction in the scatter plot (rightmost panel in Fig. 3). Furthermore, even our best CNNs are sensitive to gauge transformations, i.e. the output of a non-equivariant CNN can change drastically when processing a gauge equivalent lattice configuration, whereas L-CNNs are gauge invariant by construction. We have also demonstrated that L-CNNs are able to solve similar regression problems on four-dimensional lattices (up to 4×4 Wilson loops on $8 \cdot 16^3$ lattices). We refer the reader to the original paper [12] for more details. Our code and datasets for these computational experiments can be found in our GitLab repository[1].

4. Summary and outlook

In these proceedings we have highlighted some aspects of the L-CNN architecture with a particular focus on the concept of gauge equivariance and motivating the formulation of L-Conv and L-Bilin layers. The L-CNN is a general, genuinely gauge equivariant neural network that can be used for generic machine learning tasks in $\mathrm{SU}(N_c)$ lattice gauge theory. In our computational experiments we have demonstrated that L-CNNs are able to solve non-linear regression problems, where non-equivariant CNNs fail to make sensible predictions.

Up until now we have not made use of (or implemented) L-Act and L-Exp layers, as our main goal was to establish that L-CNNs can be used to generate arbitrary loops, which requires only L-Conv and L-Bilin layers. Although we have only tested $\mathrm{SU}(2)$, our implementation works for any $\mathrm{SU}(N_c)$ gauge group. Using slight modifications, the code could also be adapted for $\mathrm{U}(1)$. However, the L-CNN code requires large computational resources during training, in particular with regards to memory consumption, and a more efficient implementation would be desirable. This is of particular importance when using larger lattice sizes in four dimensions or studying more complicated problems.

L-CNNs have been explicitly formulated for $\mathrm{SU}(N_c)$ lattice gauge theory, which is a discretization of (continuum) non-Abelian gauge theory. Although such a lattice formulation is required to perform practical computations, a more general mathematical formulation in terms of principal bundles (similar to gauge equivariant CNNs [6]) would be desirable from a theoretical viewpoint. Putting the L-CNN on better theoretical foundations could help us generalize L-CNNs to other types of data such as fermionic fields and develop new equivariant layers.

Our computational experiments focused on toy model regression problems, which we conducted to compare the performance of L-CNNs and CNNs in a transparent way. It would be interesting to see how L-CNNs (in particular L-Convs) could be applied to more practical problems in e.g. normalizing flow models [7–9] or models that modify gauge links [10]. Another exciting direction would be to use L-CNNs to formulate gauge equivariant continuous flows [14] to potentially speed up the generation of lattice configurations for independent Monte Carlo sampling, while retaining a large

[1]See https://gitlab.com/openpixi/lge-cnn/.

degree of symmetry. A related problem would be to use L-CNNs to upscale real-time simulations of classical lattice gauge fields used in the early stages of relativistic heavy-ion collisions [15].

Acknowledgments

This work has been supported by the Austrian Science Fund FWF No. P32446-N27, No. P28352 and Doctoral program No. W1252-N27. The Titan V GPU used for this research was donated by the NVIDIA Corporation.

References

[1] E. Noether, *Invariante Variationsprobleme*, *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse* **1918** (1918) 235.

[2] M.M. Bronstein, J. Bruna, T. Cohen and P. Veličković, *Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges*, 2104.13478.

[3] J.E. Gerken, J. Aronsson, O. Carlsson, H. Linander, F. Ohlsson, C. Petersson et al., *Geometric Deep Learning and Equivariant Neural Networks*, 2105.13926.

[4] S. Bulusu, M. Favoni, A. Ipp, D.I. Müller and D. Schuh, *Generalization capabilities of translationally equivariant neural networks*, *Phys. Rev. D* **104** (2021) 074504 [2103.14686].

[5] T.S. Cohen and M. Welling, *Group equivariant convolutional networks*, in *Proceedings of The 33rd International Conference on Machine Learning*, vol. 48, pp. 2990–2999, JMLR, Jun, 2016 [1602.07576].

[6] T.S. Cohen, M. Weiler, B. Kicanaoglu and M. Welling, *Gauge equivariant convolutional networks and the icosahedral CNN*, in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97, pp. 1321–1330, JMLR, Jun, 2019 [1902.04615].

[7] G. Kanwar, M.S. Albergo, D. Boyda, K. Cranmer, D.C. Hackett, S. Racanière et al., *Equivariant flow-based sampling for lattice gauge theory*, *Phys. Rev. Lett.* **125** (2020) 121601 [2003.06413].

[8] D. Boyda, G. Kanwar, S. Racanière, D.J. Rezende, M.S. Albergo, K. Cranmer et al., *Sampling using $SU(N)$ gauge equivariant flows*, *Phys. Rev. D* **103** (2021) 074504 [2008.05456].

[9] M.S. Albergo, D. Boyda, D.C. Hackett, G. Kanwar, K. Cranmer, S. Racanière et al., *Introduction to Normalizing Flows for Lattice Field Theory*, 2101.08176.

[10] A. Tomiya and Y. Nagai, *Gauge covariant neural network for 4 dimensional non-abelian gauge theory*, 2103.11965.

[11] K.G. Wilson, *Confinement of quarks*, *Phys. Rev. D* **10** (1974) 2445.

[12] M. Favoni, A. Ipp, D.I. Müller and D. Schuh, *Lattice gauge equivariant convolutional neural networks*, 2012.12901.

[13] D.-X. Zhou, *Universality of deep convolutional neural networks*, *Applied and Computational Harmonic Analysis* **48** (2020) 787 [1805.10769].

[14] P. de Haan, C. Rainone, M. Cheng and R. Bondesan, *Scaling Up Machine Learning For Quantum Field Theory with Equivariant Continuous Flows*, 2110.02673.

[15] A. Ipp and D.I. Müller, *Progress on 3+1D Glasma simulations*, *Eur. Phys. J. A* **56** (2020) 243 [2009.02044].