Machine learning with gauge symmetry

Matteo Favoni

Institute for Theoretical Physics, TU Wien
April 6, 2022

Quark Matter 2022

Based on: M. Favoni, A. Ipp, D. I. Müller, D. Schuh, Phys.Rev.Lett. **128**, 032003

Code: gitlab.com/openpixi/lge-cnn

TECHNISCHE UNIVERSITÄT WIEN Vienna | Austria

Der Wissenschaftsfonds.

Doktoratskolleg Particles and Interactions

Introduction

- Neural networks (NNs) are a widely used tool in many scientific areas
- NNs are universal approximators of any given function
- In general, symmetries in data are learnt, therefore only approximated
- Good strategy: meet the requirements of the specific problem
- In quantum field theories, symmetries play a key role
- A desirable approach is to design NNs so that such symmetries are respected
- Example: translational symmetry is incorporated by construction in convolutional neural networks (CNNs) (under certain circumstances)
- Here: build NNs respecting gauge symmetries by construction

Lattice gauge theories

- Gauge links $\mathcal{U}$ and 1×1 loops $\mathcal{W}$

$$U_{\mathbf{x},\mu} \simeq \exp\left(iga^{\mu}A_{\mu}(\mathbf{x}+\mathbf{a}^{\mu}/2)\right) \in \mathrm{SU}(N_c)$$
$$W_{\mathbf{x},\mu\nu} = U_{\mathbf{x},\mu}U_{\mathbf{x}+\mu,\nu}U_{\mathbf{x}+\mu+\nu,-\mu}U_{\mathbf{x}+\nu,-\nu}$$

- Lattice gauge transformations for $\mathcal{U}$ and $\mathcal{W}$

$$T_{\Omega}U_{\mathbf{x},\mu} = \Omega_{\mathbf{x}}U_{\mathbf{x},\mu}\Omega^{\dagger}_{\mathbf{x}+\mu}, \qquad \Omega_{\mathbf{x}} \in \mathrm{SU}(N_c)$$
$$T_{\Omega}W_{\mathbf{x},\mu\nu} = \Omega_{\mathbf{x}}W_{\mathbf{x},\mu\nu}\Omega^{\dagger}_{\mathbf{x}}$$

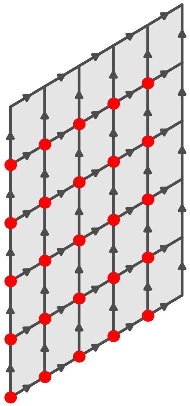
- Gauge equivariant function

$$g(T_{\Omega}\mathcal{U}, T_{\Omega}\mathcal{W}) = T'_{\Omega}g(\mathcal{U},\mathcal{W})$$

- Gauge invariant function (e.g. observables, action)

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

- Neural network layers designed to respect gauge equivariance

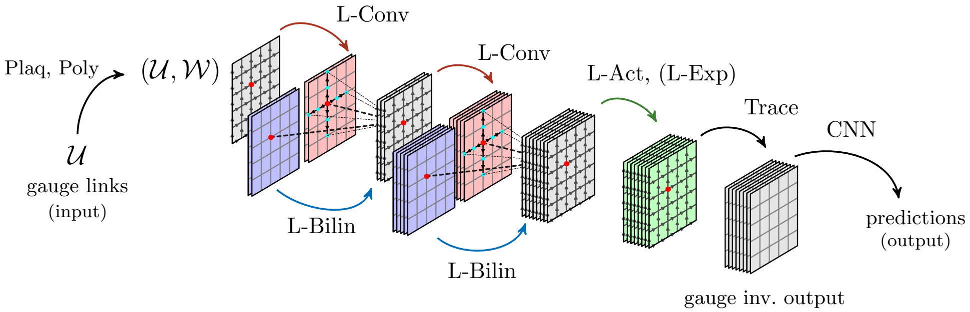


$\mathcal{U} = \{U_{\mathbf{x},\mu}\}$

$\mathcal{W} = \{W_{\mathbf{x},\mu\nu}\}$

Lattice gauge equivariant convolutional neural networks

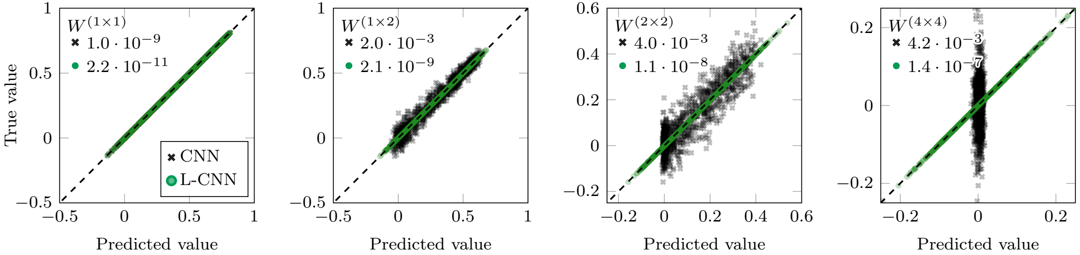
L-CNNs: A collection of gauge equivariant layers for lattice gauge configurations



- Preprocessing: **Plaq, Poly**
  $\mathcal{U} \to (\mathcal{U}, \mathcal{W})$
  [no trainable parameters]
- Convolutions: **L-Conv**
  $(\mathcal{U}, \mathcal{W}) \to (\mathcal{U}, \mathcal{W}')$
  [Conv + parallel transport]
- Bilinear layer: **L-Bilin**
  $(\mathcal{U}, \mathcal{W}) \to (\mathcal{U}, \mathcal{W} \cdot \mathcal{W}')$
  [local matrix mult.]
- Activation layer: **L-Act**
  $(\mathcal{U}, \mathcal{W}) \to (\mathcal{U}, a(\mathcal{W}) \cdot \mathcal{W})$
  [local scalar mult.]
- Exponential maps: **L-Exp**
  $(\mathcal{U}, \mathcal{W}) \to (e^{i\omega} \cdot \mathcal{U}, \mathcal{W})$
  [local matrix mult.]
- Postprocessing: **Trace**
  Invariant output
  [no trainable parameters]

Performance of standard CNNs vs LCNNs

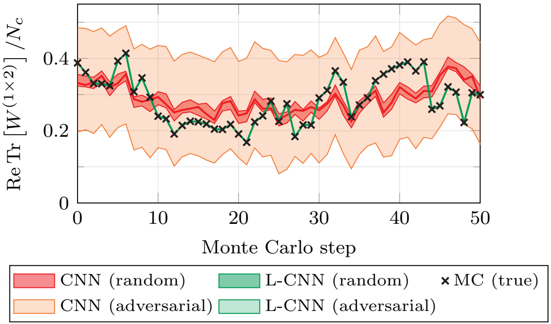
Benchmark problem: regression of Wilson loops from 1×1 to 4×4 on 2D lattice

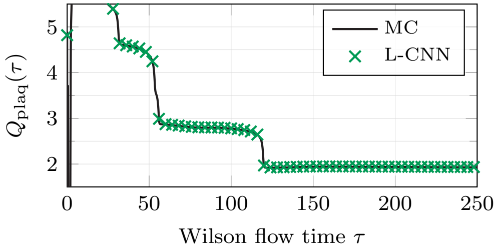


True vs. predicted values for CNNs and L-CNNs for $n \times m$ Wilson loops (best models)

- Increasing loop size from left to right $\rightarrow$ increasingly harder problem
- Deteriorating performance of baseline CNNs with increased loop size
- Best L-CNN **always** beats best baseline CNN
- Consistent performance of L-CNNs across all loop and lattice sizes

Additional tests



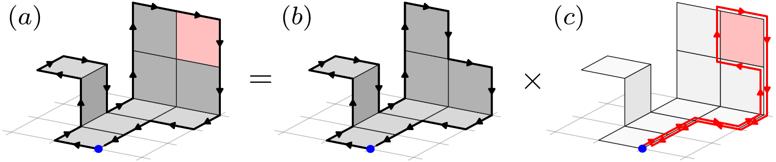


- Test for sensitivity to random and adversarial gauge transformations for 1×2 Wilson loop: baseline CNNs are sensitive to gauge transformations, L-CNNs are invariant
- L-CNN predictions vs. true values (MC) for Q_{plaq} on a 8×24^3 configuration: L-CNNs also work in 3+1 dimensions

Backup slides

Construction of arbitrary Wilson loops

- Repeated applications of **L-Conv** and **L-Bilin** operations can be used to generate arbitrarily sized Wilson loops if input $\mathcal{W}$ consists of plaquettes (preprocessing layer **Plaq**)



- Non-contractible loops can also be generated by including Polyakov loops in the input $\mathcal{W}$ (preprocessing layer **Poly**)
- Non-linear functions of Wilson loops are possible through **L-Act**, **Trace** and passing gauge invariant output to traditional CNNs
- L-CNNs are universal approximators for gauge invariant functions on the lattice