Remove Symmetries to Control Model Expressivity and Improve Optimization

Liu Ziyin[1,2,*], Yizhou Xu[3,*], Isaac Chuang[1,4]

[1]*Research Laboratory of Electronics, Massachusetts Institute of Technology*
[2]*Physics & Informatics Laboratories, NTT Research*
[3]*School of Computer and Communication Sciences, École Polytechnique Fédérale de Lausanne*
[4]*Department of Physics, Massachusetts Institute of Technology*

Abstract

When symmetry is present in the loss function, the model is likely to be trapped in a low-capacity state that is sometimes known as a "collapse." Being trapped in these low-capacity states can be a major obstacle to training across many scenarios where deep learning technology is applied. We first prove two concrete mechanisms through which symmetries lead to reduced capacities and ignored features during training. We then propose a simple and theoretically justified algorithm, *syre*, to remove almost all symmetry-induced low-capacity states in neural networks. The proposed method is shown to improve the training of neural networks in scenarios when this type of entrapment is especially a concern. A remarkable merit of the proposed method is that it is model-agnostic and does not require any knowledge of the symmetry.

1. Introduction

Recent literature has shown that symmetries in the loss function of neural networks often lead to the formation of low-capacity saddle points within the loss landscape [14, 23]. These saddle points are located at the symmetric solutions and often possess a lower capacity than the minimizers of the loss. When a model encounters these saddle points during training, the model parameters are not only slow to escape them but also attracted to these solutions because these the gradient noise also vanish close to these saddles [5]. Essentially, the model's learning process stagnates, and it fails to achieve optimal performance due to reduced capacity. However, while many works have characterized the dynamical properties of training algorithms close to symmetric solutions, no methods are known to enable full escape from them.

Because these low-capacity saddles are created by symmetries, we propose a method to explicitly remove these symmetries from the loss functions of neural networks. The method we propose is theoretically justified and only takes one line of code to implement. By removing these symmetries, our method allows neural networks to explore a more diverse set of parameter spaces and access more expressive solutions. The main contributions of this work are:

1. We show how discrete symmetries in the model can severely limit the expressivity of neural networks in the form commonly known as "collapses" (Section 3);
2. We propose a simple method that provably removes almost all symmetries in neural networks, without having any knowledge of the symmetry (Section 4);

[*]Equal contribution.

We introduce the notations and problem setting for our work in the next section. Closely related works are discussed in Appendix A. All proofs and experimental details are deferred to the appendix.

2. Discrete Symmetries in Neural Networks

Notation. For a matrix A, we use A^+ to denote the pseudo-inverse of A. For groups U and G, $U \triangleleft G$ denotes that U is a subgroup of G. For a vector w and matrix D, $\|\theta\|_D^2 := w^T D w$ is the norm of w with respect to D. $\odot$ denotes the element-wise product between vectors.

Let $f(\theta, x)$ be a function of the model parameters θ and input data point x. For example, f could either be a sample-wise loss function ℓ or the model itself. Whenever f satisfies the following condition, we say that f has the P-reflection symmetry (general symmetry groups are dealt with in Theorem 6 in Section 4).

Definition 1 *Let P be a projection matrix and θ' be a point. f is said to have the (θ', P)-reflection symmetry if for all x and θ, (1) $f(\theta + \theta', x) = f((I - 2P)\theta + \theta', x)$, and (2) $P\theta' = \theta'$.*

The second condition is due to the fact that there is a redundancy in the choice of θ' when $\theta' \neq 0$. Requiring $P\theta' = \theta'$ removes this redundancy and makes the choice of θ' unique. Since every projection matrix can be written as a product of a (full-rank or low-rank) matrix O with orthonormal columns, one can write $P = OO^T$ and refer to this symmetry as an O symmetry. In common deep learning scenarios, it is almost always the case that $\theta' = 0$ (for example, this holds for the common cases of rescaling symmetries, (double) rotation symmetries, and permutation symmetries,[1] see Theorem 2-4 of Ref. [35]). A consequence of $\theta' = 0$ is that the symmetric projection $P\theta$ of any θ always has a smaller norm than θ: thus, a symmetric solution is coupled to the solutions of weight decay, which also favors small-norm solutions. As an example of reflection symmetry, consider a simple tanh network $f(\theta, x) = \theta_1 \tanh(\theta_2 x)$. The model output is invariant to a simultaneous sign flip of θ_1 and θ_2. This corresponds to a reflection symmetry whose projection matrix is the identity $P = ((1,0),(0,1))$. The symmetric solutions correspond to the trivial state where $\theta_1 = \theta_2 = 0$.

3. Symmetry Impairs Model Capacity and Optimization

We first show that reflection symmetry directly affects the model capacity. For simplicity, we let $\theta' = 0$ for all symmetries. Let $f(x, \theta) \in \mathbb{R}$ be a Taylor-expandable model that contains a P-reflection symmetry. Let $\Delta = \theta - \theta_0$. Then, close to any symmetric point θ_0 (any θ_0 for which $P\theta_0 = 0$), for all x, Ref. [35] showed that

$$f(x,\theta) - f(x,\theta_0) = \underbrace{\nabla_\theta f(x,\theta_0)(I-P)\Delta + O(\|P\Delta\|)^2}_{\text{Symmetry subspace}} + \underbrace{\frac{1}{2}\Delta^T P H(x) P\Delta + O(\|\Delta\|^3)}_{\text{Symmetry-broken subspace}}, \quad (1)$$

where $H(x)$ is the Hessian matrix of f. An important feature is that the symmetry subspace is a generic expansion where both odd and even terms are present, and the first order term does not vanish in general. In contrast, in the symmetry-broken subspace, all odd-order terms in the expansion vanish, and the leading order term is the second order. This implies that close to a symmetric solution, escaping from it will be slow, and if at the symmetric solution, it is impossible for gradient descent to leave it. The effect of this entrapment can be quantified by the following two propositions.

[1] All appear in deep learning [9, 12, 15, 27, 31, 32].

Proposition 2 *(Symmetry removes feature.) Let f have the P-symmetry, and θ be intialized at θ_0 such that $P\theta_0 = 0$. Then, the kernalized model, $g(x,\theta) = \lim_{\lambda\to 0}(\lambda^{-1}f(x,\lambda\theta+\theta_0) - f(x,\theta_0))$, converges to $\theta^* = A^+\sum_x \nabla_\theta f(x,\theta_0)^T y(x)$ under GD for a sufficiently small learning rate. Here $A := (I-P)\sum_x \nabla_\theta f(x,\theta_0)^T \nabla_\theta f(x,\theta_0)(I-P)$ and A^+ denotes the Moore–Penrose inverse of A.*

This means that in the kernel regime[2], being at a symmetric solution implies that the feature kernel features are being masked by the projection matrix $\nabla_\theta f(x,\theta_0) \to (I-P)\nabla_\theta f(x,\theta_0)$, and learning can only happen given these masks. This implies that the model is not using the full feature space that is available to it.

Proposition 3 *(Symmetry reduces parameter dimension.) Let f have the P-symmetry, and $\theta \in \mathbb{R}^d$ be intialized at θ_0 such that $P\theta_0 = 0$. Then, for all time steps t under GD or SGD, there exists a model $f'(x,\theta')$ and sequence of parameters θ'_t such that for all x, $f'(x,\theta'_t) = f(x,\theta_t)$, where $\dim(\theta') = d - \mathrm{rank}(P)$.*

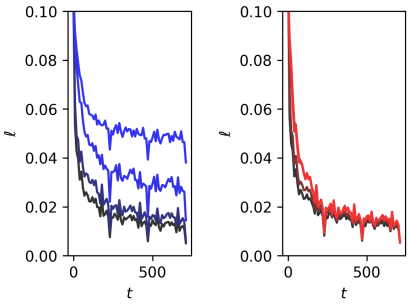

Figure 1: Training loss ℓ as a function of the iteration t for a fully connected network on MNIST. Starting from a low-capacity state, vanilla neural networks are trapped under SGD or Adam training (**left**). Blacker lines correspond to higher-capacity initializations, where more neurons are away from the permutation-symmetric state. When the symmetries are removed, the capacity of the initialization no longer affects the solution found at the end of the training (**right**).

The existence of this type of dimension reduction when symmetry is present has also been noticed by previous works in case of permutation symmetry [31]. See Figure 1 for an illustration. We initialize a two-layer ReLU neural network on a low-capacity state where a fraction of the hidden neurons are identical (corresponding to the symmetric states of the permutation symmetry) and train with and without removing the symmetries. We see that when the symmetries are removed (with the method proposed in the next section), the model is no longer stuck at these neuron-collapsed solutions.

4. Removing Symmetry with Static Bias

Next, we prove that a simple algorithm that involves almost no modification to any deep learning training pipeline can remove almost all such symmetries from the loss without creating new ones. From this section onward, we will consider the case where the function under consideration is the loss function (a per-batch loss or its expectation): $f = \ell$.

Countable Symmetries. We seek an algorithm that eliminates the reflection symmetries from the loss function ℓ. We show that when the number of reflection symmetries in the loss function is finite, one can completely remove them using a simple technique. The symmetries are required to have the following property and the loss function is assumed to obey assumption 1.

Property 1 *(Enumeratability) There exists a countable set of pairs of projection matrices and biases $S = \{(\theta_i^\dagger, P_i)\}_i^N$ such that $\ell(\theta,x)$ has the $\theta_i^\dagger$-centric P_i-reflection symmetry for all i. In addition, ℓ does not have any $(\theta^\dagger, P)$ symmetry for $(\theta,P) \notin S$.*

Assumption 1 *There only exists countably many pairs $(c_0,\tilde{\theta})$ such that $g(x) = \ell(\theta,x) - c_0\theta$ contains a $\tilde{\theta}$-centric P symmetry, where we require $Pc_0 = c_0$ and $P\tilde{\theta} = \tilde{\theta}$.*

[2] Technically, this is the lazy training limit [7].

This assumption is satisfied by common neural networks with standard activations. The main purpose of this assumption is to rule out the pathological of a linear or quadratic deterministic objective, which never appears in practice or for which symmetry is not a concern.[3]

For symmetry removal, we propose to utilize the following alternative loss function. Let θ_0 be drawn from a Gaussian distribution with variance σ_0 and ℓ be the original loss function:

$$\ell_{\mathrm{r}}(\theta, x) = \ell(\theta + \theta_0) + \gamma\|\theta\|^2. \tag{2}$$

γ is nothing but the standard weight decay. We will see that using a static bias along with weight decay is essential for the method to work. We find that with unit probability, the loss function ℓ_{r} contains no reflection symmetry:

Theorem 4 *Let ℓ satisfy Property 1 and Assumption 1. Then, with probability 1 (over the sampling of θ_0), there exists no projection matrix P and reflection point θ' such that ℓ_{r} has the (θ', P)-symmetry.*

This hints at the possibility of using a small and random θ_0, which removes all symmetries in principle and also does not essentially affect the solutions of the original objective. In this work, we will refer to the method in Eq. (2) as *syre*, an abbreviation for "symmetry removal."

Uncountably Many Symmetries. In neural networks, it is possible for the model to simultaneously contain infinitely many reflection symmetries. This happens, for example, when the model parameters have the rotation symmetry or the double rotation symmetry (common in self-supervised learning problems or transformers). It turns out that simply adding a static bias and weight decay is not sufficient to remove all symmetries.

We propose to train on the following alternative loss, where ar stands for "advanced removal":

$$\ell_{\mathrm{ar}}(\theta) = \ell(\theta + \theta_0) + \gamma\|\theta\|_D^2, \tag{3}$$

where D is a positive diagonal matrix in which all diagonal elements of D are different. The simplest way to achieve such a D is to set $D_{ii} \sim \mathrm{Uniform}(1-\epsilon, 1+\epsilon)$, where ϵ is a small quantity.

Theorem 5 *Any (θ', P)-symmetry that ℓ_{ar} satisfies obeys: (1) $P\theta_0 = \theta'$ (2) and $PD = DP$.*

Conditions (1) and (2) implies that there are at most finitely many (θ', P)-symmetry ℓ_{ar} can have. When there does not exist any symmetry that satisfies this condition, we have removed all the symmetries. In the worst case where ℓ is a constant function, there are 2^N symmetries where N is the number of reflection symmetries. If we further assume that every P is associated with at most finitely many θ', then we, again, remove all symmetries with probability 1. The easiest way to determine this D matrix is through sampling from a uniform distribution with a variance $\sigma_D \ll 1$.

Symmetry Removal for General Groups. Lastly, one can generalize the theory to prove that the proposed method removes symmetries from a generic group. Let G be the linear representation of a generic finite group, possibly with many nontrivial subgroups. If the loss function ℓ is invariant under transformation by the group G, then $\forall g,\ \ell(\theta) = \ell(g\theta)$. Because G is finite, it follows that the representations g must be full-rank and unipotent. Letting U be a subgroup of G, we denote with the overbar the following matrix: $\overline{U} = \frac{1}{|U|}\sum_{u\in U} u$. Note that $\overline{U}$ is a projection matrix: $\overline{U}\,\overline{U} = \overline{U}$. This

[3] An example that violates this assumption is when $\ell(\theta, x) = c_0^T\theta$. $\ell(\theta, x) - c_0^T\theta$ has infinitely many reflection symmetries everywhere and for every data point because it is a constant function.

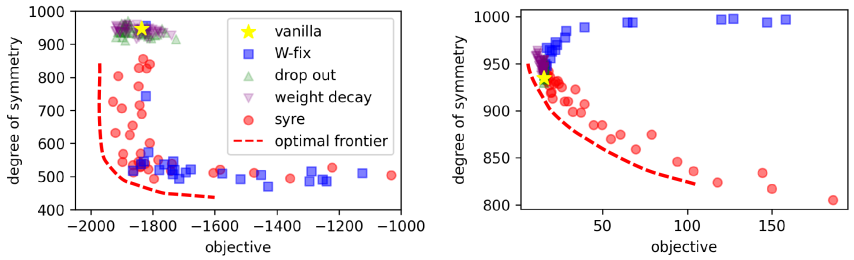

Figure 2: The degree of symmetry versus the objective value for two choices of B and various training methods with different hyperparameters. The proposed method is the only method to smoothly interpolate between optimized solutions and solutions with low symmetry. *syre* performs well in both cases. **Left**: objective with unstructured symmetry **Right**: structured symmetry.

means that $I - \overline{U}$ is also a projection matrix. Importantly, $\overline{U}$ projects a vector into the symmetric subspace. For any $u \in U$, $u\overline{U} = \overline{U}$. Likewise, $I - \overline{U}$ projects any vector into the symmetry-broken subspace, a well-known result in the theory of finite groups [17]. We denote, by $\Delta_V = \|(I-\overline{V})\nabla_\theta \ell\|$, the strength of the symmetry removal for the subgroup V.

Theorem 6 *Let $\Gamma(G)$ denote the smallest minimal generating set for the group G. Z denotes the number of minimal subgroups of G. Let ℓ be invariant under the group transformation G and let θ be in the invariant subspace of a subgroup $U \triangleleft G$. Then, for every subgroup $V \triangleleft U \triangleleft G$,*

1. $\Delta_V = \Omega(\gamma\sigma_0 \mathrm{rank}(I - \overline{V}))$;
2. $\min_{V \triangleleft U} \Delta_V = \Omega(\gamma\sigma_0 \mathrm{rank}(I - \overline{V})Z^{-1})$;
3. *if G is abelian,* $\min_{V \triangleleft U} \Delta_V = \Omega(\gamma\sigma_0 \mathrm{rank}(I - \overline{V})|\Gamma(U)|^{-1})$;[4]
4. *additionally, for any $\epsilon > 0$ and $\delta < 1$,* $\Pr(\min_{V \triangleleft U} \Delta_V > \epsilon) > \delta$, *if* $\gamma\sigma_0 = \Omega\left(\frac{2\epsilon|\Gamma(U)|}{1-\delta}\right)$.

Item (2) is due to the fact that removing symmetries from a larger group can be reduced to removing them from one of its subgroups. In general, $1 \leq Z \leq |G|$ (and sometimes $\ll |G|$), and so this scaling is not bad. For the part (3) of the theorem, the term $|\Gamma(U)|$ is especially meaningful. It is well-known that $|\Gamma(U)| \leq \log |U|$, and so the worst-case symmetry-breaking strength is only of order $1/\log|U|$, which is far slower than what one would expect. In fact, for a finite group with size N, the number of subgroups can grow as fast as $N^{\log N}$ [3], and thus, one might naively think that the minimal breaking strength decreases as $N^{-\log N}$. This theorem shows that the proposed method is highly effective at breaking the symmetries in the loss function or the model.

5. Benchmarking Symmetry Removal

In this section, we benchmark the effect of symmetry control of the proposed method for two controlled experiments. To compare the influence of *syre* and other training methods on the degree of symmetry, we consider minimizing the following objective function: $(w^T w)^2 - w^T B w := (w^T w)^2 - \sum_{i=1}^d \lambda_i (v_i^T w)^2$,. where $w \in \mathbb{R}^d$ is the optimization parameter and $B \in \mathbb{R}^{d\times d}$ is a given symmetric matrix with eigenvalues λ_i and eigenvectors v_i ($v_i^T v_i = 1$). The objective function has n reflection symmetries $P_i w := w - 2(v_i^T w)v_i$. Hence, we define the degree of symmetry as $\sum_{i=1}^d \mathbf{1}\{v_i^T w < c_{\mathrm{th}}\}$, where c_{th} is a given threshold. Depending on the spectrum of B, the nature

[4] This result can be generalized to the case where all projectors $\bar{V}$ commute with each other, even if G is nonabelian. Namely, this is a consequence of the properties of the representations of G and of G itself.

of the task is different. We thus consider two types of spectra: (1) an unstructured spectrum where $B = G + G^T$ for a Gaussian matrix G, and (2) a structured spectrum where $B = \mathrm{diag}(v)$ where v is a random Gaussian vector. Conceptually, the first type is more similar to rotation and double rotation symmetries in neural networks where the basis can be arbitrary, while the second is a good model for common discrete symmetries where the basis is often diagonal or sparse. For the first case we choose $c_{\rm th} = 10^{-3}$ and for the second case we choose $c_{\rm th} = 10^{-1}$.

In Figure 2, we compare *syre*, W-fix, drop out, weight decay, and the standard training methods in this setting for $d = 1000$ and two choices of B. In both cases, we use Gaussian initialization and gradient descent with a learning rate of 10^{-4}. For *syre* and weight decay, we choose weight decay from 0.1 to 10. For W-fix, we choose ϕ from 0.001 to 0.1. For dropout, we choose a dropout rate from 0.01 to 0.6. Figure 2 shows that for both cases, *syre* is the only method that effectively and smoothly interpolates between solutions with low symmetry and best optimization. This is a strong piece of evidence that the proposed method can *control* the degree of symmetries in the model.

Also, see Appendix C for the application of *syre* on various realistic problems of deep non-linear networks, where we apply *syre* to alleviate and remove feature and neural collapse, low-capacity trap in self-supervised learning and loss of plasticity in supervised and reinforcement learning.

6. Conclusion

We have shown that the existence of symmetries in the model or loss function may severely limit the expressivity of the trained model. We then developed a theory that leverages the power of representation theory to show that adding random static biases to the model, along with weight decay, is sufficient to remove almost all symmetries, explicit or hidden. We have demonstrated the relevance of the method to a broad range of applications in deep learning, and a possible future direction is to deploy the method in large language models, which naturally contain many symmetries.

Acknowledgements

The authors acknowledge support from NTT Research, and from the Institute for Artificial Intelligence and Fundamental Interactions (IAIFI) through NSF Grant No. PHY-2019786.

References

[1] Zaheer Abbas, Rosie Zhao, Joseph Modayil, Adam White, and Marlos C Machado. Loss of plasticity in continual deep reinforcement learning. *arXiv preprint arXiv:2303.07507*, 2023.

[2] Jordan Ash and Ryan P Adams. On warm-starting neural network training. *Advances in neural information processing systems*, 33:3884–3894, 2020.

[3] Alexandre Borovik, Laszlo Pyber, and Aner Shalev. Maximal subgroups in finite and profinite groups. *Transactions of the American Mathematical Society*, 348(9):3745–3761, 1996.

[4] Marco Cariglia. Hidden symmetries of dynamics in classical and quantum physics. *Reviews of Modern Physics*, 86(4):1283, 2014.

[5] Feng Chen, Daniel Kunin, Atsushi Yamamura, and Surya Ganguli. Stochastic collapse: How gradient noise attracts sgd dynamics towards simpler subnetworks. *arXiv preprint arXiv:2306.04251*, 2023.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[7] Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. *arXiv preprint arXiv:1812.07956*, 2018.

[8] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning, 2016.

[9] L. Dinh, R. Pascanu, S. Bengio, and Y. Bengio. Sharp Minima Can Generalize For Deep Nets. *ArXiv e-prints*, March 2017.

[10] Shibhansh Dohare, Richard S Sutton, and A Rupam Mahmood. Continual backprop: Stochastic gradient descent with persistent randomness. *arXiv preprint arXiv:2108.06325*, 2021.

[11] Shibhansh Dohare, J Fernando Hernandez-Garcia, Parash Rahman, Richard S Sutton, and A Rupam Mahmood. Maintaining plasticity in deep continual learning. *arXiv preprint arXiv:2306.13812*, 2023.

[12] Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. The role of permutation invariance in linear mode connectivity of neural networks. *arXiv preprint arXiv:2110.06296*, 2021.

[13] Valeri P Frolov, Pavel Krtouš, and David Kubizňák. Black holes, hidden symmetries, and complete integrability. *Living reviews in relativity*, 20:1–221, 2017.

[14] Kenji Fukumizu. A regularity condition of the information matrix of a multilayer perceptron network. *Neural networks*, 9(5):871–879, 1996.

[15] Charles Godfrey, Davis Brown, Tegan Emerson, and Henry Kvinge. On the symmetries of deep learning models and their internal representations. *Advances in Neural Information Processing Systems*, 35:11893–11905, 2022.

[16] Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.

[17] Daniel Gorenstein. *Finite groups*, volume 301. American Mathematical Soc., 2007.

[18] Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *arXiv preprint arXiv:1903.08560*, 2019.

[19] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016.

[20] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[21] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[22] Anders Krogh and John A Hertz. A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957, 1992.

[23] Xingguo Li, Junwei Lu, Raman Arora, Jarvis Haupt, Han Liu, Zhaoran Wang, and Tuo Zhao. Symmetry, saddle points, and global optimization landscape of nonconvex matrix factorization. *IEEE Transactions on Information Theory*, 65(6):3489–3514, 2019.

[24] Derek Lim, Moe Putterman, Robin Walters, Haggai Maron, and Stefanie Jegelka. The empirical impact of neural parameter symmetries, or lack thereof. *arXiv preprint arXiv:2405.20231*, 2024.

[25] James Lucas, George Tucker, Roger Grosse, and Mohammad Norouzi. Don't blame the elbo! a linear vae perspective on posterior collapse, 2019.

[26] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[27] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. *arXiv preprint arXiv:1412.6614*, 2014.

[28] Massimiliano Patacchiola and Amos J Storkey. Self-supervised relational reasoning for representation learning. *Advances in Neural Information Processing Systems*, 33:4003–4014, 2020.

[29] Clarice Poon and Gabriel Peyré. Smooth bilevel programming for sparse regularization. *Advances in Neural Information Processing Systems*, 34:1543–1555, 2021.

[30] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[31] Berfin Simsek, François Ged, Arthur Jacot, Francesco Spadaro, Clément Hongler, Wulfram Gerstner, and Johanni Brea. Geometry of the loss landscape in overparameterized neural networks: Symmetries and invariances. In *International Conference on Machine Learning*, pages 9722–9732. PMLR, 2021.

[32] Ryan J Tibshirani. Equivalences between sparse models and neural networks. *Working Notes. URL https://www. stat. cmu. edu/˜ ryantibs/papers/sparsitynn. pdf*, 2021.

[33] Yixin Wang, David Blei, and John P Cunningham. Posterior collapse and latent variable nonidentifiability. *Advances in neural information processing systems*, 34:5443–5455, 2021.

[34] Zihao Wang and Liu Ziyin. Posterior collapse of a linear latent variable model. *Advances in Neural Information Processing Systems*, 35:37537–37548, 2022.

[35] Liu Ziyin. Symmetry induces structure and constraint of learning. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=7AF0AMI4AE.

[36] Liu Ziyin and Zihao Wang. spred: Solving L1 Penalty with SGD. In *International Conference on Machine Learning*, 2023.

Appendix A. Related Works

One closely related work is that of Ref. [35], which shows that every reflection symmetry in the model leads to a low-capacity solution that is favored when weight decay is used. This is because the minimizer of the weight decay is coupled with stationary points of the reflection symmetries – the projection of any parameter to a symmetric subspace always decreases the norm of the parameter, and is thus energetically preferred by weight decay. Our work develops a method to decouple symmetries and weight decay, thus avoiding collapsing into low-capacity states during training. Besides weight decay, an alternative mechanism for this type of capacity loss is gradient-noise induced collapse, which happens when the learning rate - batchsize ratio is high [5].

Contemporarily, Ref. [24] empirically explores how removing symmetries can benefit neural network training and suggests a heuristic for removing symmetries by hold a fraction of the weights unchanged during training. However, the proposed method is only proved to work for explicit permutation symmetries in fully connected layers. This is particularly a problem because most of the symmetries in nonlinear systems are unknown and hidden [4, 13]. In sharp contrast, the technique proposed in our work is both architecture-independent and symmetry-agnostic and provably removes all known and unknown P-reflection symmetries in the loss.

Appendix B. Theoretical Concerns

B.1. Proof of Proposition 2

Proof By (1), $g(x,\theta)$ simplifies to a kernel model

$$g(x,\theta) = \nabla_{\theta_0} f(x,\theta_0)(I-P)\theta. \tag{4}$$

Let us consider the squared loss $\ell(\theta) = \sum_x ||y(x) - g(x,\theta)||^2$ and denote

$$A \coloneqq \sum_x (I-P)\nabla_{\theta_0} f(x,\theta_0)^T \nabla_{\theta_0} f(x,\theta_0)(I-P),\ b \coloneqq (I-P)\sum_x \nabla_{\theta_0} f(x,\theta_0)^T y(x).$$

Assuming the learning rate to be η, the GD reads

$$\theta^{t+1} = \theta^t - 2\eta(A\theta^t - b), \tag{5}$$

where $\theta^0 = 0$. If

$$\eta < \frac{1}{2\lambda_{max}(A)}, \tag{6}$$

GD converges to

$$\begin{aligned} \theta^* &= \lim_{t\to\infty} \sum_{k=0}^{t} (I - 2\eta A)^k * 2\eta b \\ &= A^+ b, \end{aligned} \tag{7}$$

which is the well-known least square solution. ■

B.2. Proof of Proposition 3

Proof According to [35, Theorem 4], we have

$$P\nabla_\theta \ell(x, \theta_0) = 0. \tag{8}$$

Therefore, after one step of GD or SGD, we still have $P\theta_1 = 0$. By induction, we have $P\theta_t = 0$.

Finally, suppose that $\{a_i\}_{i=1}^{d-\mathrm{rank}(P)}$ forms a basis of $\ker P$, and define $f'(x, \theta') := f(x, \sum_{i=1}^{d-\mathrm{rank}(P)} \theta'_i a_i)$ for $\dim(\theta') = d - \mathrm{rank}(P)$. By choosing $\theta'_i = \theta^T a_i$, we have $f'(x, \theta'_t) = f(x, \theta_t)$. ■

B.3. Lemmas

Lemma 7 *Let $x \in \mathbb{R}^d$ and P be a projection matrix. Let $f(x)$ be a scalar function that satisfies*

$$f(x + x') = f((I - 2P)x + x') + c^T P x, \tag{9}$$

where c is a constant vector. Then, there exists a unique function $g(x)$ such that

1. *$g(x)$ has the x'-centric P-symmetry,*
2. *and $f(x) = g(x) + \frac{1}{2} c_0^T P x$.*

Proof (a) Existence. $f(x) = g(x) + \frac{1}{2} c_0^T x$. Let us suppose $g(x)$ is not x'-centric P-symmetry. Then, there exists x such that

$$g(x + x') - g((I - 2P)x + x') = \Delta \neq 0. \tag{10}$$

Then, by definition, we have that

$$\begin{aligned} c_0^T P x &= f(x + x') + f((I - 2P)x + x') && (11)\\ &= g(x + x') - \frac{1}{2} c_0^T P(x + x') - g((I - 2P)x + x') - c_0^T P(x + x') && (12)\\ &= \Delta + \frac{1}{2} c_0^T P(x + x') - \frac{1}{2} c_0^T P((I - 2P)x + x') && (13)\\ &= \Delta + c_0^T P x. && (14) \end{aligned}$$

This is a contradiction. Therefore, there must exist $g(x)$ that satisfies the lemma statement.

(b) Uniqueness. Simply note that the expression of g is uniquely given by[5]

$$g(x) = f(x + x') - f((I - 2P)x + x'). \tag{15}$$

■

[5] Alternatively, note that $c^T x$ is odd and that $g(x)$ can be shifted by a constant to be an even function. The uniqueness follows directly from the fact that every function can be uniquely factorized into an odd function and an even function.

B.4. Proof of Theorem 4

Proof We prove by contradiction. Let us suppose there exists such pair, (θ', P). By definition, we have that

$$\ell_{\mathrm{r}}(\theta + \theta') = \ell(\theta + \theta' + \theta_0) + \gamma\|\theta + \theta'\|^2. \tag{16}$$

By assumption, we have that

$$\ell((I - 2P)\theta + \theta' + \theta_0) + \gamma\|(I - 2P)\theta + \theta'\|^2 = \ell(\theta + \theta' + \theta_0) + \gamma\|\theta + \theta'\|^2, \tag{17}$$

and, so, for all θ,

$$\ell((I - 2P)\theta + \theta' + \theta_0) = \ell(\theta + \theta' + \theta_0) + 4\gamma\theta^T P\theta'. \tag{18}$$

There are two cases: (1) $P\theta' = 0$ and (2) $P\theta' \neq 0$.

For case (1), we have that $\ell((I - 2P)\theta + \theta' + \theta_0) = \ell(\theta + \theta' + \theta_0)$, but this can only happen if the original loss ℓ has the $(\theta' + \theta_0)$-centric P-symmetry. By Property 1, this implies that

$$\theta' + \theta_0 = \theta_i^\dagger \tag{19}$$

for some i. Applying P on both sides, we obtain that

$$P\theta_0 = P\theta_i^\dagger. \tag{20}$$

But, θ_0 is a random variable with a full-rank covariance while the set $\{P\theta_i^\dagger\}$ has measure zero in the real space, and so this equality holds with probability zero.

For case (2), it follows from Lemma 7 that for a fixed x and θ_0, $\ell(\theta)$ can be uniquely decomposed in the following form

$$\ell(\theta) = g(\theta) - 2\gamma\theta^T P\theta', \tag{21}$$

where $g(\theta)$ has the $\theta' + \theta_0$-centric P-symmetry.

Let $c_0 = 2\gamma P\theta'$ and $\tilde{\theta} = P(\theta' + \theta_0)$. Then $\ell(\theta) + c_0\theta$ has the $\tilde{\theta}$-centric P symmetry. We also have $\tilde{\theta} - \frac{c_0}{2\gamma} = P\theta_0$. According to Assumption 1, there are only countable many such $\{c_0, \tilde{\theta}\}$ pairs. However, $P\theta_0$ is a standard Gaussian random variable, which leads to a contradiction with probability 1. ■

Remark 8 *It is easy to see that Assumption 1 could be slightly relaxed. We only require* $\{\tilde{\theta} - \frac{c_0}{2\gamma}\}$ *to have a zero measure, which is also a necessary and sufficient condition.*

B.5. Proof of Theorem 5

Proof We prove by contradiction. Let us suppose there exists such pair, (θ', P). By definition, we have that

$$\ell_{\mathrm{ar}}(\theta + \theta') = \ell(\theta + \theta' + \theta_0) + \gamma\|\theta + \theta'\|_D^2. \tag{22}$$

By assumption, we have that

$$\ell((I - 2P)\theta + \theta' + \theta_0) + \gamma\|(I - 2P)\theta + \theta'\|_D^2 = \ell(\theta + \theta' + \theta_0) + \gamma\|\theta + \theta'\|_D^2, \tag{23}$$

and, so, for all θ,

$$\ell((I - 2P)\theta + \theta' + \theta_0) = \ell(\theta + \theta' + \theta_0) + 4\theta^T PD((I - P)\theta + \theta'). \tag{24}$$

There are two cases: (1) $PD((I-P)\theta+\theta')=0$ and (2) $PD((I-P)\theta+\theta')\neq 0$.

Like before, there are two cases. For case (2), the proof is identical, and so we omit it. For case (1), it must be the case that for some P, and θ'

$$\ell((I-2P)\theta+\theta'+\theta_0)=\ell(\theta+\theta'+\theta_0). \tag{25}$$

This is possible if and only if $P(\theta'+\theta_0)=\theta_i^\dagger$ for some i and $P=P_i$ for the corresponding projection matrix. However, because $P\theta'=0$, this requires that

$$P(\theta'+\theta_0)=\theta_i^\dagger. \tag{26}$$

By the definition of the reflection symmetry, we have that

$$P\theta'=\theta'. \tag{27}$$

This means that

$$\theta'=\theta_i^\dagger-P\theta_0. \tag{28}$$

At the same time, we have

$$PD((I-P)\theta+\theta')=0, \tag{29}$$

which implies that

$$PD(I-P)\theta=-PD\theta'. \tag{30}$$

Because the right hand side is a constant that only depends on θ. This can only happen if both sides are zero, which is achieved if:

$$PD(I-P)=0, \tag{31}$$

and

$$\theta_i^\dagger=P\theta_0. \tag{32}$$

The first condition implies that

$$PD=PDP=D^TP^T=DP, \tag{33}$$

which implies that P and D must share the eigenvectors because they commute. Noting that

$$P\theta_i^\dagger=\theta_i^\dagger, \tag{34}$$

we obtain that $\theta_i^\dagger$ is an eigenvector of P and so $\theta_i^\dagger$ is an eigenvector of D, but D is diagonal and with nonidentical diagonal entries, $\theta_i^\dagger$ much then be a one-hot vector, and P must also be diagonal and consists of values of 1 and 0 in the diagonal entries.

■

B.6. Proof of Theorem 6

Before we start proving Theorem 6, we introduce a definition that facilitates the proof.

Definition 9 *(Symmetry reduction.) We say that removing a symmetry from group G_1 reduces to removing the symmetry due to group G_2 if for any vector n,*

$$\|(I - \overline{G_2})n\| \leq \|(I - \overline{G_1})n\|. \tag{35}$$

Now, we are ready to prove the main theorem.

Proof We first show $(I - \overline{V})^T \nabla_\theta \ell(\theta) = 0$. For any $g \in V$ and $z \in \mathbb{R}$, we have

$$\ell(\theta + zn) = \ell(g(\theta + zn)), \tag{36}$$

where n is an arbitrary unit vector. Taking the derivative with respect to z, and recalling that $g\theta = \theta$, we have

$$(gn)^T \nabla_\theta \ell(\theta) = n^T \nabla_\theta \ell(\theta). \tag{37}$$

Accordingly, we have

$$(Vn)^T \nabla_\theta \ell(\theta) := \frac{1}{|V|} \sum_{g \in V} (gn)^T \nabla_\theta \ell(\theta) = n^T \nabla_\theta \ell(\theta). \tag{38}$$

Due to the arbitrary choice of n, we have $(I - \overline{V})^T \nabla_\theta \ell(\theta) = 0$.

Therefore,

$$\begin{aligned}
(I - \overline{V})^T \nabla_\theta \ell_{\mathrm{ar}}(\theta) &= \gamma (I - \overline{V})^T \nabla_\theta \|\theta - \theta_0\|_D^2 && (39)\\
&= 2\gamma (I - \overline{V})^T D(\theta - \theta_0) && (40)\\
&= 2\gamma (I - \overline{V})^T \theta_0 + o(\gamma \sigma_0 (1 + \sigma_D)). && (41)
\end{aligned}$$

As θ_0 is a Gaussian vector with mean 0 and variance σ_0^2, $\|(I - \overline{V})^T \theta_0\|$ is a Gaussian variable with mean 0 and variance $\|I - \overline{V}\|^2 \sigma_0^2 = \Omega(\mathrm{rank}(I - \overline{V})\sigma_0^2)$, which gives $\|(I - \overline{V})\nabla_\theta \ell\| = \Omega(\gamma \sigma_0 \sqrt{\mathrm{rank}(I - \overline{V})})$.

Now, we prove part (2) of the theorem. Note that if $V \triangleleft U$ and if $\theta_0 \in \ker \overline{V}$, then $\theta_0 \in \ker \overline{U}$. This means that for any group U such that $V \triangleleft U$ and any vector θ_0

$$\|(I - \overline{V})\theta_0\| < \|(I - \overline{U})\theta_0\|. \tag{42}$$

This means that to remove the symmetry from a large group U, it suffices to remove the symmetry from one of its minimal subgroups. Thus, let M_G denote the set of minimal subgroups of the group G, we have

$$\min_{V \triangleleft G} \|(I - \overline{V})\theta_0\| \geq \min_{V \triangleleft M_G} \|(I - \overline{V})\theta_0\|. \tag{43}$$

The number of minimal subgroups is strictly upper bounded by the number of elements of the group because all minimal subgroups are only trivially intersect each other. This follows from the fact that the intersection of groups must be a subgroup, which can only be the identity for two different minimal subgroups. Therefore, the number of minimal subgroups cannot exceed the number of elements of the group. This finishes the second part of the theorem.

For the third part, we show that the symmetry broken subspace of any subgroup contains the symmetry broken subspace of a group generated by one of the generators and so it suffices to only remove the symmetries due to the subgroups generated by each generator. Let us introduce the following notation for a matrix representation z of a group element:

$$\overline{z} = \frac{1}{\mathrm{ord(z)}} \sum_{i=1}^{\mathrm{ord(z)}} z^i, \tag{44}$$

where $\mathrm{ord}(z)$ denotes the order of z. This is equivalent to the symmetry projection matrix of the subgroup generated by z.

Now, let G be abelian. Then, both U and V are abelian. Let us denote by $\Gamma(U) = \{z_i\}$ the mininal generating set of U. Suppose that for all $n \neq 0$ such that $n \in \mathrm{im}(I - \overline{V})$, we must have $n \notin \mathrm{im}(I - \overline{z_j})$ for all j. This means that

$$n \notin \bigcup_j \mathrm{im}(I - \overline{z_j}). \tag{45}$$

Or, equivalently,

$$n \in \bigcap_j \mathrm{im}(\overline{z_j}). \tag{46}$$

However, the space $\bigcap_j \mathrm{im}(\overline{z_j}) \subseteq \mathrm{im}(\overline{V})$ because V is a subgroup of U, which is generated by $z_1, \cdots, z_m$. To see this, let $n \in \mathrm{im}(\overline{z_j})$ for all j, then,

$$z_j n = n \tag{47}$$

for all j. Now, let $v = \prod_i z_i^{d_i(v)} \in V$, we have

$$(I - \overline{V})n = (I - \sum_v \prod_i z_i^{d_i(v)})n = 0 \tag{48}$$

This means that n is in both $\mathrm{im}(\overline{V})$ and $\mathrm{im}(I - \overline{V})$, which is possible only if $n = 0$ – a contradiction. Therefore, as long as $I - \overline{V}$ is not rank 0, it must share a common subspace with one of the $I - \overline{z_j}$, and so removing the symmetry from any subgroup V of U can be reduced to removing the symmetry from the cyclic group generated by one of its generators from the minimal generating set.[6]

Therefore, we have proved that removing symmetries due to any subgroup of U can be reduced to removing the symmetry from a (proper or trivial) subgroup of each of the cyclic decompositions of the group U, each of which is generated by a minimal generator of U. By the fundamental theorem of finite abelian groups, each of these groups is of order p^k for some prime number p. Because each of these groups is cyclic, it contains exactly k nontrivial subgroups. Taken together, this means that if $|U| = p_1^{k_1} ... p_m^{k_m}$, we only have to remove symmetries from at most

$$\sum_i^m k_i = \log |U| \tag{49}$$

many subgroups. This completes part (3).

[6] This holds true even if V is a subset of $\langle z_j \rangle$.

For part (4), we denote $\Delta_i := (I - \overline{V_i})^T \nabla_\theta \ell_{\text{ar}}(\theta)$ for $i = 1, \cdots, |\Gamma(U)|$. According to (41), Δ_i is approximately a Gaussian variable with zero mean and variance $\gamma^2 \text{rank}(I - \overline{V})\sigma_0^2$. Therefore,

$$\sum_i^{|\Gamma(U)|} \Pr(|\Delta_i| > \epsilon) \geq |\Gamma(U)| - \frac{2\epsilon|\Gamma(U)|}{\min_i \sqrt{2\pi\gamma^2 \text{rank}(I - \overline{V})\sigma_0^2}}. \tag{50}$$

For $\Pr(\min_i |\Delta_i| > \epsilon)$ to be larger than $1 - \delta$, we must have

$$\gamma\sigma_0 \geq \frac{2\epsilon|\Gamma(U)|}{\sqrt{2\pi \text{ rank}(I - \overline{V})}(1 - \delta)}. \tag{51}$$

■

Appendix C. Experiment

First, we show that the proposed method is compatible with standard training methods. We then apply the method to a few settings where symmetry is known to be a major problem in training. We see that the algorithm leads to improved model performance on these problems.

C.1. Hyperparameter and Implementation Remark

As discussed, there are two ways to implement the method (Eq. (2) or Eq. (3)). In our experiments, we stick to the definition of Eq. (2), where the model parameters are biased, and weight decay is the same as the standard implementation. For the choice of hyperparameters, we always set $\sigma_D = 0$ as we find only introducing σ_0 to be sufficient for most tasks. Experiments with standard training settings (see the next section for the Resnet18 experiment on CIFAR-10) show that choosing σ_0 to be at least an order of magnitude smaller than the standard initialization scale (usually of order $1/\sqrt{d}$ for a width of d) works the best. We thus recommend a default value of σ_0 to be $0.01/\sqrt{d}$, where $1/d$ is the common initialization variance. For the rest of the paper, we state σ_0 in relative units of $\sqrt{d}^{-1}$ for this reason. That being said, we stress that σ_0 is a hyperparameter worth tuning, as it directly controls the tradeoff between optimization and symmetry removal.

C.2. Compatibility with Standard Training

Ridge linear regression. Let us first consider the classical problem of linear regression with d-dimensional data, where one wants to find $\min_w \sum_i (w^T x_i - y_i)^2$. Here, the use of weight decay has a well-known effect of preventing the divergence of generalization loss at a critical dataset size $N = d$ [18, 22]. This is due to the fact that the Hessian matrix of the loss becomes singular exactly at $N = d$ (at infinite N and d). The use of weight decay shifts all the eigenvalues of the Hessian by γ and removes this singularity. In this case, one can show that the proposed method is essentially identical to the simple ridge regression. The ridge solution is $w^* = \mathbb{E}[\gamma I + A]^{-1}\mathbb{E}[xy]$, where $A = \mathbb{E}[xx^T]$, and the solution to the biased model is

$$w^* = \mathbb{E}[\gamma I + A]^{-1}(\mathbb{E}[xy] + \gamma\theta_0). \tag{52}$$

The difference is negligible with the original solution if either γ and θ_0 are small. See Figure 3-left.

Reparametrized Linear Regression. A minimal model with emergent interest in the problem of compressing neural networks is the reparametrized version of linear regression [36], the loss function of which is $\ell(u, w) = \|(u \odot w)^T x - y\|^2$, where we let $u,\ w,\ x \in \mathbb{R}^{200}$ and $y \in \mathbb{R}$. Due to the rescaling symmetry between every parameter u_i and w_i, the solutions where $u_i = w_i = 0$ is a low-capacity state where the i-th neuron is "dead." For this problem, we compare the training with standard SGD and *syre*. We also compare with a heuristic method (**W-fix**), where a fraction $\phi = 0.3$ of weights of every layer is held fixed with a fixed variance $\kappa = 0.01$ at initialization. This method has been suggested in Ref. [24] as a heuristic for removing symmetries and is found to work well when there is permutation symmetry. We see that both the vanilla training and the W-fix collapse to low-capacity states during training, whereas the proposed method stayed away from them throughout. The reason is that the proposed method is model-independent and symmetry-agnostic, working effectively for any type of possibly unknown symmetry in an arbitrary architecture.

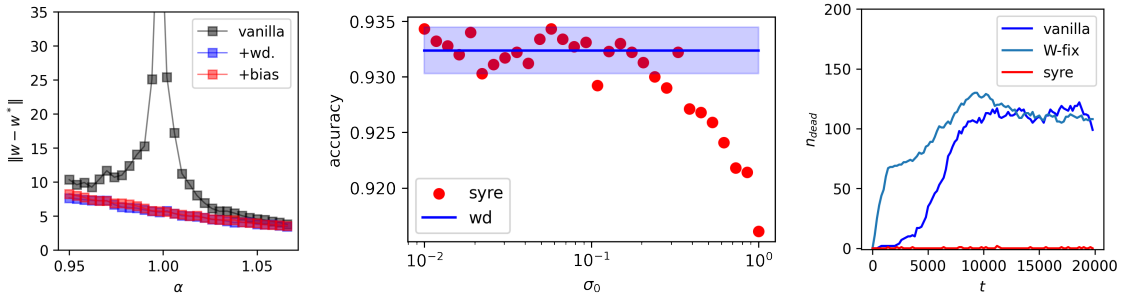

Figure 3: Training with syre in standard settings. The result shows that biasing the models by a small static bias does not change the performance of standard training settings. **Left**: Application of the method to a linear regression problem. Here, $\alpha^{-1} = d/N$ is the degree of parameterization. A well-known use of weight decay is to prevent double descent when $\alpha = 1$. Here, we see that the proposed method works as well as vanilla weight decay. Because there is no reflection symmetry in the problem, the proposed method should approximate vanilla weight decay. **Mid**: Test accuracy of Resnet18 on the CIFAR-10 datasets. The blue line denotes the performance of Resnet, and the shadowed area denotes its standard deviation estimated over 10 trials. For $\sigma_0 < 0.2$, there is no significant difference between the performance of the vanilla Resnet and *syre* Resnet. **Right**: linear regression with a redundant parametrization [29, 36]. The loss function takes the form $\ell(u, w) = \|(u \odot w)^T x - y\|^2$. Due to symmetry, the point $(u_i, w_i) = 0$ is a low-capacity state where the i-th neuron is "dead". Training with style, the model stayed away from any trapping low-capacity state during training. In comparison, training with vanilla SGD or a heuristic for fixing the weights does not fix the problem of collapsing to a low-capacity state.

ResNet. We also benchmark the performance of the proposed method for ResNet18 with different σ_0 on the CIFAR-10 datasets in Figure 3. When $\sigma_0 = 0$, the *syre* model is equivalent to the vanilla model. Figure 3 shows that the difference in performance between the vanilla Resnet and *syre* Resnet is very small and becomes neglectable when $\sigma_0 < 0.2$. We thus recommend a default value of $\sigma_0 \leq 0.01/\sqrt{d}$.

C.3. Feature and Neuron Collapses in Supervised Learning

See Figure 4, where we train the vanilla and *syre* four-layer networks with various levels of weight decay γ and various levels of input-output covariance α. The dataset is constructed by rescaling the input by a factor of α for the MNIST dataset. The theory predicts that the *syre* model can remove the permutation symmetry in the hidden layer. This is supported by subfigures in Figure 4, where vanilla training results in a low-rank solution. Meanwhile, the accuracy of the low-rank solution is significantly lower for a large γ or a small α, which corresponds to the so-called neural collapses. Also, we observe that *syre* shifts the eigenvalues of the representation by a magnitude proportional to σ_0, thus explaining the robustness of the method against collapses in the latent representation (See Figure 5).

Figure 5 gives the eigenvalue distribution of the networks in Figure 4, which further supports the claim that the vanilla network leads to a low-rank solution. In all the experiments above, we use a four-layer FCN with 300 neurons in each layer trained on the MNIST dataset with batch size 64.

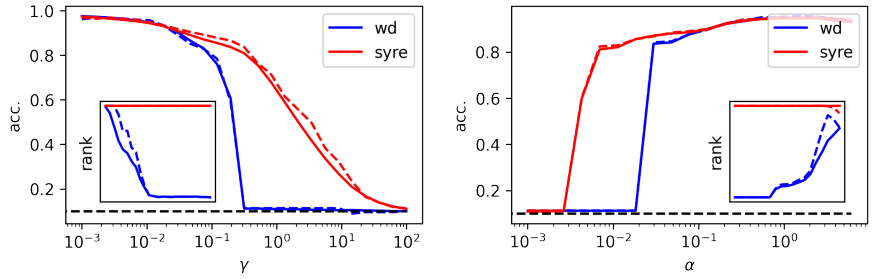

Figure 4: Performance of a 4-layer FCN for datasets with various weight decay γ and data distributions with varyings strengths of linear correlation α. As the theory predicts, the covariance of the vanilla model has a low-rank structure and performs significantly worse. In the main figures, solid lines denote training accuracy and dashed lines denote test accuracy. The dashed black line corresponds to random guess. Subfigures show the rank of the covariance matrix of the first layer output before (solid lines) and after (dashed lines) activation with batch size 1000 scaled by α^2. We set eigenvalues smaller than 10^{-4} to 0. **Left**: $\alpha = 1$ and different γ. **Right**: $\gamma = 0.01$ and different α.

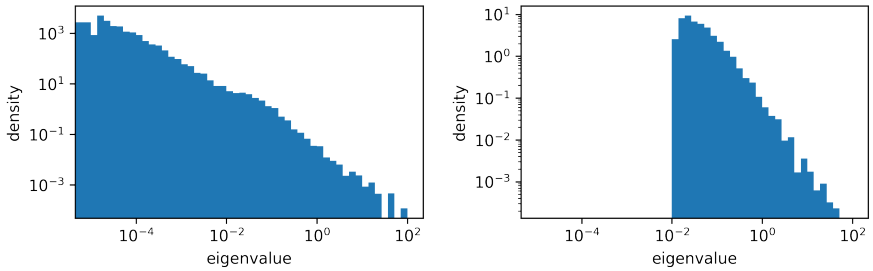

Figure 5: The spectrum of the covariance matrix of the vanilla model (**Left**) and the *syre* model (**Right**) for $\gamma = 0.01$ and $\alpha = 1$. Clearly, the vanilla model learns a low-rank solution.

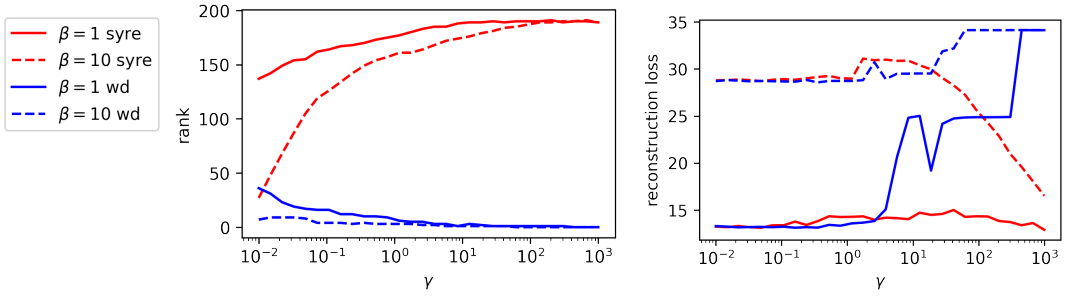

Figure 6: Rank and reconstruction loss for a VAE is trained on the Fashion MNIST dataset. The covariance of the vanilla model has a low-rank structure and larger reconstruction loss. More importantly, posterior collapse happens at $\beta = 5$ but is mitigated with weight decay. **Left**: the rank of the encoder output with batch size 1000. We set eigenvalues smaller than 10^{-6} to 0. **Right**: reconstruction loss of vanilla and *syre* models.

C.4. Posterior Collapse in Bayesian learning

Ref. [34] points out that a type of posterior collapse in Bayesian learning [25, 33] is caused by the low-rankness of the solutions. We show that training with *syre* could overcome this kind of posterior

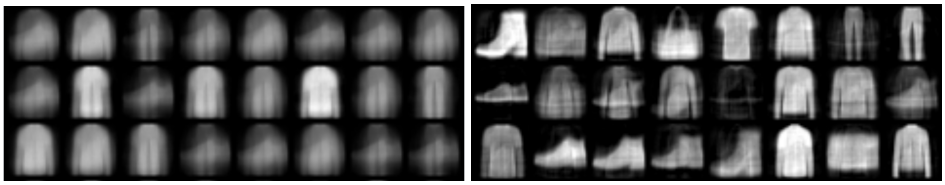

Figure 7: Examples of Fashion MNIST reconstruction with *syre* and $\beta = 10$. **Left**: No weight decay. **Right**: $\gamma = 1000$. Clearly, the posterior collapse is mitigated by imposing *syre* with weight decay.

	Hyperparameter	Low-rankness	Penult. Layer Acc.	Last Layer Acc.
vanilla	-	70%	46.8%	22.2%
syre	$\sigma_0 = 0.1$	0%	46.8%	31.7%
	$\sigma_0 = 0.01$	0%	46.2%	32.5%
	$\sigma_0 = 0.1$, all layers	0%	44.6%	30.7%
	$\sigma_0 = 0.01$, all layers	0%	45.4%	32.4%

Table 1: Performance of the linearly evaluated Resnet18 on CIFAR100 for the unsupervised self-constrastive learning task. Here, the low-rankness measures the proportion of eigenvalues smaller than 10^{-5}. Our experiment indicates that symmetry-induced reduction in model capacity can explain about 50% of the performance difference between the representation of the two layers.

collapse. In Figure 6, we train a β-VAE [19, 20] on the Fashion MNIST dataset. Following Ref. [34], we use β to weigh the KL loss, which can be regarded as the strength of prior matching. Both the encoder and the decoder are a two-layer network with SiLU activation. The hidden dimension and the latent dimension are 200. Only the encoder has weight decay because the low-rank problem is caused by the encoder rather than the decoder. We also choose the prior variance of the latent variable to be $\eta_{enc} = 0.01$. Other settings are the same as Ref. [34]. Posterior collapse happens at $\beta = 10$, signalized by a large reconstruction loss in the right side of Figure 6. However, the reconstruction loss decreases, and the rank of the encoder output increases (according to the left side of Figure 6) after we use weight decay and *syre*. This is further verified by the generated image in Figure 7. Therefore, we successfully remove the permutation symmetry of the encoder.

C.5. Low-Capacity Trap in Self-supervised Learning

A common but bizarre practice in self-supervised learning (SSL) is to throw away the last layer of the trained model and use the penultimate learning representation, which is found to have much better expressiveness than the last layer representation. From the perspective of symmetry, this problem is caused by the rotation symmetry of the last weight matrix in the SimCLR loss. We train a Resnet-18 together with a two-layer projection head over the CIFAR-100 dataset according to the setting for training SimCLR in Ref. [6]. Then, a linear classifier is trained using the learned representations. Our implementation reproduces the typical accuracy of SimCLR over the CIFAR-100 dataset [28]. As in Ref. [6], the hidden layer before the projection head is found to be a better representation than the layer after. Therefore, we apply our *syre* method to the projection head or to all layers. According to Table 1, *syre* removes the low-rankness of the learned features and increases the accuracy trained with the features after projection while not changing the representation before projection. Thus, symmetry-induced reduction in model capacity can explain about 50% of the per-

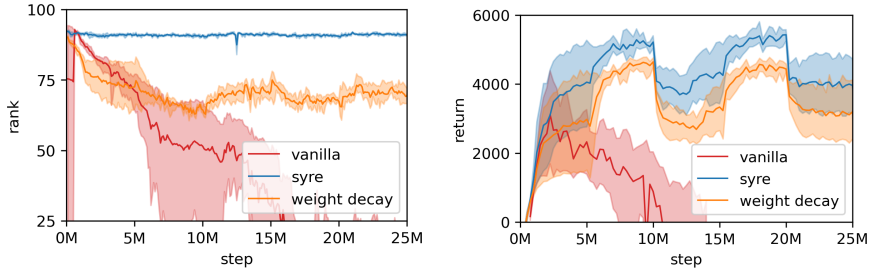

Figure 8: Loss of plasticity in continual learning in an RL setting. We use the PPO algorithm [30] to solve the Slippery-Ant problem [11]. The rank and the performance of the vanilla PPO decrease quickly, while the rank and the performance of PPO with *syre* remain the same, beyond that of PPO with weight decay. **Left**: the effective rank of the policy network as defined in Ref. [11]. **Right**: returns. Each trajectory is averaged over 5 different random seeds.

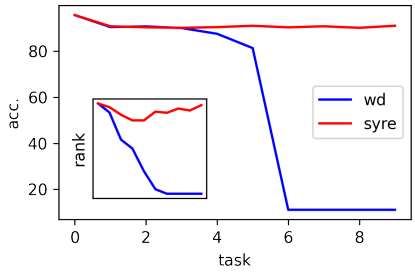

Figure 9: Performance and accuracy of a CNN trained on a continual learning task (permuted MNIST [16, 21]). The main figure shows the test accuracy, and the subfigure shows the rank of the convolution layers output with batch size 1000, where we set eigenvalues smaller than 10^{-4} to 0. The results suggest that the rank of the vanilla model gradually decreases, and the model completely collapses after the sixth task, while the *syre* model remains unaffected.

formance difference between the representation of the two layers. Also, an interesting observation is that just improving the expressivity of the last layer is insufficient to close the gap between the performance of the last layer and the penultimate layer. This helps us gain a new insight: symmetry is not the only reason why the last layer representation is defective.

C.6. Loss of Plasticity in Continual Learning

A form of low-capacity collapse also happens during continual learning, i.e., the plasticity of the network gradually decreases as the model is trained on more and more tasks. This problem is common in both supervised and reinforcement learning settings and may also be relevant to the finetuning of large language models [1, 2, 10, 26].

Supervised Learning. In Figure 9, we train a CNN with two convolution layers (10 channels and 20 channels) and two fully connected layers (320 units and 50 units) over the MNIST datasets. For the data, we randomly permute the pixels of the training and test sets for 9 times, forming 10 different tasks (including the original MNIST). We then train a vanilla CNN and a *syre* CNN over the 10 tasks continually with SGD and weight decay 0.01. The inset of Figure 9 shows that

the rank of the original model gradually decreases, but the *syre* model remains close to full rank. Correspondingly, in the right side of Figure 9, the accuracy over the test set drops while the rank of the original model collapses, but the accuracy of the *syre* model remains similar.

Reinforcement Learning. In Figure 8, we use the PPO algorithm for the Slippery-Ant problem [11], a continual variant of the Pybullet's Ant problem [8] with friction that changes every $5M$ steps. Hyperparameters for the PPO algorithm are borrowed from Ref. [11], and we use a weight decay of 0.002 for both PPO with weight decay and with *syre*. Figure 8 suggests that *syre* is effective in maintaining the rank of the model during continual training and obtains better performance than pure weight decay.