Group equivariant networks for leakage detection in vacuum bagging

Christoph Brauer*†, Dirk Lorenz*, Lionel Tondji*
*Institute of Analysis and Algebra, Technical University of Braunschweig, Braunschweig, Germany
†Institute of Composite Structures and Adaptive Systems, German Aerospace Center, Stade, Germany
Email: {ch.brauer, d.lorenz, l.ngoupeyou-tondji}@tu-braunschweig.de

***Abstract*—The incorporation of prior knowledge into the machine learning pipeline is subject of informed machine learning. Spatial invariances constitute a class of prior knowledge that can be taken into account especially in the design of model architectures or through virtual training examples. In this contribution, we investigate fully connected neural network architectures that are equivariant with respect to the dihedral group of order eight. This is practically motivated by the application of leakage detection in vacuum bagging which plays an important role in the manufacturing of fiber composite components. Our approach for the derivation of an equivariant architecture is constructive and transferable to other symmetry groups. It starts from a standard network architecture and results in a specific kind of weight sharing in each layer. In numerical experiments, we compare equivariant and standard networks on a novel leakage detection dataset. Our results indicate that group equivariant networks can capture the application specific prior knowledge much better than standard networks, even if the latter are trained on augmented data.**

***Index Terms*—geometric deep learning, neural networks, equivarince, group symmetry**

I. Introduction

The production of fiber composite components requires that stacked and impregnated materials are cured by means of heat and pressure. During this curing process one usually vacuumizes the structure in order to apply sufficient pressure. The quality of the vacuum is crucial for the quality of the product [1], [2]. However, small leakages in the vacuum bag result in lower pressure and hence, lower the quality. While the presence of a leakage is easy to *detect* by residual mass flow through the vacuum pumps, it is much harder to *locate* [3], [4]. This results in long delays during production, since one has to find and patch any leakage in the vacuum bag. In this work, we derive a method to locate leakages from the steady-state data of mass flow rates at the vacuum pumps.

Our setup is shown in Figure 1: Breather cloth and bagging film are superimposed on a square table with a side length of one and a half meters. There are four equal vacuum pumps in the four corners of the table and the structure is made airtight by means of sealant tape along the boundary. The breather material provides an air conducting layer below the vacuum film that prevents air inclusions before full evacuation. We can measure the mass flow in each pump in dependence of time, but we will only use the steady-state data of the vacuum pumps, *i.e.*, the residual mass flow that is still present after full evacuation. Our goal is to develop a neural network that predicts the position of a leakage from the mass flow rates at the four pumps. Hence, the input of the neural network will be four positive real numbers and the output of the network shall be two-dimensional coordinates, thus just two real numbers.

To produce measurement data, the airtight setup is first manually pierced at a random position using a hypodermic needle (see Figure 1 top right). Then, valves are closed and residual mass flow rates at all four vacuum pumps are recorded. After patching the leakage, we iterated this procedure until we generated 317 samples. Since the described procedure was quite time consuming (it took about 40 hours), we have to work with limited data and thus, we would like to enrich the leakage detection with as much additional knowledge as possible. Due to the square setup of the table, we can be sure that there are certain symmetries, namely the full symmetries of a square, *i.e.*, if we rotate the full table, the position of the leakage is rotated in the same way we rotate the vacuum pumps. The same applies to mirroring along the four axes of symmetry.

The incorporation of prior knowledge into the machine learning pipeline has a long history and comes in different shapes. Following the taxonomy recently proposed in [5], informed machine learning approaches can be classified based on knowledge source, representation and integration. In terms of that framework, spatial invariances constitute a specific type of knowledge representation that is usually integrated through the training data [6]–[8] or through the hypothesis set [9]–[11]. Convolutional neural networks [12]–[14] have a built-in translation invariance. The term *equivariance* has recently complemented this property, and is one of the basic principles for geometric deep learning [15], [16]. Related approaches usually integrate equivariance with respect to certain group operations directly into CNN architectures [17]–[20] and are thus further examples of knowledge integration through the hypothesis set. In this work, we investigate the direct integration of group equivariance constraints into *fully connected* architectures, *i.e.*, through the hypothesis set (Section II). It is possible to create an augmented dataset that reflects the considered equivariance constraints in terms of the original dataset. As part of our numerical experiments, we compare equivariant and standard architectures that were trained on both original and augmented data (Section III).

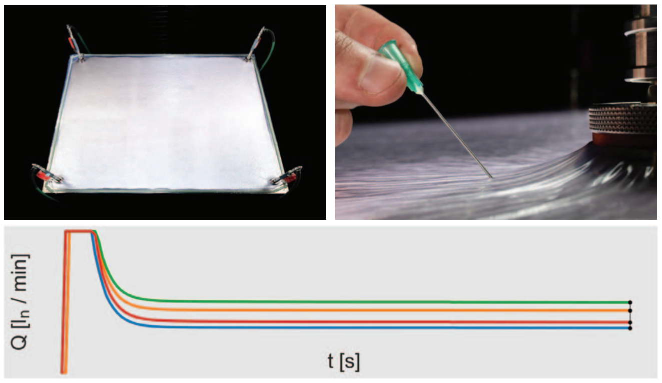


Fig. 1: Experimental setup and input data. A previously airtight vacuum setup (top left) is pierced by means of a hypodermic needle (top right). Residual flows at four vacuum pumps are recorded through volumetric flow meters and used as inputs.

II. Equivariant Architecture

In our application, we aim to learn a function f_θ that predicts leakage coordinates $y \in [-1,1]^2$ from normalized flow rates $x \in \Delta := \{x \in \mathbb{R}^4 \mid \sum_j x_j = 1,\ x \geq 0\}$. For that purpose, we have training data $(x^i, y^i)_{i=1}^m$ available and we know that certain symmetries are present: For a fixed vector of flow rates $x = (\mu_1, \dots, \mu_4)$ and associated leakage coordinates y, suppose that the position of the leakage is rotated clockwise by 90°. By symmetry, the rotated coordinates y_{90} should be associated with accordingly shifted flow rates $x_{90} = (\mu_4, \mu_1, \mu_2, \mu_3)$. By analogous arguments, we can also perform a flip along the vertical axis and claim that the resulting coordinates y_{flipped} come along with flow rates $x_{\text{flipped}} = (\mu_2, \mu_1, \mu_4, \mu_3)$.

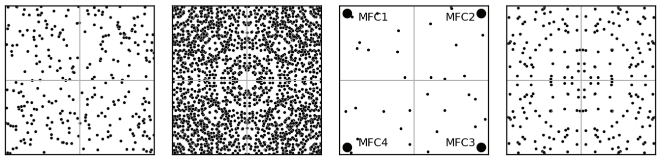
A. Group theoretic view

In mathematical terms, we have that the input-output relation is equivariant with respect to the symmetry group of the square, also called dihedral group of order 8 and denoted by D_4 (see [21] for a hands-on introduction to group theory). We use two different representations of the square, one through the four corners, *i.e.*, through vectors in $\mathbb{R}^4$ (the vector of the normalized flow rates) and one through coordinates in $[-1,1]^2$ (for the position of the leakage). If T represents one symmetry operation for the first representation in $\mathbb{R}^4$ (*i.e.*, $T \in \mathbb{R}^{4\times 4}$) and t is the respective operation for the second representation (*i.e.*, $t \in \mathbb{R}^{2\times 2}$), we require that our network fulfills the equivariance condition

$$\forall x \in \Delta : (f_\theta \circ T)(x) = (t \circ f_\theta)(x)\ . \tag{1}$$

The group D_4 has eight elements (identity, counter-clockwise rotations by $90°, 180°$, and $270°$, and four flips along the four axes of symmetry) but is generated by just two elements, namely one rotation by $90°$ and one flip. Hence, we only need to require that

$$(f_\theta \circ R)(x) = (r \circ f_\theta)(x) \quad \text{and} \tag{2}$$
$$(f_\theta \circ S)(x) = (s \circ f_\theta)(x) \tag{3}$$



(a) Train (b) Train DA (c) Test (d) Test DA

Fig. 2: Target data. The original training data (a) comprises 285 examples. Applying all group operations of the dihedral group D_4 yields an augmented dataset (b) including 2280 examples. The original test set (c) and its augmented counterpart (d) comprise 32 and 256 examples, respectively.

for matrices

$$R := \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad r := \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{4}$$

(representing a counterclockwise rotation by 90°) and

$$S := \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \text{and} \quad s := \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \tag{5}$$

(representing a flip along the vertical axis). A rigorous proof will be given in the next subsection.

B. Sufficient conditions

To find a convenient architecture satisfying (2)–(3), we start from a standard fully connected neural network whose input $a^0 := x$ is propagated through layers

$$a^\ell := g(W^\ell a^{\ell-1} + b^\ell) \quad \text{for} \quad \ell = 1, \dots, L \tag{6}$$

and that generates an output $f_\theta(x) := a^L$. The parameters of the network are the weight matrices W^ℓ and the bias vectors b^ℓ. In the following, we assume that the activation function g is applied componentwise in case of vector-valued inputs. Moreover, we drop all bias vectors and hence, $b^\ell = 0$ holds throughout, *i.e.*, our network is

$$f_\theta(x) = g(W^L g(\cdots g(W^1 x) \cdots))\ . \tag{7}$$

Finally, we assume that $W^\ell \in \mathbb{R}^{4\times 4}$ for $\ell = 1, \dots, L-1$ and $W^L \in \mathbb{R}^{2\times 4}$, *i.e.*, we restrict ourselves to neural networks with four neurons per hidden layer. Our notation suggests that the same activation function is used in all layers. However, this is only for convenience and each layer could be equipped with its own activation function. All the following statements stay valid as long as all activation functions are applied componentwise.

One easily sees that f_θ from (7) fulfills (2) and (3) if every layer fulfills the correct equivariance, *i.e.*, if we have for $T \in \{R, S\}$ and $t \in \{r, s\}$ and $\ell = 1, \dots, L-1$ that

$$g(W^\ell T a) = T g(W^\ell a) \tag{8}$$

and for the last layer that

$$g(W^L T a) = t g(W^L a)\,. \tag{9}$$

Based on that, we deduce sufficient properties of the matrices W^ℓ such that f_θ satisfies (2)–(3).

Lemma 1. *If all conditions* (8) *and* (9) *hold for $T \in \{R, S\}$ and $t \in \{r, s\}$, respectively, then f_θ as defined in (7) satisfies (1) for all $T \in \mathbb{R}^{4\times4}$ and $t \in \mathbb{R}^{2\times2}$ that represent an arbitrary operation from the dihedral group D_4.*

Proof. All representations (T, t) of a group operation in D_4 can be generated in terms of (R, r) and (S, s), namely there exist $k \in \{0, \dots, 3\}$ and $l \in \{0, 1\}$ such that $T = R^k S^l$ and $t = r^k s^l$. With that said, repeated applications of (8) and (9) show that

$$g(W^\ell T a) = T g(W^\ell a) \text{ and } g(W^L T a) = t g(W^L a)\,.$$

Hence, conditions (8) and (9) hold true for arbitrary operations in D_4. □

As a consequence of this, weight matrices that comply with conditions (8) and (9) are sufficient for the resulting model to be equivariant. Hence, our next goal is to derive explicit matrix structures from these conditions that can be incorporated into trainable models.

C. Architecture derivation

Next, we derive the intended matrix structures. First, we consider weight matrices W^ℓ and afterwards W^L that shall satisfy (8) and (9).

1) Hidden layers: For the full network to be equivariant, it is sufficient to fulfill the matrix equations $W^\ell R = R W^\ell$ and $W^\ell S = S W^\ell$. As both R and S are permutation matrices, both equations boil down to a very specific structure. If $\pi = (\pi(1), \pi(2), \pi(3), \pi(4))$ denotes the permutation that is associated with a permutation matrix $T \in \mathbb{R}^{4\times4}$, and w_{ij} indicate the entries of $W^\ell \in \mathbb{R}^{4\times4}$, then the matrix equation $W^\ell T = T W^\ell$ is equivalent to

$$w_{ij} = w_{\pi(i)\pi(j)} \quad \text{for} \quad i, j = 1, \dots, 4\,. \tag{10}$$

Hence, it is enough to look for cycles in the form of

$$((i,j),\, (\pi(i), \pi(j)), \dots, (\pi^\gamma(i), \pi^\gamma(j)))\;, \tag{11}$$

i.e., $\gamma + 1$ applications of π yield (i, j) again. The search is stopped as soon as all indices are contained in a cycle. As a consequence, $W^\ell T = T W^\ell$ holds if and only if the entries

$$w_{ij} = w_{\pi(i)\pi(j)} = \dots = w_{\pi^\gamma(i)\pi^\gamma(j)} \tag{12}$$

are equal for each such cycle.

Theorem 2. *Every matrix satisfying* (8) *is of the form*

$$W^\ell := \begin{bmatrix} a & b & c & b \\ b & a & b & c \\ c & b & a & b \\ b & c & b & a \end{bmatrix} \quad \textit{with } a, b, c \in \mathbb{R}. \tag{13}$$

Proof. R represents the permutation $\pi_R = (4, 1, 2, 3)$. A search for cycles according to (10)–(12) yields that W^ℓ is constant along the diagonals and the same procedure applied to S and the related permutation $\pi_S = (2, 1, 4, 3)$ results in the following equations:

$$\begin{array}{llll} w_{11} = w_{22} & w_{12} = w_{21} & w_{13} = w_{24} & w_{14} = w_{23} \\ w_{31} = w_{42} & w_{32} = w_{41} & w_{33} = w_{44} & w_{34} = w_{43} \end{array}$$

Putting both sets of equations together and removing redundancies shows that a matrix W^ℓ has the claimed structure. □

2) Output layer: Our procedure to find a matrix that satisfies (9) is similar to the one just described. The difference lies in the fact that the matrix equations involve a 4×4 permutation matrix T *and* a 2×2 matrix t and the latter is a composition of a permutation matrix and a possible sign flip. So let again π denote the permutation associated with T. Moreover, let $\tau = (\tau_1, \tau_2)$ be the permutation related to t, and let $\sigma = (\sigma_1, \sigma_2)$ denote the corresponding sign flip. To make this clear, consider the following example: For

$$t = \left[\begin{smallmatrix} 0 & -1 \\ 1 & 0 \end{smallmatrix}\right] \quad \text{we have} \quad \tau = (2, 1),\ \sigma = (1, -1)\,. \tag{14}$$

Therewith, the matrix equation $W^L T = t W^L$ is equivalent to

$$w_{ij} = \sigma(i) w_{\tau(i)\pi(j)} \text{ for } i = 1, 2 \text{ and } j = 1, \dots, 4 \tag{15}$$

and again this can be used to identify cycles

$$((i,j),\, (\tau(i), \pi(j)), \dots, (\tau^\gamma(i), \pi^\gamma(j))) \tag{16}$$

and derive according equations

$$w_{ij} = \sigma(i) w_{\tau(i)\pi(j)} = \dots = \left(\prod_{k=0}^{\gamma-1} \sigma(\tau^k(i))\right) w_{\tau^\gamma(i)\pi^\gamma(j)} \tag{17}$$

that are equivalent to the matrix equation.

Theorem 3. *Every matrix satisfying* (9) *has the form*

$$W^L := \begin{bmatrix} a & -a & -a & a \\ -a & -a & a & a \end{bmatrix} \quad \textit{with } a \in \mathbb{R}. \tag{18}$$

Proof. As stated above, R represents the permutation $\pi_R = (4, 1, 2, 3)$. Moreover, the example in (14) relates to r, *i.e.*, we have $\tau_r = (2, 1)$ and $\sigma_r = (1, -1)$. A search for cycles according to (15)–(17) results in the following equations:

$$\begin{aligned} w_{11} &= w_{24} = -w_{13} = -w_{22} \\ w_{12} &= w_{21} = -w_{14} = -w_{23} \end{aligned}$$

Repeating the same with $\pi_S = (2, 1, 4, 3)$, $\tau_s = (1, 2)$ and $\sigma_s = (-1, 1)$ results in

$$\begin{array}{ll} w_{11} = -w_{12} & w_{21} = w_{22} \\ w_{13} = -w_{14} & w_{23} = w_{24} \end{array}$$

and putting both sets of equations together shows that a matrix W^L with entries

$$\begin{aligned} w_{11} = &-w_{12} = -w_{13} = w_{14} \\ = -w_{21} = &-w_{22} = \quad w_{23} = w_{24} \end{aligned}$$

will satisfy $W^L R = r W^L$ and $W^L S = s W^L$ as desired. □

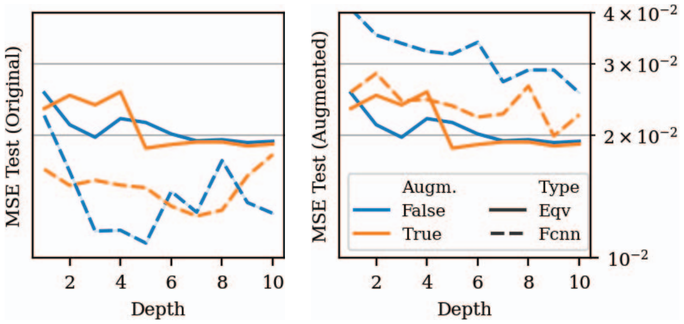
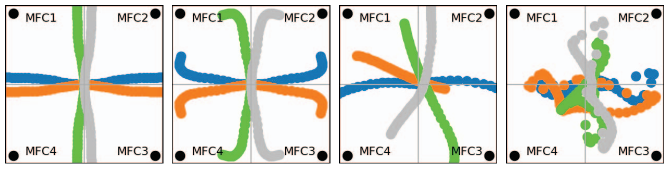


Fig. 3: Mean squared error yielded by equivariant and standard models on original (left) and augmented (right) test data. Each figure includes one plot for equivariant models trained on original data (solid, blue), one for equivariant models trained on augmented data (solid, orange), one for standard models trained on original data (dashed, blue), and one for standard models trained on augmented data (dashed, orange).



(a) EQV (b) EQV DA (c) FCNN (d) FCNN DA

Fig. 5: Comparison of model trajectories under flow shift between adjacent sensors. Blue: $x_3 = x_4 = 0.25$ fixed and x_1, x_2 vary, orange: $x_1 = x_2 = 0.25$ fixed and x_3, x_4 vary, green: $x_2 = x_3 = 0.25$ are fixed and x_1, x_4 vary, silver: $x_1 = x_4 = 0.25$ are fixed and x_2, x_3 vary.

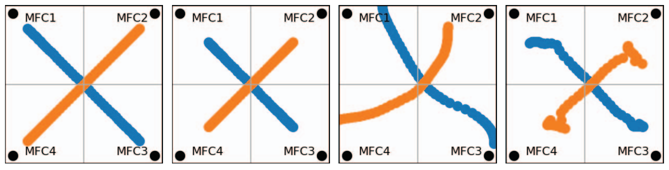
III. Numerical Experiments

In this section, we present and discuss our numerical results. First, group equivariant and standard networks are compared in terms of their performance on the leakage detection dataset (Figure 3). Second, we carry out a visual inspection of model predictions on specifically structured inputs (Figures 4–6). Code and data to reproduce our results are available under https://github.com/chrbraue/leakage_detection.

All models were trained on a GeForce GTX 1080 Ti GPU using TensorFlow 2.6.0 [22]. Throughout, we used the MSE loss, the Nadam optimizer [23] with standard parameters, batch size 32, the ELU activation function [24] in all hidden layers, linear output layers, and trained for 1000 epochs with early stopping. The width of equivariant hidden layers that are equipped with a weight matrix as in (13) is by construction restricted to four neurons. To allow for a fair comparison, we also restricted the width of standard hidden layers to four neurons. Output layers of equivariant models feature a weight matrix of type (18). In both cases, equivariant and standard, the number of hidden layers ranges between 1 and 10, learning rates 0.01, 0.001 and 0.0001 were tried, and models were trained on original as well as on augmented data. Each possible hyperparameter configuration was applied repeatedly ten times with random initialization. To assess early stopping conditions and to select best models with respect to different learning rates and random initialization, we randomly separated a 10% validation split from the training data. The evaluation in Figure 3 is in terms of previously unseen test data (see Figures 2c–2d).

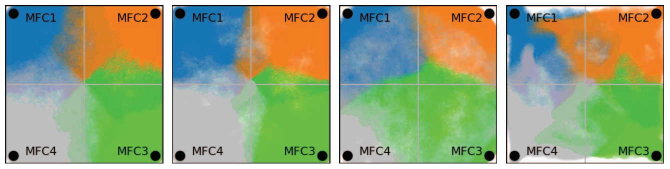
The results in Figure 3 (left) show that standard networks perform best in terms of the original (unaugmented) test data. This is arguably due to the fact that both training and test data are relatively small and subject to measurement noise, and thus do not properly reflect prior knowledge in terms of group equivariance. The picture changes when we look at Figure 3 (right) which shows that equivariant models outperform standard networks when evaluated on augmented data, regardless on which dataset they were trained. In additional experiments, we considered standard networks with an increased number of 16 neurons per hidden layer. Associated results show that the performance of standard networks without restricted width approaches that of equivariant networks, but only in case they are trained on augmented data.

Additionally, we subjected four best performing models to a visual inspection which is illustrated in Figures 4–6. The depicted models were selected as follows: For each combination in {equivariant, standard} $\times$ {trained on original data, trained on augmented data}, we selected the best set of hyperparameters with respect to the performance on the validation data, retrained each configuration in 10 trials, and selected in each case the best performing model with respect



(a) EQV (b) EQV DA (c) FCNN (d) FCNN DA

Fig. 4: Comparison of model trajectories under flow shift between diagonally opposed sensors. Blue: $x_2 = x_4 = 0.25$ are fixed and x_1, x_3 vary, orange: $x_1 = x_3 = 0.25$ are fixed and x_2, x_4 vary. EQV and FCNN refer to equivariant and standard models trained on original data. EQV DA and FCNN DA refer to models trained on augmented data.



(a) EQV (b) EQV DA (c) FCNN (d) FCNN DA

Fig. 6: Comparison of model predictions on subsets with $x_1 = \max\{x_i\}$ (blue), $x_2 = \max\{x_i\}$ (orange), $x_3 = \max\{x_i\}$ (green), $x_4 = \max\{x_i\}$ (silver).

to the validation data. Standard architectures with layer width 16, as mentioned above, were included here.

The results in Figures 4–5 show that both equivariant models exhibit the expected behavior. In Figures 4a–4b this becomes apparent because equivariance enforces diagonal trajectories when diagonally opposed inputs are fixed to the same value. Figures 5a–5b emphasize the symmetry of equivariant models with respect to coordinate axes. In contrast, Figures 4c–4d and 5c–5d show that standard networks feature neither behaviour. Moreover, standard networks do not necessarily map the input $0.25 \cdot (1, 1, 1, 1)$ to the origin, which must be the case under equivariance conditions.

Figure 6 illustrates predictions on specific subsets of the input domain, namely

$$\Delta_j := \{x \in \Delta \mid x_j = \max(x_1, x_2, x_3, x_4)\} \qquad (19)$$

for different values of j. In an ideal environment, it would be clear that the leakage position associated with a measurement $x \in \Delta_j$ must be located in the same quadrant as the related mass flow controller (MFC) j. However, this prior knowledge is not reflected in terms of equivariance conditions and thus, neither incorporated in equivariant networks nor in standard networks. Hence, the results in Figure 6 give information about the generalization capabilities of both architectures, in view of the application at hand. It can be seen that predictions of equivariant networks roughly fit the target segmentation of the output domain into four distinct quadrants, while standard networks are more clearly off target. Moreover, the deviation induced by equivariant networks appears more systematic and is likely more accessible by appropriate regularization strategies.

An overall comparison of both equivariant networks shows that the one trained on augmented data is slightly superior to its counterpart, at least from the point of view of the application. Regarding Figures 4a–4b, it is more plausible that the depicted trajectories do not approach the positions of the sensors too strong. Close proximity to sensors should only occur for $x_1, x_3 \ll 0.25$ and $x_2, x_4 \ll 0.25$, respectively. Moreover, regarding Figures 5a–5b, it is rather plausible that trajectories get closer to a sensors position when the associated flow rate is further increased. Finally, the deviation from the target segmentation in Figures 6a–6b is significantly lower in case of the equivariant network trained on augmented data.

IV. Conclusion

In this paper, we presented a fully connected neural network architecture that features equivariance with respect to the dihedral group of order eight. Our experiments and visual inspection suggest that the combination of incorporated domain knowledge and training on accordingly augmented data yields the most promising results in a small data regime. The proposed method for the construction of equivariant layers is likely transferable to symmetry groups of different order. Here, we concentrated on equivariant models with a fixed hidden layer width of four. More flexible approaches allowing for wider equivariant architectures, regularization approaches to incorporate additional prior knowledge, as well as a generalization to additional input features and multiple leakages will be subject of future work.

References

[1] F. Campbell Jr, *Manufacturing processes for advanced composites*. Elsevier, 2003.

[2] G. Fernlund, J. Wells, L. Fahrang, J. Kay, and A. Poursartip, "Causes and remedies for porosity in composite manufacturing," in *IOP conference series: materials science and engineering*, vol. 139, no. 1. IOP Publishing, 2016, p. 012002.

[3] A. Haschenburger and C. Heim, "Two-stage leak detection in vacuum bags for the production of fibre-reinforced composite components," *CEAS Aeronautical Journal*, vol. 10, no. 3, pp. 885–892, 2019.

[4] A. Haschenburger, N. Menke, and J. Stüve, "Sensor-based leakage detection in vacuum bagging," *The International Journal of Advanced Manufacturing Technology*, vol. 116, no. 7, pp. 2413–2424, 2021.

[5] L. von Rueden *et al.*, "Informed machine learning–a taxonomy and survey of integrating knowledge into learning systems," *arXiv preprint arXiv:1903.12394*, 2019.

[6] P. Niyogi, F. Girosi, and T. Poggio, "Incorporating prior information in machine learning by creating virtual examples," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2196–2209, 1998.

[7] J. Wu, H. Xiao, and E. Paterson, "Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework," *Physical Review Fluids*, vol. 3, no. 7, p. 074602, 2018.

[8] D. Bergman, "Symmetry constrained machine learning," in *Proceedings of SAI Intelligent Systems Conference*. Springer, 2019, pp. 501–512.

[9] B. Schölkopf, P. Simard, A. J. Smola, and V. Vapnik, "Prior knowledge in support vector kernels," *Advances in neural information processing systems*, pp. 640–646, 1998.

[10] J. Ling, A. Kurzawski, and J. Templeton, "Reynolds averaged turbulence modelling using deep neural networks with embedded invariance," *Journal of Fluid Mechanics*, vol. 807, pp. 155–166, 2016.

[11] A. Butter, G. Kasieczka, T. Plehn, and M. Russell, "Deep-learned top tagging with a Lorentz layer," *SciPost Phys*, vol. 5, no. 3, p. 028, 2018.

[12] L. Atlas, T. Homma, and R. Marks, "An artificial neural network for spatio-temporal bipolar patterns: Application to phoneme classification," in *Neural Information Processing Systems*, 1987, pp. 31–40.

[13] T. Waibel, A.and Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme recognition using time-delay neural networks," *IEEE transactions on acoustics, speech, and signal processing*, vol. 37, no. 3, pp. 328–339, 1989.

[14] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[15] J. Masci, E. Rodolà, D. Boscaini, M. Bronstein, and H. Li, "Geometric deep learning," in *SIGGRAPH ASIA 2016 Courses*. ACM, 2016, pp. 1–50.

[16] M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

[17] T. Cohen and M. Welling, "Group equivariant convolutional networks," in *International conference on machine learning*. PMLR, 2016, pp. 2990–2999.

[18] S. Dieleman, J. De Fauw, and K. Kavukcuoglu, "Exploiting cyclic symmetry in convolutional neural networks," in *International conference on machine learning*. PMLR, 2016, pp. 1889–1898.

[19] D. Worrall, S. Garbin, D. Turmukhambetov, and G. Brostow, "Harmonic networks: Deep translation and rotation equivariance," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5028–5037.

[20] J. Li, Z. Yang, H. Liu, and D. Cai, "Deep rotation equivariant network," *Neurocomputing*, vol. 290, pp. 26–33, 2018.

[21] Y. Shapira, *Linear Algebra and Group Theory for Physicists and Engineers*. Springer, 2019.

[22] M. Abadi *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[23] T. Dozat, "Incorporating Nesterov momentum into Adam," Workshop ICLR, 2016.

[24] D. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.