Equivariance Discovery by Learned Parameter-Sharing

Raymond A. Yeh[†] **Yuan-Ting Hu** **Mark Hasegawa-Johnson** **Alexander G. Schwing**
Toyota Technological Institute at Chicago[†] University of Illinois at Urbana-Champaign

Abstract

Designing equivariance as an inductive bias into deep-nets has been a prominent approach to build effective models, *e.g.*, a convolutional neural network incorporates translation equivariance. However, incorporating these inductive biases requires knowledge about the equivariance properties of the data, which may not be available, *e.g.*, when encountering a new domain. To address this, we study how to *discover interpretable equivariances* from data. Specifically, we formulate this discovery process as an optimization problem over a model's parameter-sharing schemes. We propose to use the partition distance to empirically quantify the accuracy of the recovered equivariance. Also, we theoretically analyze the method for Gaussian data and provide a bound on the mean squared gap between the studied discovery scheme and the oracle scheme. Empirically, we show that the approach recovers known equivariances, such as permutations and shifts, on sum of numbers and spatially-invariant data.

1 INTRODUCTION

Encoding equivariance and invariance into deep-nets has been an effective method to improve the data-efficiency of machine learning models. For example, convolutional neural nets (CNNs) or recurrent neural nets (RNNs) encode shift-equivariant properties either in the spatial or temporal domain (LeCun et al., 1999; Hochreiter & Schmidhuber, 1997). More recently, equivariance has also been studied in other domains, *e.g.*, over sets, graphs, or other geometric structures (Zaheer et al., 2017; Bronstein et al., 2017).

Proceedings of the 25[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

While encoding equivariance has been remarkably successful, encoding requires a-priori knowledge about the desirable equivariance properties to be built into a model. Such knowledge of equivariance requires domain expertise which may not be available. To tackle this concern, we study *discovery of interpretable equivariance* from data rather than manually imposing it.

For this, we consider discovery of equivariance over a family of discrete group actions. This family of equivariances can be built into deep-nets via parameter-sharing (Ravanbakhsh et al., 2017a). In other words, if we can learn how to share parameters, then we can discover equivariance. To achieve this goal, we identify a parametric representation of the parameter-sharing scheme. This permits to cast the discovery process as an optimization problem. Intuitively, we aim to find the parameter-sharing scheme that results in the best generalization capability. To estimate this generalization capability we use empirical data in a validation set. This results in a bi-level optimization: optimize for the best parameter-sharing scheme on a validation set, given that the model parameters which use this sharing are 'optimal' on the training set.

Contributions:
1) We theoretically analyze the benefits of learning a parameter-sharing scheme and show how to choose the validation and training set that are required in the proposed algorithm. For a family of multivariate Gaussian distributions with a shared mean, we show that the proposed method provably yields better generalization in terms of a mean squared error than standard maximum likelihood training.

2) We also study how to evaluate the learned sharing scheme. Specifically, we advocate for the use of partition distance as a quantitative metric. This differs from prior practice which relies on visual inspection. Finally, we discuss practical considerations for using the proposed approach and validate its effectiveness through a range of experiments. Empirically, we demonstrate that the approach can recover known permutation invariance and spatial equivariance from data.

2 RELATED WORK

We briefly highlight works on designing equivariance and invariance in machine learning. Then we review recent advances towards discovering equivariance from data. Lastly, we discuss how hyperparameter optimization is related to our work.

Invariance and Equivariance. Designing invariance and equivariance representations has been widely utilized in building effective models. Well-known examples are hand-crafted features in computer vision, such as SIFT (Lowe et al., 1999) which is scale invariant, or shift-invariant systems (Vetterli et al., 2014) in signal processing. Naturally, learning-based representations have also adopted these properties. For example, the widely used CNN (LeCun et al., 1999) or RNN (Hochreiter & Schmidhuber, 1997) are shift-invariant in space or time. The success of CNNs has also been generalized to other sets of equivariances. For example, Cohen & Welling (2016) propose a group-equivariant CNN, which is equivariant to rotations, reflections and translations, or TI-pooling (Laptev et al., 2016), which pools over the desirable transformations to achieve invariance. Other architectures, *e.g.*, equivariant transformers (Tai et al., 2019; Fuchs et al., 2020; Romero & Cordonnier, 2021) have also been studied.

Equivariance has also been extended to other domains, such as sets, which are permutation invariant (Ravanbakhsh et al., 2017b; Zaheer et al., 2017; Qi et al., 2017; Maron et al., 2020), graphs (Shuman et al., 2013; Defferrard et al., 2016; Kipf & Welling, 2017; Maron et al., 2019), meshes (de Haan et al., 2021), spherical images (Cohen et al., 2018; Kondor et al., 2018), keypoints (Yeh et al., 2019a), trajectories (Yeh et al., 2019b; Liu* et al., 2019, 2021) and tabular data (Hartford et al., 2018). These works demonstrate that designing equivariance into models/representations is beneficial. However, these approaches require a practitioner to select the suitable equivariance properties. Instead, in this work, we are interested in discovering this equivariance property explicitly from data rather than manually imposing it.

Learning Equivariance. Recently, Benton et al. (2020) proposed to learn invariance for deep-nets from data. At a high-level, their approach achieves invariance by applying augmentations at the input and averaging the output. To learn the invariance, they parameterize the augmentation distribution and jointly learn these parameters with the deep-net's model parameters. Note that the model is only invariant to the sampled augmentations which may require many samples. *E.g.*, for the permutation group, their approach requires to sample all permutations to achieve invariance. In contrast, we achieve equivariance through parameter-sharing, and cast equivariance discovery as learning the parameter-sharing scheme.

In recent work, Zhou et al. (2021) consider learning equivariance in a meta-learning framework. Our work differs in the following ways: (a) We use an assignment matrix consisting of elements between zero and one to parameterize the sharing of parameters while Zhou et al. (2021) do not enforce any constraints. (b) This constraint permits to quantitatively evaluated the discovered schemes. We propose to assess the discovered sharing scheme via the partition distance (PD) (Gusfield, 2002), while prior works rely on visual inspection. (c) We demonstrate that the proposed method has advantages over standard maximum likelihood training without parameter-sharing on multivariate Gaussian distributions with a shared mean. We also show a trade-off between the size of training and validation sets. Next, we review hyperparameter optimization, as the sharing scheme can be viewed as a hyperparameter.

Hyperparameter Optimization. Typically formulated as a bi-level optimization problem, hyperparameter optimization consists of an upper/lower-level optimization task which, respectively, minimizes the loss on a validation/training set. Numerous hypergradient based methods have been proposed to solve this problem (Larsen et al., 1996; Bengio, 2000; Maclaurin et al., 2015; Luketina et al., 2016; Shaban et al., 2018; Lorraine et al., 2020; Ren* et al., 2020). Lorraine et al. (2020) provide a comprehensive review.

As bi-level optimization requires a validation set, here we also discuss how to select this set. Prior works have studied how to split a validation set (Kearns, 1996; Guyon et al., 1997; Amari et al., 1997) and what test set size is necessary to yield a good generalization error estimate (Guyon et al., 1998). More recently Afendras & Markatou (2019) study the optimal size of the validation set in the context of cross-validation. In this work, we optimize and study a specific hyperparameter, *i.e.*, the parameter-sharing scheme, to achieve equivariance discovery.

3 PRELIMINARIES

Abstractly, equivariance and invariance capture properties of a function's input-output relationship. Consider the task of image segmentation. If an object is shifted within the image, one would expect the predicted segmentation to shift accordingly, *i.e.*, the model is *shift-equivariant*. Similarly, for the task of image classification, the class prediction should remain the same for a shifted object. In this case, the classifier is *shift-invariant*. Incorporating these properties into a multilayer perceptron (MLP) via parameter-sharing results in a convolutional neural net (CNN). To generalize

this success, Ravanbakhsh et al. (2017a) theoretically study types of equivariances that can be encoded via parameter-sharing, and how to construct such layers. We will briefly review their results in the remainder of this section.

We start with the definitions of equivariance and invariance over the family of discrete group actions. A function $f : \mathbb{R}^N \mapsto \mathbb{R}^M$ is $\mathcal{G}_{N,M}$**-equivariant** if and only if *(iff)*

$$f(P_{\pi_N}\mathbf{x}) = P_{\pi_M} f(\mathbf{x}) \;\; \forall(\pi_N, \pi_M) \in \mathcal{G}_{N,M}, \; \mathbf{x} \in \mathbb{R}^N, \quad (1)$$

where P_π denotes a permutation matrix and $\mathcal{G}_{N,M}$ denotes the set of group actions each characterized by two permutations π_N and π_M on input and output dimensions. Informally, equivariance describes how the output changes when the input is transformed in a "predefined way." Similarly, a function is $\mathcal{G}_N$**-invariant** *iff*

$$f(P_{\pi_N}\mathbf{x}) = f(\mathbf{x}) \;\; \forall\pi_N \in \mathcal{G}_N, \; \mathbf{x} \in \mathbb{R}^N. \quad (2)$$

Invariance is a special case of equivariance where $P_{\pi_M} = I_M$ is the identity matrix, *i.e.*, the output remains the same. Next, we discuss how to construct deep-nets that satisfy equivariance.

Equivariance Through Parameter-Sharing. Ravanbakhsh et al. (2017a) theoretically study how to design deep-nets that are equivariant to any discrete group action, which are characterized above via permutation matrices. They prove that symmetries in model parameters, *i.e.*, the sharing of the parameters, leads to equivariance. Importantly, a fully connected layer $f_{\boldsymbol{W}}$ can be designed to be $\mathcal{G}_{N,M}$-equivariant for any set of group actions. For example, a fully connected layer, $f_{\boldsymbol{W}}(\mathbf{x}) \triangleq \boldsymbol{W}\mathbf{x}$, where $\boldsymbol{W} \in \mathbb{R}^{M\times N}$, is $\mathcal{G}_{N,M}$-equivariant if the weight $\boldsymbol{W}$ satisfies the following sharing scheme:

$$\boldsymbol{W}_{m,n} = \boldsymbol{W}_{\pi_M(m),\pi_N(n)} \;\; \forall(\pi_N, \pi_M) \in \mathcal{G}_{N,M}. \quad (3)$$

They also propose a method to achieve such a sharing. In summary, by tying parameters, one can design fully connected layers that are equivariant to various discrete group actions. Note that Ravanbakhsh et al. (2017a) *design a model given an equivariance property*, *i.e.*, a practitioner needs to decide which equivariance to build into a model. In contrast, we study *how to discover equivariance* from data.

4 EQUIVARIANCE DISCOVERY BY LEARNED PARAMETER-SHARING

Here, we are interested in discovering equivariance from data, *i.e.*, learning explicit parameter-sharing schemes rather than manually imposing them.

Consider a supervised learning setup, given a dataset $\mathcal{D} = \{(\mathbf{x}, \mathbf{y})\}$, the goal is to learn the parameters $\boldsymbol{\theta}$ of a model $f_{\boldsymbol{\theta}}(\mathbf{x})$, by minimizing a desired loss function $\mathcal{L}$, *i.e.*,

$$\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \mathcal{D}) = \min_{\boldsymbol{\theta}} \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{D}} \ell(f_{\boldsymbol{\theta}}(\mathbf{x}), \mathbf{y}). \quad (4)$$

To discover equivariance, we introduce a parametric representation of the sharing scheme and develop an algorithm to optimize over it.

Parameterizing Parameter-Sharing. We use an assignment matrix $\boldsymbol{A}$ to select which of the parameters are shared. Formally, let

$$\boldsymbol{\theta} = \boldsymbol{A}\boldsymbol{\psi}, \quad (5)$$

where $\boldsymbol{\theta}, \boldsymbol{\psi} \in \mathbb{R}^K$, $\boldsymbol{A} \in \{0,1\}^{K\times K}$, and $\boldsymbol{A}$ is row stochastic, *i.e.*, $\forall i \sum_j \boldsymbol{A}_{ij} = 1$. Hence, entries in $\boldsymbol{\theta}$ may originate from the same entries in $\boldsymbol{\psi}$. Using $\boldsymbol{A}$, we can represent all possible sharing configurations, which in turn permits to incorporate different equivariances. For example, $\boldsymbol{A}$ encoding a Toeplitz matrix results in the convolution operation, *i.e.*, shift equivariance. With this parametrization at hand, we now describe how to learn both $\boldsymbol{A}$ and $\boldsymbol{\psi}$ from data.

Learning Parameter-Sharing. Note, jointly optimizing $\boldsymbol{\psi}$ and $\boldsymbol{A}$ on $\mathcal{D}$ doesn't yield the desired result: the trivial solution $\boldsymbol{A} = \boldsymbol{I}$ selects all the parameters, leading to the lowest loss on the training set. This is not necessarily desirable.

Recall, the motivation for an equivariant model is to improve *generalization*. Therefore, we directly estimate generalization on data. For this we split the dataset $\mathcal{D}$ into two sets, the training set $\mathcal{T}$ and the validation set $\mathcal{V}$. We then aim to solve the following bi-level program:

$$\overbrace{\min_{\boldsymbol{A}} \mathcal{L}(\underbrace{\boldsymbol{A}\boldsymbol{\psi}^*(\boldsymbol{A})}_{\boldsymbol{\theta}}, \mathcal{V})}^{\text{upper-level task}} \text{ s.t. } \boldsymbol{\psi}^*(\boldsymbol{A}) = \overbrace{\arg\min_{\boldsymbol{\psi}} \mathcal{L}(\boldsymbol{A}\boldsymbol{\psi}, \mathcal{T})}^{\text{lower-level task}},$$
$$\boldsymbol{A} \in \{0,1\}^{K\times K}, \sum_j \boldsymbol{A}_{ij} = 1 \;\; \forall i. \quad (6)$$

Intuitively, we aim to find the "best sharing scheme" on validation set $\mathcal{V}$ (upper-level task), given model parameters $\boldsymbol{\psi}$ trained on $\mathcal{T}$ (lower-level task). Once having discovered the optimal parameter-sharing scheme $\boldsymbol{A}_{\texttt{val}}$, we fix it and train $\boldsymbol{\psi}$ on the entire dataset $\mathcal{D}$ to obtain the best model.

This naturally leads to the following questions: (a) Is learning $\boldsymbol{A}$ beneficial?; and (b) Can the optimization in Eq. (6) be solved efficiently? We analyze (a) in Sec. 4.1, followed by discussing considerations for (b) in Sec. 4.2.

4.1 Analysis on Gaussian Data

We analyze the approach given in Eq. (6) on K-dimensional Gaussian vectors that are independent and identically distributed (*i.i.d.*), *i.e.*,

$$\mathbf{y} \sim \mathcal{N}(\boldsymbol{A}_{\texttt{gt}}\boldsymbol{\psi}_{\texttt{gt}}, \sigma^2\boldsymbol{I}). \tag{7}$$

Here, $\boldsymbol{A}_{\texttt{gt}}$ denotes the unknown ground-truth sharing scheme. The task is to estimate the ground-truth mean, $\boldsymbol{\theta}_{\texttt{gt}} \triangleq \boldsymbol{A}_{\texttt{gt}}\boldsymbol{\psi}_{\texttt{gt}}$ and the sharing scheme $\boldsymbol{A}_{\texttt{gt}}$. With our supervised learning setup, $\mathcal{D} = \{\mathbf{y}\}$, $f_{\boldsymbol{\theta}} = \boldsymbol{\theta}$ and $\ell = \ell_2$.

Benefits of Learning $\boldsymbol{A}$. Let $\hat{\boldsymbol{\theta}}(\mathcal{D})$ be the maximum likelihood estimate on dataset $\mathcal{D}$. To analyze, we consider the mean squared error (MSE) of an estimator,

$$\begin{aligned}\text{MSE}(\hat{\boldsymbol{\theta}}(\mathcal{D})) &\triangleq \mathbb{E}\left\|\hat{\boldsymbol{\theta}}(\mathcal{D}) - \boldsymbol{\theta}_{\texttt{gt}}\right\|^2 \\ &= \left\|\text{Bias}(\hat{\boldsymbol{\theta}}(\mathcal{D}))\right\|^2 + \text{Trace}(\mathbb{V}(\hat{\boldsymbol{\theta}}(\mathcal{D}))),\end{aligned} \tag{8}$$

where $\text{Bias}(\hat{\boldsymbol{\theta}}) = \mathbb{E}(\hat{\boldsymbol{\theta}}) - \boldsymbol{\theta}_{\texttt{gt}}$, and $\mathbb{V}(\hat{\boldsymbol{\theta}})$ denotes the covariance matrix of $\hat{\boldsymbol{\theta}}$. **Note:** the expectation is with respect to the distribution that generated the finite dataset with $|\mathcal{D}|$ number of samples, *i.e.*, $\hat{\boldsymbol{\theta}}(\mathcal{D})$ is a random variable and $\boldsymbol{\theta}_{\texttt{gt}}$ is fixed (Wasserman, 2013).

We further let $\boldsymbol{A}_{\texttt{val}}$ denote the sharing scheme discovered using Eq. (6) and $\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{D})$ is the maximum likelihood estimate on $\mathcal{D}$ following the sharing scheme $\boldsymbol{A}_{\texttt{val}}$, *i.e.*, $\hat{\boldsymbol{\theta}}_{\texttt{val}} = \boldsymbol{A}_{\texttt{val}}\boldsymbol{\psi}_{\texttt{val}}$. With the notation defined, we study the form of the mean squared error given an estimator using the parameter-sharing scheme $\boldsymbol{A}_{\texttt{val}}$. Specifically, we identify the role of the rank $\text{rk}(\boldsymbol{A}_{\texttt{val}})$ in the MSE.

Claim 1. *For Gaussian data (Eq. (7)) and a given sharing scheme $\boldsymbol{A}_{\texttt{val}}$ we have*

$$\textit{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{D})) = \left\|\boldsymbol{A}_{\texttt{val}}\bar{\boldsymbol{A}}_{\texttt{val}}^{\intercal}\boldsymbol{\theta}_{\texttt{gt}} - \boldsymbol{\theta}_{\texttt{gt}}\right\|^2 + \frac{\textit{rk}(\boldsymbol{A}_{\texttt{val}})\sigma^2}{|\mathcal{D}|},$$

where $\bar{\boldsymbol{A}}_{\texttt{val}}$ refers to the column normalized $\boldsymbol{A}_{\texttt{val}}$ and $\textit{rk}(\cdot)$ denotes the rank of a matrix.

Proof. Let S_i denote the set of indices that share parameters for the i^{th} dimension of $\hat{\boldsymbol{\theta}}_{\texttt{val}}$ as characterized in $\boldsymbol{A}_{\texttt{val}}$. Formally, $S_i \triangleq \{k \in \{1,\dots,K\} |\ \forall j\ \boldsymbol{A}_{\texttt{val}}[i,j] = 1 \wedge \boldsymbol{A}_{\texttt{val}}[k,j] = 1\}$. As the dimensions and samples are independent, the maximum likelihood estimator is the average over the shared dimensions and samples, *i.e.*,

$$\mathbb{E}(\hat{\boldsymbol{\theta}}_{\texttt{val}}[i]) = \frac{1}{|S_i|}\sum_{k\in S_i}\boldsymbol{\theta}_{\texttt{gt}}[k], \tag{9}$$

Similarly, the variance is

$$\mathbb{V}(\hat{\boldsymbol{\theta}}_{\texttt{val}}[i]) = \frac{\sigma^2}{|S_i||\mathcal{D}|}. \tag{10}$$

Substituting Eq. (9) and Eq. (10) into the MSE definition in Eq. (8) concludes the proof. Additional details are deferred to Appendix Sec. A. □

Consider the case without any parameter-sharing, *i.e.*, the parameters are independent ($\boldsymbol{A}_{\texttt{ind}} = \boldsymbol{I}$). As the estimator is unbiased, we obtain

$$\text{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{ind}}(\mathcal{D})) = \frac{K\sigma^2}{|\mathcal{D}|}. \tag{11}$$

Observe that when $K \geq \text{rk}(\boldsymbol{A}_{\texttt{val}}) \geq \text{rk}(\boldsymbol{A}_{\texttt{gt}})$ then there exists an *unbiased* $\boldsymbol{A}_{\texttt{val}}$ (as rank is larger) such that the MSE (in Claim 1) is lower than the MSE without parameter sharing (*i.e.*, when using $\boldsymbol{A}_{\texttt{ind}}$). Specifically, there exists an unbiased estimator, $\text{Bias}(\hat{\boldsymbol{\theta}}) = 0$, with a lower variance term compare to the estimator $\hat{\boldsymbol{\theta}}_{\texttt{ind}}$ without any parameter sharing. This means that it is possible to find an estimator that generalizes better in terms of MSE.

Next, we show that the procedure in Eq. (6) will select such an $\boldsymbol{A}_{\texttt{val}}$. Recall, the algorithm selects $\boldsymbol{A}_{\texttt{val}}$ based on the loss over the empirically sampled validation set $\mathcal{V}$, *i.e.*,

$$\mathcal{L}(\hat{\boldsymbol{\theta}}, \mathcal{V}) = \sum_{\mathbf{y}\in\mathcal{V}}\left\|\hat{\boldsymbol{\theta}} - \mathbf{y}\right\|^2 \geq \left\|\hat{\boldsymbol{\theta}} - \frac{1}{|\mathcal{V}|}\sum_{\mathbf{y}\in\mathcal{V}}\mathbf{y}\right\|^2. \tag{12}$$

Eq. (12) (left-hand side) is an upper bound on the squared error between the parameter estimates. Left- and right-hand side have the same global minimum. Therefore, we directly analyze $\widehat{\text{MSE}}$, an MSE estimate based on the validation set $\mathcal{V}$, defined as follows:

$$\widehat{\text{MSE}}(\hat{\boldsymbol{\theta}}) = \mathbb{E}\left\|\hat{\boldsymbol{\theta}} - \hat{\boldsymbol{\theta}}_{\texttt{ind}}(\mathcal{V})\right\|^2. \tag{13}$$

Note that the expectation is w.r.t. the estimator $\hat{\boldsymbol{\theta}}$.

Recall, the procedure in Eq. (6) selects the sharing scheme $\boldsymbol{A}_{\texttt{val}}$ which minimizes the $\widehat{\text{MSE}}$. By law of large numbers, when $|\mathcal{V}| \to \infty$ then $\widehat{\text{MSE}}(\hat{\boldsymbol{\theta}}) \to \text{MSE}(\hat{\boldsymbol{\theta}})$. This means that the program in Eq. (6) characterizes the $\boldsymbol{A}_{\texttt{val}}$ which minimizes the MSE if $\mathcal{V}$ is sufficiently large.

In practice, we have a limited amount of data. Therefore we further study the finite sample behavior and how to decide the sizes of training and validation sets.

Finite Sample Analysis. We study the MSE gap

$$\text{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{D})) - \text{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{gt}}(\mathcal{D})) \tag{14}$$

between learned and ground-truth *sharing scheme*. This gap is useful as it measures the quality of the estimator of the proposed procedure. Specifically, we

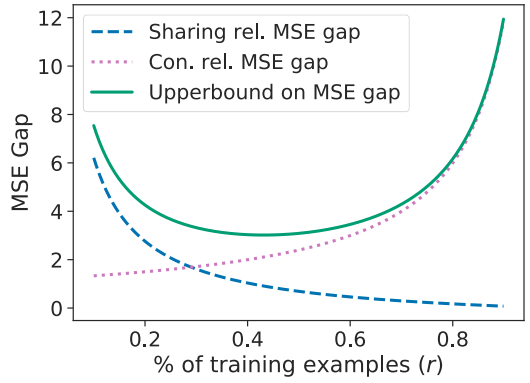

Figure 1: Illustration of the upper bound on the MSE gap in Eq. (15).

construct an upper bound to identify the role of dataset sizes $|\mathcal{T}|$ and $|\mathcal{V}|$. Recall, that we split a given dataset $\mathcal{D}$ into a training set $\mathcal{T}$ and a validation set $\mathcal{V}$, which affects the characterized $\boldsymbol{A}_{\texttt{val}}$.

Claim 2. *Given data drawn i.i.d. following Eq. (7), with probability* $1-\alpha$ *and* $\alpha < \exp\frac{-K}{10}$*, the MSE gap in Eq. (14) is upper bounded by*

$$\sigma^2\Big(\underbrace{\frac{1-r}{r|\mathcal{D}|}\left(rk(\boldsymbol{A}_{\texttt{gt}})-1\right)}_{\textit{sharing rel. MSE gap}} - \underbrace{\frac{40\ln(\alpha)}{(1-r)|\mathcal{D}|}}_{\textit{con. rel. MSE gap}}\Big), \quad (15)$$

where $r = \frac{|\mathcal{T}|}{|\mathcal{D}|}$ *denotes the ratio between the size of training and overall dataset.*

Proof sketch. The high-level idea is to decompose Eq. (14) into three parts:

$$\mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{D})) - \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{T})), \quad (16)$$
$$+ \quad \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{T})) - \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{gt}}(\mathcal{T})), \quad (17)$$
$$+ \quad \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{gt}}(\mathcal{T})) - \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{gt}}(\mathcal{D})). \quad (18)$$

We prove the claim by upper bounding each of the terms. The complete proof is deferred to Appendix Sec. B. □

We now highlight this results' significance. Observe that the upper bound in Eq. (15) consists of two terms, a *sharing* related MSE gap and a *confidence* related MSE gap. The two terms form a trade-off w.r.t. the optimal percentage of the training examples. We provide an illustration in Fig. 1 demonstrating the trade-off between the sharing term and the confidence term. Note, the optimal validation set size may be much larger than the commonly used 80-20 train/val split.

Intuitively, when there is less parameter sharing, *i.e.*, $\mathrm{rk}(\boldsymbol{A}_{\texttt{gt}})$ is large, then more training data should be used to get a good estimate of the model parameters.

Algorithm 1 Equivariance discovery via learned parameter-sharing

Initialize model parameters $\boldsymbol{\psi}$, $\boldsymbol{A}$
while not converged **do**
  Sample batch $\mathcal{V}' \subseteq \mathcal{V}$.
  # Solve lower-level task.
  $\boldsymbol{\psi}^*(\boldsymbol{A}) \leftarrow \arg\min_{\boldsymbol{\psi}} \mathcal{L}(\boldsymbol{A}\boldsymbol{\psi}, \mathcal{T})$
  $\boldsymbol{A} \leftarrow \boldsymbol{A} - \eta \cdot \nabla_{\boldsymbol{A}}\big(\mathcal{L}(\boldsymbol{A}\boldsymbol{\psi}^*, \mathcal{V}') + H(\boldsymbol{A}) + \|\boldsymbol{A}\|_*\big)$
end while
$\boldsymbol{A}_{\texttt{val}} \leftarrow \boldsymbol{A}$
Train with all the data with fixed $\boldsymbol{A}_{\texttt{val}}$.
$\boldsymbol{\psi}^*(\boldsymbol{A}_{\texttt{val}}) \leftarrow \arg\min_{\boldsymbol{\psi}} \mathcal{L}(\boldsymbol{A}_{\texttt{val}}\boldsymbol{\psi}, \mathcal{D})$
Return $\boldsymbol{A}_{\texttt{val}}, \boldsymbol{\psi}^*(\boldsymbol{A}_{\texttt{val}})$

Similarly, if one aims to have more confidence in $\boldsymbol{A}_{\texttt{val}}$, *i.e.*, $-\ln(\alpha)$ is large, then more validation data should be used. Also, this bound suggests that it can be desirable to use a validation set that is larger than the training set, which is not a common practice to date.

Further, we can use the upper bound to identify data distributions where the proposed algorithm is provably better in generalization than standard maximum likelihood training without parameter sharing.

For example, when $\mathrm{rk}(\boldsymbol{A}_{\texttt{gt}}) = 1$,

$$\begin{aligned}\mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{D})) &\leq \frac{-40\ln\alpha}{(1-r)|\mathcal{D}|}\sigma^2 + \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{gt}}(\mathcal{D})) \\ &= \frac{-40\ln\alpha}{(1-r)|\mathcal{D}|}\sigma^2 + \frac{\sigma^2}{|\mathcal{D}|} \\ &\leq \frac{K\sigma^2}{|\mathcal{D}|} = \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{ind}}) \end{aligned} \quad (19)$$

given that K is sufficiently large. This shows that the sharing approach achieves a lower MSE than standard maximum likelihood training where all the parameters are independent, *i.e.*, $\hat{\boldsymbol{\theta}}_{\texttt{ind}}$.

We will next discuss how to address the bi-level optimization in Eq. (6).

4.2 Practical Considerations

For really small-scale problems a brute-force search over all $\boldsymbol{A}$ solves the proposed program in Eq. (6). However, brute-force search quickly becomes infeasible when the dimensions grow. Instead of brute-force search, we relax the integrality constraint to $\boldsymbol{A} \in [0,1]^{K\times K}$. This permits the use of continuous optimization, *e.g.*, projected gradient descent on $\boldsymbol{A}$ or use of a softmax to avoid the constraints altogether.

However, this relaxation may yield a result that doesn't satisfy the original constraints. To alleviate this sce-

nario, we found use of two penalty functions to help:

$$H(\boldsymbol{A}) = -\sum_{i,j} \log(\boldsymbol{A}[i,j]) \cdot \boldsymbol{A}[i,j] \tag{20}$$

$$\text{and } \|\boldsymbol{A}\|_* = \text{trace}\left(\sqrt{\boldsymbol{A}^\intercal \boldsymbol{A}}\right). \tag{21}$$

The first entropy term encourages elements of $\boldsymbol{A}$ to be closer to 0 or 1. The second nuclear norm encourages $\boldsymbol{A}$ to be low-rank. Empirically we find both to improve robustness to random initializations of the model parameters. We illustrate the overall algorithm in Alg. 1, where we iteratively solve the upper-level optimization via mini-batch gradient descent. In practice, we monitor the validation loss to determine convergence. Additionally, more advanced gradient based optimization algorithms, *e.g.*, Adam (Kingma & Ba, 2015), can be utilized. A natural question is how to quantitatively evaluate the recovered sharing scheme, which we discuss next.

4.3 Quantitative Evaluation of Equivariance

Prior works rely on visual inspection of the parameter sharing to assess the quality, or use the final task performance as a surrogate; both of which are not a direct comparison with the ground-truth sharing scheme. The main challenge is that an element-wise distance between $\boldsymbol{A}_{\texttt{val}}$ and $\boldsymbol{A}_{\texttt{gt}}$ is not meaningful, as $\boldsymbol{A}$ is unique up-to permutations.

Hence, we propose to use the *Partition Distance (PD)* (Gusfield, 2002) as an evaluation metric. Specifically, PD between two sharing schemes measures the number of assignments that must be changed for one sharing scheme to be identical to the other. We will show that PD is related to the symmetric difference of the equivariance groups $\mathcal{G}$ encoded by sharing scheme $\boldsymbol{A}$. We first review the definitions of cluster, partition, and partition distance (Gusfield, 2002).

Definition 1 (Cluster)**.** *Given a set $\mathcal{S}$, a cluster is a non-empty subset of $\mathcal{S}$,* i.e.*, $\mathcal{C} \subseteq \mathcal{S}$ where $\mathcal{C} \neq \emptyset$.*

Definition 2 (Partition)**.** *A partition of $\mathcal{S}$ is a set of clusters, $\mathcal{P}_\mathcal{S} = \{\mathcal{C}_i\}$, where $\mathcal{C}_j \cap \mathcal{C}_k = \emptyset \;\; \forall j \neq k$ and $\bigcup_{i=1}^{|\mathcal{P}_\mathcal{S}|} \mathcal{C}_i = \mathcal{S}$. In other words, the elements in $\mathcal{S}$ are "partitioned" into mutually exclusive sets.*

Definition 3 (Partition Distance)**.** *Given two partitions of a set, the partition distance is the number of elements that must be moved between clusters such that the two partitions are identical.*

Next, consider the Gaussian with shared means problem in Sec. 4.1 where we model the mean as $\boldsymbol{\theta} = \boldsymbol{A}\boldsymbol{\psi}$. Here, $\boldsymbol{\theta}, \boldsymbol{\psi} \in \mathbb{R}^K$, $\boldsymbol{A} \in \{0,1\}^{K \times K}$, and $\boldsymbol{A}$ is row stochastic, *i.e.*, $\forall i \; \sum_j \boldsymbol{A}_{ij} = 1$. A parameter-sharing scheme $\boldsymbol{A}$ can be viewed as a partition over the set of model parameters. As $\boldsymbol{A}$ is integral row stochastic, it forms a partition of mutually exclusive clusters.

Given two parameter-sharing schemes $\boldsymbol{A}^{(1)}$ and $\boldsymbol{A}^{(2)}$ the parition distance $PD(\boldsymbol{A}^{(1)}, \boldsymbol{A}^{(2)})$ can be efficiently computed, in polynomial time, as proposed by Gusfield (2002). We will next explain how this distance relates to equivariance.

For a sharing scheme $\boldsymbol{A}^{(i)}$, we first construct sets to form a partition $\mathcal{P}_K^{(i)}$ consisting of clusters $\mathcal{C}_k^{(i)}$ indicating indices of shared parameters, *i.e.*,

$$\mathcal{P}_K^{(i)} = \{\mathcal{C}_k^{(i)} | \; \forall k \in [1, \dots, K]\} \text{ and} \tag{22}$$

$$\mathcal{C}_k^{(i)} = \{j | \boldsymbol{A}^{(i)}[j,k] = 1\}. \tag{23}$$

Due to the shared parameters, the indices within a cluster can be permuted. *I.e.*, given $\boldsymbol{A}^{(i)}$, the model is $\mathcal{G}_{K,K}^{(i)}$-equivariant, where

$$\mathcal{G}_{K,K}^{(i)} = \bigcup_{\mathcal{C}_k^{(i)} \in \mathcal{P}_k^{(i)}} \Pi_{\mathcal{C}_k^{(i)}}. \tag{24}$$

We use $\Pi_{\mathcal{C}_k^{(i)}}$ to denote the set of all possible permutation matrices over the indices in $\mathcal{C}_k^{(i)}$, while holding the other indices fixed.

For an effective evaluation metric, it should capture the similarity between two equivariances, $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(2)}$, each encoded by respective sharing schemes, $\boldsymbol{A}^{(1)}$ and $\boldsymbol{A}^{(2)}$. We consider the symmetric difference of two sets to quantify the similarity between $\mathcal{G}$-equivariances, *i.e.*,

$$\mathcal{G}^{(1)} \Delta \mathcal{G}^{(2)} = (\mathcal{G}^{(1)} - \mathcal{G}^{(2)}) \cup (\mathcal{G}^{(2)} - \mathcal{G}^{(1)}). \tag{25}$$

This captures the non-overlapping elements within groups. We now relate this quantity to the PD.

Claim 3. *Considering the Gaussian sharing setup,*

$$\kappa PD(\boldsymbol{A}^{(1)}, \boldsymbol{A}^{(2)}) \geq |\mathcal{G}_{K,K}^{(1)} \Delta \mathcal{G}_{K,K}^{(2)}| \geq PD(\boldsymbol{A}^{(1)}, \boldsymbol{A}^{(2)}),$$

where $\kappa = K! - 1$ is a constant and K is the number of dimensions.

Proof. Deferred to Sec. C in the Appendix. □

From Claim 3's upperbound,

$$PD(\boldsymbol{A}_1, \boldsymbol{A}_2) = 0 \implies |\mathcal{G}_{N,N}^{(1)} \Delta \mathcal{G}_{N,N}^{(2)}| = 0. \tag{26}$$

Hence, when a partition distance is zero, the two sharing schemes achieve the same equivariance. Next, as the symmetric difference is lower bounded by the partition distance, when the PD is non-zero, then the two equivariances are not identical. Base on these properties and its efficiency to compute, we find partition distance to be a suitable evaluation metric.

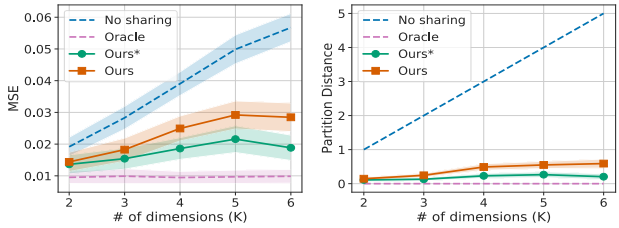

Figure 2: MSE/PD *vs.* # of dimensions.

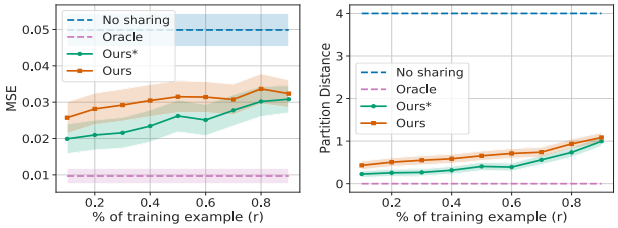

Figure 3: MSE/PD *vs.* % of training data.

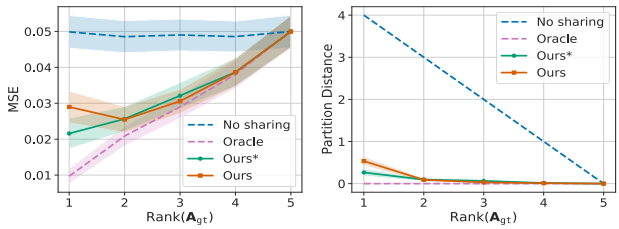

Figure 4: MSE/PD *vs.* $\text{rk}(\boldsymbol{A}_{\text{gt}})$.

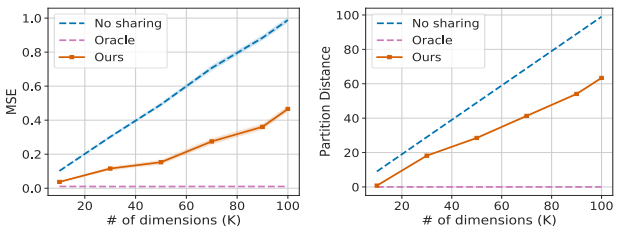

Figure 5: MSE/PD *vs.* # of dimensions.

5 EXPERIMENTS

We first conduct experiments on Gaussian data with random sharing as analyzed in Sec. 4.1. Next, we study recovery of known equivariances including permutation invariance and shift equivariance. Additional experimental details and results are in the Appendix.

5.1 Gaussian Data with Shared Means

Task, Data and Metrics. We study the same task of mean estimation that we analyzed in Sec. 4.1. We generate datasets of Gaussian vectors, following Eq. (7). We consider two evaluation metrics: 1) MSE which quantifies the performance of the mean estimation; 2) The *Partition Distance (PD)*, discussed in Sec. 4.3, which quantifies the accuracy of the recovered sharing-scheme compared to $\boldsymbol{A}_{\text{gt}}$.

Baselines. We consider `No sharing` and `Oracle`. We fix $\boldsymbol{A}$ to be the identity matrix for `No sharing`. For `Oracle`, we directly use $\boldsymbol{A}_{\text{gt}}$. Both of these methods are trained on the entire dataset $\mathcal{D}$. We also compare to `Ours*` which uses brute-force to exactly solve the bi-level optimization in Eq. (6).

Results. We conduct empirical studies over the dimension K, the ratio r between sizes of training and overall dataset, the amount of sharing $\text{rk}(\boldsymbol{A}_{\text{gt}})$, and lastly scalability of our approach when K is large. We report the mean and shade the 95% confidence interval computed from 200 runs for all experiments.

First, we empirically study the method's performance over the number of dimensions with data generated for $\text{rk}(\boldsymbol{A}_{\text{gt}}) = 1$. In Fig. 2, we report both the MSE and Partition distance for each of the baselines. We observe that the approach consistently outperforms `No sharing` across different number of dimensions in both evaluation metrics. When comparing `Ours*` to `Ours`, solving the optimization via brute-force achieves the best result, while the relaxed optimization, `Ours`, remains competitive.

Second, we evaluate the effect of adjusting the train/val split in Fig. 3. In this experiment $\text{rk}(\boldsymbol{A}_{\text{gt}}) = 1$. We observe that the performance decreases as the percentage of training examples increases. This is consistent with the upper bound in Eq. (15) and the intuition that more validation data should be used when there is more sharing.

Third, we study the effect of $\text{rk}(\boldsymbol{A}_{\text{gt}})$. Results are shown in Fig. 4. In this case, the data consists of 5 dimensions with varying $\text{rk}(\boldsymbol{A}_{\text{gt}})$ from 1 to 5. We observe that `Ours*` and `Ours` outperform `No sharing` in both metrics across the ranks.

Lastly, we demonstrate that the approach scales to higher dimensions. In Fig. 5, we increase the dimensions from 10 to 100 with $\text{rk}(\boldsymbol{A}_{\text{gt}}) = 1$. In these cases, brute-force optimization, *i.e.*, `Ours*`, is no longer possible. Still, the relaxed optimization `Ours` consistently outperforms the `No sharing` baseline across dimensions.

Ablation Study on Proposed Penalty Functions. We perform an ablation study on the two regularization/penalty terms using Gaussian data with, the same setup reported in Fig. 2. The results are summarized in Tab. 1: both of the introduced penalty terms improve the performance in both MSE and PD.

Validating Theory on Gaussian Data. Here, we generate data with $\text{rk}(\boldsymbol{A}_{\text{gt}}) = 4$ to demonstrate the trade-off between the size of training and validation sets. From Fig. 6, we can empirically observe the trade-

Table 1: Ablation study on the proposed penalty terms on Gaussian data.

# dim	Entropy	Rank	MSE	PD
2	✗	✗	$0.019 \pm .003$	$.915 \pm 0.03$
2	✓	✗	$0.016 \pm .003$	$.385 \pm 0.07$
2	✓	✓	$0.014 \pm .003$	$.145 \pm 0.05$
4	✗	✗	$0.035 \pm .003$	$2.05 \pm .069$
4	✓	✗	$0.033 \pm .003$	$1.68 \pm .070$
4	✓	✓	$0.025 \pm .003$	$0.49 \pm .095$
6	✗	✗	$0.050 \pm .004$	$3.35 \pm .075$
6	✓	✗	$0.048 \pm .004$	$3.36 \pm .094$
6	✓	✓	$0.028 \pm .004$	$0.59 \pm .132$

off characterized by the theoretical upper-bound. We also observe the same trade-off in terms of partition distance (PD), which further suggests that PD is a suitable evaluation metric.

5.2 Recovering Permutation Invariance

Tasks, Data and Metrics. Following Zaheer et al. (2017), the task is to regress to the sum of a sequence of numbers provided in text format. *E.g.*, given the input ("one," "five") the model should output 6. We also consider a variant of this task, where the "even position" (zero indexing) numbers are negated, *i.e.*, an input of ("one," "five") results in 4. Note that this latter task is only permutation invariant within the even/odd positions. The numbers are uniformly sampled from the interval $[1, 10]$ and the labels contain additive noise uniformly sampled from $[-0.5, 0.5]$.

For evaluation, we report the average squared difference (ℓ_2-loss) between prediction and ground-truth for each of the models. As in Sec. 5.1, we also report the *partition distance* to evaluate the quality of the recovered $\boldsymbol{A}_{\texttt{val}}$.

Baselines. We consider baselines of `No sharing`, `Oracle`, and `Augerino` (Benton et al., 2020). For `No sharing`, the model parameters are independent. For `Oracle`, in the standard sum of numbers, the model parameters are shared across all number positions. In the variant which negates some numbers, the model parameters are shared only across even/odd positions respectively. For `Augerino`, we parameterize the permutation transformations using the Gumbel-Sinkhorn method (Mena et al., 2018) to enable the training of augmentation parameters. We sample three transformations during both train and test.

Results. We report quantitative results across sequence length for standard sum of numbers and negated sum of numbers in Fig. 7. All results are averaged over 5 runs using different random seeds to generate the data. We report the mean and 95% confidence interval.

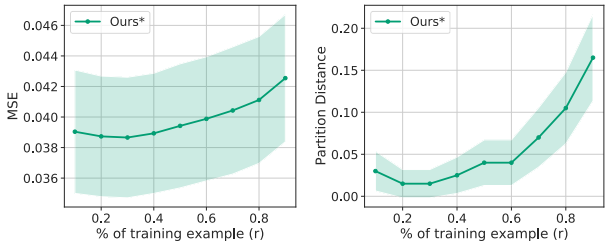

Figure 6: MSE/PD *vs.* % of training data on data with $\text{rk}(\boldsymbol{A}_{\texttt{gt}}) = 4$.

In the standard sum of numbers, `Ours` outperforms the `No sharing` baseline and `Augerino`, and performs on par with `Oracle` in terms of ℓ_2-loss. Next, looking at the partition distance, `Ours` successfully recovers permutation invariance for sequence length of two and four. While the partition distance increases for longer sequences, the model did learn partial permutation invariance among the number positions.

Next, for the negated sum of number variant, `Augerino` did not learn a competitive model. The main challenge: the probability of sampling a permutation matrix that leads to the correct invariance is low. The model seems sensitive to Gumbel Sinkhorn's temperature term. In contrast, `Ours` outperforms `No sharing` and is competitive to `Oracle`.

5.3 Recovering Shift Equivariance

Task, Data and Metrics. The task is to regress to the output of the cross-correlation operation with additive Gaussian noise, *i.e.*,

$$\mathbf{y}[k] = \epsilon + \sum_{j=0}^{G-1} \mathbf{x}[k+j]\mathbf{g}[j], \tag{27}$$

where $\epsilon \sim \mathcal{N}(0, 0.1)$ and $\mathbf{g} \in \mathbb{R}^G$ denotes a 1D kernel. We sample the input $\mathbf{x} \in \mathbb{R}^K$ from a Gaussian distribution; noise is not added for the test set. Note that cross-correlation is equivariant to shifts and is a linear system, $\mathbf{y} = \boldsymbol{G}\mathbf{x}$, where $\boldsymbol{G}$ is a Toeplitz matrix.

We investigate whether the studied approach recovers this sharing scheme, *i.e.*, $\boldsymbol{A}_{\texttt{val}}\boldsymbol{\psi} = \text{Flatten}(\boldsymbol{G})$, where G is a Toeplitz matrix. We report the ℓ_2-*loss* between the prediction and the label. We also report *partition distance* following Sec. 5.1.

Baselines. We consider baselines: `No sharing`, `Oracle`, and `Augerino` (Benton et al., 2020). For `Augerino`, we use their augmentation over the set of shift transformations with five augmented samples.

Results. In Fig. 8, we report the ℓ_2-*loss* and partition distance for each of the models. We observe `Ours`

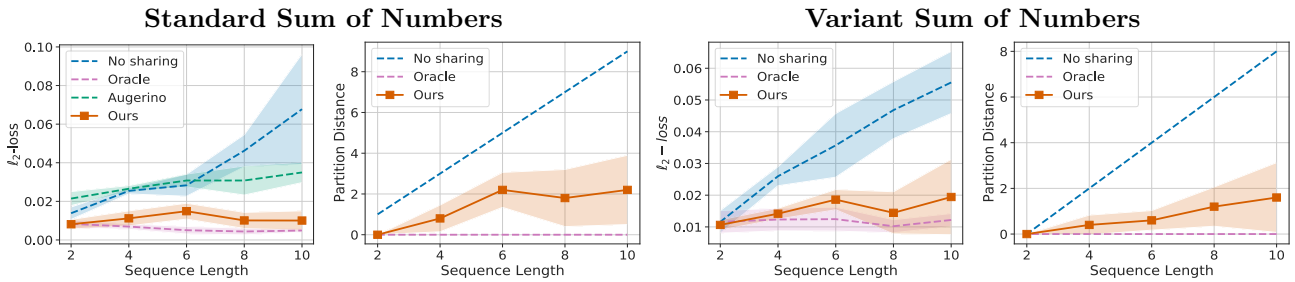

Figure 7: ℓ_2-loss and partition distance results on standard and variant sum of numbers.

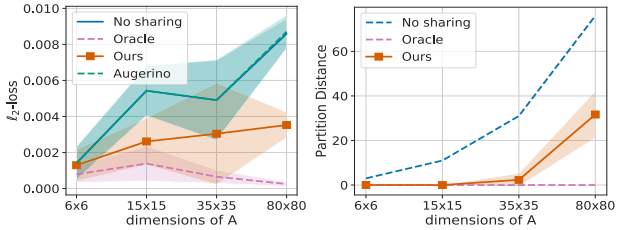

Figure 8: Results for cross-correlation.

to outperform baselines `No sharing` and `Augerino` in terms of ℓ_2-*loss*. Due to padding at the boundaries, `Augerino` learns not to shift the data, hence the performance is similar to `No sharing`. Next, we observe that `Ours` can fully recover the sharing scheme for shift equivariance when the dimension of $\boldsymbol{A}$ is 6×6 and 15×15, achieving a partition distance of 0. For a larger $\boldsymbol{A}$, *e.g.*, 35×35 and 80×80, recovering the sharing scheme is much more challenging. In this case, `Ours` can partially recover the sharing scheme.

6 CONCLUSION

We cast the process of equivariance discovery as an optimization over the parameter-sharing schemes. We analyze the proposed method using Gaussian data and provide a bound on the MSE gap. We illustrate that the approach can lead to better generalization than standard maximum likelihood training. We discuss practical considerations useful for solving the proposed optimization. We also propose to use the partition distance (PD) to quantitatively evaluate the recovered sharing schemes and show how PD is related to the symmetric difference between two equivariant groups. Through experiments, we demonstrate that the approach can recover known equivariance properties and PD is a useful evaluation metric.

Limitations. Our theoretical analysis assumes Gaussian distributions which may not be met in practice. Also, the approach considers discrete group actions. However, we think the formulation and study pave the way to future research in equivariance discovery.

Acknowledgments

We thank NVIDIA for providing GPUs used for this work. This work was supported in part by NSF under Grant #1718221, 2008387, 2045586, 2106825, MRI #1725729, NIFA award 2020-67021-32799 and Cisco Systems Inc. (Gift Award CG 1377144 - thanks for access to Arcetri). RY is supported by a Google PhD Fellowship.

References

Afendras, G. and Markatou, M. Optimality of training/test size and resampling effectiveness in cross-validation. *Journal of Statistical Planning and Inference*, 199:286–301, 2019.

Amari, S., Murata, N., Müller, K.-R., Finke, M., and Yang, H. H. Asymptotic statistical theory of overtraining and cross-validation. *IEEE Transactions on Neural Networks*, 1997.

Bengio, Y. Gradient-based optimization of hyperparameters. *Neural Computation*, 2000.

Benton, G., Finzi, M., Izmailov, P., and Wilson, A. G. Learning invariances in neural networks. In *Proc. NeurIPS*, 2020.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. Geometric deep learning: going beyond euclidean data. *IEEE Signal Process. Mag.*, 2017.

Cohen, T. and Welling, M. Group equivariant convolutional networks. In *Proc. ICML*, 2016.

Cohen, T. S., Geiger, M., Köhler, J., and Welling, M. Spherical CNNs. In *Proc. ICLR*, 2018.

de Haan, P., Weiler, M., Cohen, T., and Welling, M. Gauge equivariant mesh CNNs: Anisotropic convolutions on geometric graphs. In *Proc. ICLR*, 2021.

Defferrard, M., Bresson, X., and Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proc. NeurIPS*, 2016.

Fuchs, F., Worrall, D., Fischer, V., and Welling, M. SE (3)-Transformers: 3d roto-translation equivariant attention networks. *Proc. NeurIPS*, 2020.

Gusfield, D. Partition-distance: A problem and class of perfect graphs arising in clustering. *Information Processing Letters*, 2002.

Guyon, I., Makhoul, J., Schwartz, R., and Vapnik, V. What size test set gives good error rate estimates? *IEEE Trans. Pattern Anal. Mach. Intell*, 1998.

Guyon, I. et al. A scaling law for the validation-set training-set size ratio. *AT&T Bell Laboratories*, 1997.

Hartford, J., Graham, D., Leyton-Brown, K., and Ravanbakhsh, S. Deep models of interactions across sets. In *Proc. ICML*, 2018.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 1997.

Kearns, M. A bound on the error of cross validation using the approximation and estimation rates, with consequences for the training-test split. *Proc. NeurIPS*, 1996.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *Proc. ICLR*, 2015.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *Proc. ICLR*, 2017.

Kondor, R., Lin, Z., and Trivedi, S. Clebsch–Gordan Nets: a fully Fourier space spherical convolutional neural network. In *Proc. NeurIPS*, 2018.

Laptev, D., Savinov, N., Buhmann, J. M., and Pollefeys, M. TI-POOLING: transformation-invariant pooling for feature learning in convolutional neural networks. In *Proc. CVPR*, 2016.

Larsen, J., Hansen, L. K., Svarer, C., and Ohlsson, M. Design and regularization of neural networks: the optimal use of a validation set. In *IEEE Signal Processing Society Workshop*, 1996.

Laurent, B. and Massart, P. Adaptive estimation of a quadratic functional by model selection. *Annals of Statistics*, 2000.

LeCun, Y., Haffner, P., Bottou, L., and Bengio, Y. Object recognition with gradient-based learning. In *Shape, contour and grouping in computer vision*. 1999.

Liu*, I.-J., Yeh*, R. A., and Schwing, A. G. PIC: permutation invariant critic for multi-agent deep reinforcement learning. In *Proc. CORL*, 2019. * equal contribution.

Liu*, I.-J., Ren*, Z., Yeh*, R. A., and Schwing, A. G. Semantic tracklets: An object-centric representation for visual multi-agent reinforcement learning. In *Proc. IROS*, 2021. * equal contribution.

Lorraine, J., Vicol, P., and Duvenaud, D. Optimizing millions of hyperparameters by implicit differentiation. In *Proc. AISTATS*, 2020.

Lowe, D. G. et al. Object recognition from local scale-invariant features. In *Proc. ICCV*, 1999.

Luketina, J., Berglund, M., Greff, K., and Raiko, T. Scalable gradient-based tuning of continuous regularization hyperparameters. In *Proc. ICML*, 2016.

Maclaurin, D., Duvenaud, D., and Adams, R. Gradient-based hyperparameter optimization through reversible learning. In *Proc. ICML*, 2015.

Maron, H., Ben-Hamu, H., Shamir, N., and Lipman, Y. Invariant and equivariant graph networks. In *Proc. ICLR*, 2019.

Maron, H., Litany, O., Chechik, G., and Fetaya, E. On learning sets of symmetric elements. In *Proc. ICML*, 2020.

Mena, G., Snoek, J., Linderman, S., and Belanger, D. Learning latent permutations with Gumbel-Sinkhorn networks. In *Proc. ICLR*, 2018.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. Pointnet: Deep learning on point sets for 3D classification and segmentation. In *Proc. CVPR*, 2017.

Ravanbakhsh, S., Schneider, J., and Póczos, B. Equivariance through parameter-sharing. In *Proc. ICML*, 2017a.

Ravanbakhsh, S., Schneider, J., and Poczos, B. Deep learning with sets and point clouds. In *Proc. ICLR workshop*, 2017b.

Ren*, Z., Yeh*, R., and Schwing, A. G. Not All Unlabeled Data are Equal: Learning to Weight Data in Semi-supervised Learning. In *Proc. NeurIPS*, 2020. * equal contribution.

Romero, D. W. and Cordonnier, J.-B. Group equivariant stand-alone self-attention for vision. In *Proc. ICLR*, 2021.

Shaban, A., Cheng, C., Hatch, N., and Boots, B. Truncated back-propagation for bilevel optimization. In *Proc. AISTATS*, 2018.

Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., and Vandergheynst, P. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Process. Mag.*, 2013.

Tai, K. S., Bailis, P., and Valiant, G. Equivariant transformer networks. In *Proc. ICML*, 2019.

Vetterli, M., Kovačević, J., and Goyal, V. K. *Foundations of signal processing*. Cambridge University Press, 2014.

Wasserman, L. *All of statistics: a concise course in statistical inference*. Springer Science & Business Media, 2013.

Yeh, R. A., Hu, Y.-T., and Schwing, A. G. Chirality nets for human pose regression. In *Proc. NeurIPS*, 2019a.

Yeh, R. A., Schwing, A. G., Huang, J., and Murphy, K. Diverse generation for multi-agent sports games. In *Proc. CVPR*, 2019b.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. In *Proc. NeurIPS*, 2017.

Zhou, A., Knowles, T., and Finn, C. Meta-learning symmetries by reparameterization. In *Proc. ICLR*, 2021.

Supplementary Material: Equivariance Discovery by Learned Parameter-Sharing

This appendix is organized as follows:

- In Sec. A, we provide the full proof of Claim 1.
- In Sec. B, we provide the full proof of Claim 2.
- In Sec. C, we provide the full proof of Claim 3.
- In Sec. D, we provide an ablation study and additional experimental results.
- In Sec. E, we provide additional background and proof details.
- In Sec. F, we discuss experimental and implementation details for our empirical results.
- In Sec. G, we provide link to our code.

A Proof of Claim 1

Claim 1. *For Gaussian data (Eq. (7)) and a given sharing scheme* $\boldsymbol{A}_{\texttt{val}}$ *we have*

$$MSE(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{D})) = \left\|\boldsymbol{A}_{\texttt{val}}\bar{\boldsymbol{A}}_{\texttt{val}}^{\intercal}\boldsymbol{\theta}_{\texttt{gt}} - \boldsymbol{\theta}_{\texttt{gt}}\right\|^2 + \frac{rk(\boldsymbol{A}_{\texttt{val}})\sigma^2}{|\mathcal{D}|},$$

where $\bar{\boldsymbol{A}}_{\texttt{val}}$ *refers to the column normalized* $\boldsymbol{A}_{\texttt{val}}$ *and* $rk(\cdot)$ *denotes the rank of a matrix.*

Proof. Let S_i denote the set of indices that share parameters for the i^{th} dimension of $\hat{\boldsymbol{\theta}}_{\texttt{val}}$ as characterized in $\boldsymbol{A}_{\texttt{val}}$. Formally, $S_i \triangleq \{k \in \{1,\dots,K\} |\ \forall j\ \boldsymbol{A}_{\texttt{val}}[i,j] = 1 \wedge \boldsymbol{A}_{\texttt{val}}[k,j] = 1\}$. As the dimensions and samples are independent, the maximum likelihood estimator is the average over the shared dimensions and samples, *i.e.*,

$$\mathbb{E}(\hat{\boldsymbol{\theta}}_{\texttt{val}}[i]) = \frac{1}{|S_i|}\sum_{k\in S_i}\boldsymbol{\theta}_{\texttt{gt}}[k], \tag{9}$$

Similarly, the variance is

$$\mathbb{V}(\hat{\boldsymbol{\theta}}_{\texttt{val}}[i]) = \frac{\sigma^2}{|S_i||\mathcal{D}|}. \tag{10}$$

Now substitute these into the MSE definition

$$\text{MSE}(\hat{\boldsymbol{\theta}}(\mathcal{D})) \triangleq \mathbb{E}\left\|\hat{\boldsymbol{\theta}}(\mathcal{D}) - \boldsymbol{\theta}_{\texttt{gt}}\right\|^2 = \left\|\text{Bias}(\hat{\boldsymbol{\theta}}(\mathcal{D}))\right\|^2 + \text{Trace}(\mathbb{V}(\hat{\boldsymbol{\theta}}(\mathcal{D}))).$$

For the bias term, we can verify that $\boldsymbol{A}\bar{\boldsymbol{A}}_{\texttt{val}}^{\intercal}\boldsymbol{\theta}_{\texttt{gt}}[k] = \frac{1}{|S_i|}\sum_{k\in S_i}\boldsymbol{\theta}_{\texttt{gt}}[k]$. For the variance term, each independent parameter has a variance of $\frac{\sigma^2}{|S_i||\mathcal{D}|}$, in total we have $\text{rk}(\boldsymbol{A}_{\texttt{val}})$ independent parameters, hence, $\frac{\text{rk}(\boldsymbol{A}_{\texttt{val}})\sigma^2}{|\mathcal{D}|}$. □

B Proof of Claim 2

Claim 2. *Given data drawn i.i.d. following Eq. (7), with probability* $1-\alpha$ *and* $\alpha < \exp\frac{-K}{10}$*, the MSE gap in Eq. (14) is upper bounded by*

$$\sigma^2\Big(\underbrace{\frac{1-r}{r|\mathcal{D}|}\left(rk(\boldsymbol{A}_{\texttt{gt}})-1\right)}_{\textit{sharing rel. MSE gap}} - \underbrace{\frac{40\ln(\alpha)}{(1-r)|\mathcal{D}|}}_{\textit{con. rel. MSE gap}}\Big), \tag{15}$$

where $r=\frac{|\mathcal{T}|}{|\mathcal{D}|}$ *denotes the ratio between the size of training and overall dataset.*

Proof. We first decompose Eq. (14) into three parts:

$$\begin{aligned} & \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{D})) - \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{T})), && (28)\\ + \quad & \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{T})) - \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{gt}}(\mathcal{T})), && (29)\\ + \quad & \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{gt}}(\mathcal{T})) - \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{gt}}(\mathcal{D})). && (30)\end{aligned}$$

We then prove the claim by upper bounding Eq. (28), Eq. (29), and Eq. (30).

Bounding Eq. (28). Substituting results from Claim 1,

$$\begin{aligned} & \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{D})) - \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{T})) && (31)\\ &= \frac{\mathrm{rk}(\boldsymbol{A}_{\texttt{val}})\sigma^2}{|\mathcal{D}|} - \frac{\mathrm{rk}(\boldsymbol{A}_{\texttt{val}})\sigma^2}{|\mathcal{T}|} && (32)\\ &= -\frac{1-r}{r|\mathcal{D}|}\mathrm{rk}(\boldsymbol{A}_{\texttt{val}})\sigma^2 \le -\frac{1-r}{r|\mathcal{D}|}\sigma^2. && (33)\end{aligned}$$

Bounding Eq. (29). We further decompose Eq. (29) into three parts via

$$\begin{aligned} & \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{T})) - \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{gt}}(\mathcal{T})) \\ &= \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{T})) - \widehat{\mathrm{MSE}}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{T})) && (34)\\ &\quad + \widehat{\mathrm{MSE}}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{T})) - \widehat{\mathrm{MSE}}(\hat{\boldsymbol{\theta}}_{\texttt{gt}}(\mathcal{T})) && (35)\\ &\quad + \widehat{\mathrm{MSE}}(\hat{\boldsymbol{\theta}}_{\texttt{gt}}(\mathcal{T})) - \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{gt}}(\mathcal{T})), && (36)\end{aligned}$$

and bound each term.

For Eq. (34): By reverse triangle inequality

$$\begin{aligned} & \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{T})) - \widehat{\mathrm{MSE}}(\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{T})) && (37)\\ = \quad & \left\|\mathbb{E}\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{T}) - \boldsymbol{\theta}_{\texttt{gt}}\right\|^2 - \left\|\mathbb{E}\hat{\boldsymbol{\theta}}_{\texttt{val}}(\mathcal{T}) - \hat{\boldsymbol{\theta}}_{\texttt{ind}}(\mathcal{V})\right\|^2 && (38)\\ \le \quad & \left\|-\boldsymbol{\theta}_{\texttt{gt}} + \hat{\boldsymbol{\theta}}_{\texttt{ind}}(\mathcal{V})\right\|^2 = \left\|\boldsymbol{\theta}_{\texttt{gt}} - \hat{\boldsymbol{\theta}}_{\texttt{ind}}(\mathcal{V})\right\|^2. && (39)\end{aligned}$$

Let $\boldsymbol{Z} = \boldsymbol{\theta}_{\texttt{gt}} - \hat{\boldsymbol{\theta}}_{\texttt{ind}}(\mathcal{V})$, then $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{0}, \frac{\sigma^2}{|\mathcal{V}|}\boldsymbol{I})$. This means

$$U = \frac{\|\boldsymbol{Z}\|^2}{\frac{\sigma^2}{\mathcal{V}}} = \frac{\sum_i^K Z_i^2}{\frac{\sigma^2}{|\mathcal{V}|}} \sim \chi^2_k. \tag{40}$$

From the tail bound of the χ^2 distribution (Laurent & Massart, 2000) and $t \ge 1$,

$$P(U \ge 2tK) \le \exp\left(-\frac{tK}{10}\right). \tag{41}$$

Therefore, with probability $1-\alpha$ where $\alpha \leq \exp(-\frac{tK}{10})$,

$$\widehat{\mathrm{MSE}}(\hat{\boldsymbol{\theta}}_{\mathtt{val}}(\mathcal{T})) - \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\mathtt{val}}(\mathcal{T})) \tag{42}$$

$$\leq \quad -20\ln(\alpha)\tfrac{\sigma^2}{|\mathcal{V}|} = -20\ln(\alpha)\tfrac{\sigma^2}{(1-r)|\mathcal{D}|}. \tag{43}$$

For Eq. (35): As $\boldsymbol{A}_{\mathtt{val}}$ is determined on the validation set, it has the smallest $\widehat{\mathrm{MSE}}$, hence

$$\widehat{\mathrm{MSE}}(\hat{\boldsymbol{\theta}}_{\mathtt{val}}(\mathcal{T})) - \widehat{\mathrm{MSE}}(\hat{\boldsymbol{\theta}}_{\mathtt{gt}}(\mathcal{T})) \leq 0. \tag{44}$$

For Eq. (36): Similar to Eq. (34) we obtain

$$\widehat{\mathrm{MSE}}(\hat{\boldsymbol{\theta}}_{\mathtt{gt}}(\mathcal{T})) - \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\mathtt{gt}}(\mathcal{T})) \tag{45}$$

$$= \quad \left\|\mathbb{E}\hat{\boldsymbol{\theta}}_{\mathtt{gt}}(\mathcal{T}) - \hat{\boldsymbol{\theta}}_{\mathtt{ind}}(\mathcal{V})\right\|^2 = \left\|\boldsymbol{\theta}_{\mathtt{gt}} - \hat{\boldsymbol{\theta}}_{\mathtt{ind}}(\mathcal{V})\right\|^2. \tag{46}$$

Bounding Eq. (30). Substituting results from Claim 1,

$$\mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\mathtt{gt}}(\mathcal{T})) - \mathrm{MSE}(\hat{\boldsymbol{\theta}}_{\mathtt{gt}}(\mathcal{D})) \tag{47}$$

$$= \quad \tfrac{\mathrm{rk}(\boldsymbol{A}_{\mathtt{gt}})\sigma^2}{|\mathcal{T}|} - \tfrac{\mathrm{rk}(\boldsymbol{A}_{\mathtt{gt}})\sigma^2}{|\mathcal{D}|} \tag{48}$$

$$= \quad \tfrac{1-r}{r|\mathcal{D}|}\mathrm{rk}(\boldsymbol{A}_{\mathtt{gt}})\sigma^2. \tag{49}$$

Summing up the individual bounds concludes the proof. □

C Proof of Claim 3

Claim 3. *Considering the Gaussian sharing setup,*

$$\kappa PD(\boldsymbol{A}^{(1)}, \boldsymbol{A}^{(2)}) \geq |\mathcal{G}^{(1)}_{K,K} \Delta \mathcal{G}^{(2)}_{K,K}| \geq PD(\boldsymbol{A}^{(1)}, \boldsymbol{A}^{(2)}),$$

where $\kappa = K! - 1$ is a constant and K is the number of dimensions.

Proof. Let $\mathcal{P}^{(1)}_K$ and $\mathcal{P}^{(2)}_K$ denote the corresponding partitions of $\boldsymbol{A}^{(1)}$ and $\boldsymbol{A}^{(2)}$. Let j be an element that must be moved in $\mathcal{P}^{(1)}_K$ to match $\mathcal{P}^{(2)}_K$. Let $\mathcal{C}^{(1)}_{j*}$ and $\mathcal{C}^{(2)}_{j*}$ refer to the cluster that contains the element j.

Lower bound. As j must be moved, this means that $\mathcal{C}^{(1)}_{j*}$ are $\mathcal{C}^{(2)}_{j*}$ are not identical, therefore, $|\Pi_{\mathcal{C}^{(1)}_{j*}} \Delta \Pi_{\mathcal{C}^{(2)}_{j*}}| \geq 1$. As $\Pi_{\mathcal{C}^{(1)}_{j*}} \subseteq \mathcal{G}^{(1)}_{K,K}$ and $\Pi_{\mathcal{C}^{(2)}_{j*}} \subseteq \mathcal{G}^{(2)}_{K,K}$, <u>for each j</u> there exist a difference of at least 1. Therefore,

$$|\mathcal{G}^{(1)}_{K,K} \Delta \mathcal{G}^{(2)}_{K,K}| \geq PD(\boldsymbol{A}^{(1)}, \boldsymbol{A}^{(2)}).$$

Upper bound. Similarly, we can upper bound $|\Pi_{\mathcal{C}^{(1)}_{j*}} \Delta \Pi_{\mathcal{C}^{(2)}_{j*}}|$ with $K!-1$. Consider the case, when $\mathcal{C}^{(1)}_{j*}$ contains all the elements and $\mathcal{C}^{(2)}_{j*}$ contains only j. As $\Pi_{\mathcal{C}^{(1)}_{j*}} \subseteq \mathcal{G}^{(1)}_{K,K}$ and $\Pi_{\mathcal{C}^{(2)}_{j*}} \subseteq \mathcal{G}^{(2)}_{K,K}$, <u>for each j</u> there exist a difference of at most $K! - 1$. Therefore,

$$(K! - 1) \cdot PD(\boldsymbol{A}^{(1)}, \boldsymbol{A}^{(2)}) \geq \mathcal{G}^{(1)}_{K,K} \Delta \mathcal{G}_{K,K}.$$

□

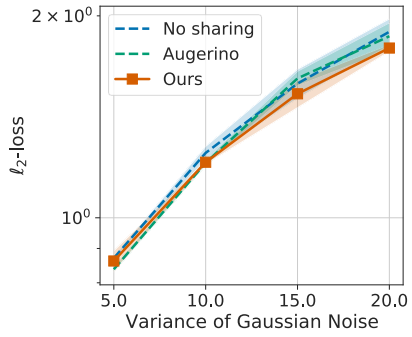

Figure A1: Quantitative results for denoising.

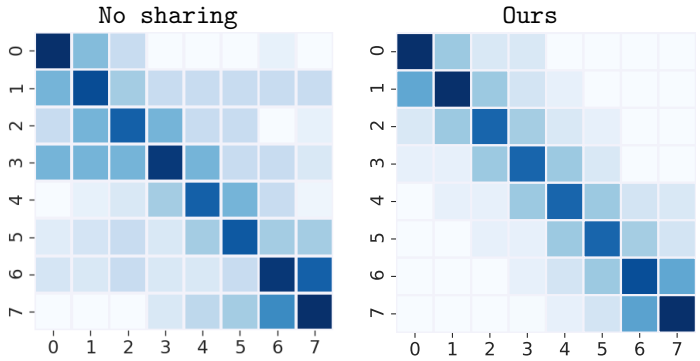

Figure A2: Visualization of the learned $\boldsymbol{G}$.

D Additional Results

D.1 Recovering Shift Equivariance from Denoising

Task, data and metrics. The task is to denoise 1D signals with additive Gaussian noise, *i.e.*,

$$\hat{\mathbf{y}} = f_\theta(\mathbf{x}), \tag{50}$$

where $\mathbf{x} \in \mathbb{R}^K$ is the noisy signal, f_θ is the denoising function and $\mathbf{y} \in \mathbb{R}^K$ is the clean signal. We denoise using a linear model, *i.e.*, $\hat{\mathbf{y}} = \boldsymbol{G}\mathbf{x}$.

We create the data by adding Gaussian noise to a randomly scaled and translated unit step signal,

$$\mathbf{x}[k] = \underbrace{s \cdot U(k-t) + b}_{\mathbf{y}[k]} + \epsilon, \tag{51}$$

where U denotes the unit step function, $s \sim \text{unif}[1, 50]$, $b \sim \text{unif}[-5, 5]$, $t \sim \text{unif}\{0, K\}$ and ϵ is zero-mean Gaussian noise. We report the mean squared error (MSE) between the prediction and the clean ground truth signal. The training set $\mathcal{T}$ consists of 50 examples, the validation set $\mathcal{V}$ consists of 100 examples, and we use 10,000 examples for testing.

Baselines. We consider two baselines: `No sharing` and `Augerino` (Benton et al., 2020). For `Augerino`, we use their augmentation over the set of shift transformations with five augmented samples, *i.e.*, during both train and test, this method requires five forward passes for each input.

Implementation details. In this experiment, we consider learning the linear system as described above and minimize the ℓ_2-*loss*. Again, the lower-level task is solved analytically. We use the Adam optimizer for the upper-level optimization with a learning rate of 0.2.

Results. In Fig. A1, we report the ℓ_2-*loss* across different amounts of added noise. This is averaged over five runs with different random seeds and we plot the mean and 95% confidence interval. We observe larger gains of `Ours` over the baselines when there is more noise to be removed. When there is less noise, `No sharing`, `Augerino` and `Ours` are comparable.

In Fig. A2, we visualize the learned $\boldsymbol{G}$ for `No sharing` and `Ours`. We observe that our approach successfully learns a Toeplitz matrix capturing the shift equivariance property of the data. This is not the case for `No sharing`.

D.2 Additional Comparison with Augerino

In our paper, we reported Augerino with three or five transformations. For completeness, we report additional experiments using more transformations. In Tab. A1 we report the training/testing time for standard sum of numbers with sequence length of 10. As can be seen, the performance of Augerino improves with increased number of transformations; similarly for the memory usage and inference time. The inference time is measured on an NVIDIA Titan X (Pascal) averaged over 100 runs.

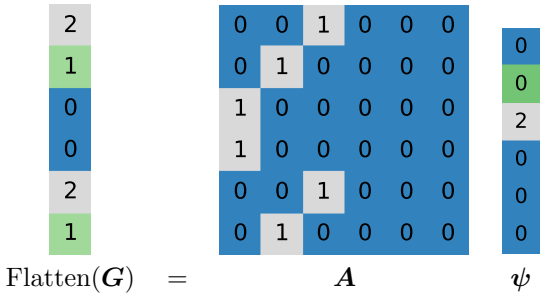

Figure A3: Parameter-sharing scheme for a cross-correlation.

Method	Number of Trans.	ℓ_2-loss	Inference time	Memory Usage
Augerino	3	0.03498 ± 0.00528	4.00 ms	727MiB
Augerino	15	0.02567 ± 0.00370	5.13 ms	1101MiB
Augerino	75	0.01966 ± 0.00503	22.0 ms	2965MiB
Ours	-	0.01005 ± 0.00503	1.73 ms	645MiB

Table A1: Comparison with Augerino over more transformations.

E Additional Background

E.1 Parameter-sharing in cross-correlation

Recall that a cross-correlation operation is defined via

$$\mathbf{y}[k] = \sum_{j=0}^{G-1} \mathbf{x}[k+j]\mathbf{g}[j]. \tag{52}$$

We can write this as a linear system $\mathbf{y} = \boldsymbol{G}\mathbf{x}$, where $\boldsymbol{G}$ is a Toeplitz matrix. To build some intuition on the parameter sharing scheme, let's consider an input $\mathbf{x} \in \mathbb{R}^3$ and $\mathbf{g} = [2, 1]$. In this case $\boldsymbol{G}$ takes the following form,

$$\boldsymbol{G} = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 1 \end{bmatrix}.$$

Observe that the parameters are shared across rows of $\boldsymbol{G}$. To capture this sharing scheme, we use an assignment matrix $\boldsymbol{A}$, *i.e.*, $\boldsymbol{G} = \boldsymbol{A}\boldsymbol{\psi}$ as illustrated in Fig. A3, where we have flattened the matrix $\boldsymbol{G}$ into a vector. Observe that $\boldsymbol{A}$ selects the parameters from $\boldsymbol{\psi}$ to form $\boldsymbol{G}$ characterizing a sharing scheme.

E.2 Miscellaneous proof details

Reverse Triangle Inequality. Let $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, then

$$\|\mathbf{x}\| - \|\mathbf{y}\| \leq \|\mathbf{x} - \mathbf{y}\|.$$

Proof. By triangle inequality,

$$\|\mathbf{x}\| = \|\mathbf{x} - \mathbf{y} + \mathbf{y}\| \leq \|\mathbf{x} - \mathbf{y}\| + \|\mathbf{y}\|.$$

□

Tail bound of χ^2 distribution. Let U be a χ^2_K random variable and $t \geq 1$ then

$$P(U \geq 2tK) \leq \exp\left(-\frac{tK}{10}\right). \tag{53}$$

Proof. From Lemma 1 of Laurent & Massart (2000), let Y_i be *i.i.d.* Gaussian variables, let a_i be non-negative, and

$$Z = \sum_{i=1}^{D} \mathbf{a}_i (Y_i^2 - 1). \tag{54}$$

Then,

$$P(Z \geq 2\left\|\mathbf{a}\right\|_2 \sqrt{x} + 2\left\|\mathbf{a}\right\|_\infty x) \leq \exp(-x). \tag{55}$$

Next, let $\mathbf{a} = [1, \ldots, 1]$, then

$$P(U \geq K + 2\sqrt{Kx} + 2x) \leq \exp(-x). \tag{56}$$

Let $x = \frac{tK}{10}$, we have

$$P(U \geq K + 2K \cdot (\sqrt{t/10} + t/10)) \leq \exp\left(-\frac{tK}{10}\right). \tag{57}$$

Lastly, we need to show

$$2tK \geq K + 2K \cdot (\sqrt{t/10} + t/10)), \tag{58}$$

$$2t - 1 \geq 2(\sqrt{t/10} + t/10). \tag{59}$$

Let $v = \sqrt{t/10}$, then we have

$$0 \geq -9v^2 + v + 0.5, \tag{60}$$

which is true when $v \geq 0.3$, *i.e.*, when $t \geq 0.9$. □

F Additional Experimental Details

We provide additional details of the experiments reported in the main paper.

F.1 Gaussian data with Shared Means

Data. We generate data following a Gaussian distribution with a shared mean as specified in Eq. (7). The dataset $\mathcal{D}$ contains 100 samples. We split $\mathcal{T}$ and $\mathcal{V}$ to contain 30 and 70 samples, except for experiment in Fig. 4 where we sweep over the different sizes of $\mathcal{T}$ and $\mathcal{V}$.

Implementation details. For this task, the lower optimization in our proposed program (Eq. (6)) can be solved analytically. To see this we write it in matrix form:

$$\min_{\boldsymbol{\psi}} \left\|\boldsymbol{X}\boldsymbol{\psi}\boldsymbol{A}^\intercal - \boldsymbol{Y}\right\|_F^2, \tag{61}$$

where $\boldsymbol{X} = \mathbf{1}_{N\times 1}$, $\boldsymbol{\psi} \in \mathbb{R}^{1\times K}$, $\boldsymbol{A} \in [0,1]^{K\times K}$ and $\boldsymbol{Y} \in \mathbb{R}^{N\times K}$. In form of ordinary least-squares,

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \left\|\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{Y}\right\|_F^2, \tag{62}$$

from which we obtain $\boldsymbol{\psi}^* = \boldsymbol{\theta}^*(\boldsymbol{A}^\intercal)^+$, where $\boldsymbol{A}^+$ denotes the pseudo-inverse. We use the Adam (Kingma & Ba, 2015) to solve the upper-level optimization.

Training details. As described in the paper, we solve the lower-level optimization analytically and use the Adam optimizer to handle the upper-level task. For the Adam optimizer, we use a learning rate of 2e−2 with weight-decay of 1e−4. We also tried lowering the learning rates to 1e−2 and 5e−3. We did not sweep over the weight-decay. These hyperparameters are used for all experiments and all models in this section. We train all models for 1000 epochs without batching.

Running experiments. We provide code to run all these experiments. Please refer to the code in folder `projects/GaussianSharing/experiments/`. We use an NVIDIA TITAN X (Pascal) to run these experiments.

F.2 Recovering Permutation Invariance

Data. For the sum of numbers dataset, we uniformly sample numbers from the set $\{1, \ldots, 10\}$ and the corresponding label is the sum of these numbers. Let $\mathbf{x}_i$ denote the i^{th} number in the sequence, then the label is defined as

$$\mathbf{y} = \sum_i \mathbf{x}_i. \tag{63}$$

Next, for the variant sum of numbers, the label is defined as

$$\mathbf{y} = \sum_i (-1)^{(i+1)} \mathbf{x}_i. \tag{64}$$

In this case, the even positioned numbers are multiplied by negative one.

For both of these experiments, we use a dataset $\mathcal{D}$ of size 250 and split it into training set $\mathcal{T}$ and validation set $\mathcal{V}$ containing 100 and 150 samples respectively. As we use deep-nets for these experiments, we created a separate set $\mathcal{H}$ of size 250 to apply early stopping and tune the learning rate. Note that all the compared methods have access to exactly the same data. Lastly, we use a test set of $100,000$ samples. For pre-processing, we standardize the label by subtracting the mean and by dividing by the standard deviation. At test-time, we scale the output back to the original range.

Model details. We use an embedding of 500 dimensions to represent the input. Next, this embedding is passed through a fully-connected layer of 50 dimensions (one per position), and a ReLU non-linearity. The baselines use this same network architecture. To output a single scalar, we sum across the position dimension and pass it through a fully-connected layer of 1 dimension. We learn how to share the model-parameters across the sequence positions at the first fully connected layer.

Training details. We solve both lower and upper-level optimization using the Adam optimizer, and compute the hypergradient using Neumann inverse approximation, with 20 iterations. The upper-level learning rate is 1e$-$2 and the lower-level learning rate is 1e$-$3. For every upper-level optimization step, we run 250 lower-level steps. For the learning rate, we studied 1e$-$1, 1e$-$2 and 1e$-$3. For the number of lower-level steps, we have assessed $50, 150, 250$ steps.

Running experiments. We provide code to run all these experiments. Please see the code in folder `projects/PermutationSharing/experiments/`. We use an NVIDIA TITAN X (Pascal) to run these experiments.

F.3 Recovering Shift Equivariance

Data. We fixed the kernel $\mathbf{g}$ to increase by two for every position, *e.g.*, a kernel with size three is $[1, 3, 5]$. The training set $\mathcal{T}$ consists of 50 examples, the validation set $\mathcal{V}$ consists of 100 examples, and we use 10,000 examples for testing.

Implementation details. In this experiment, we consider learning the linear system as described above and minimize the ℓ_2-*loss*. As in the Gaussian experiment, the lower-level task can be solved analytically. We use the Adam optimizer to address the upper-level optimization.

Training details. We solve the lower-level optimization task analytically by formulating it as an ordinary least-squares problem, *i.e.*,

$$\boldsymbol{G}^* = (\boldsymbol{X}^\intercal \boldsymbol{X})^{-1} \boldsymbol{X}^\intercal \boldsymbol{Y}, \tag{65}$$

where $\boldsymbol{X} \in \mathbb{R}^{N \times K_{\text{in}}}$ and $\boldsymbol{Y} \in \mathbb{R}^{N \times K_{\text{out}}}$. With the lower-level optimization solved, we write $\boldsymbol{\psi}^*$ as

$$\boldsymbol{\psi}^*(\boldsymbol{A}) = \boldsymbol{A}^+ \text{Flatten}(\boldsymbol{G}^*), \tag{66}$$

where $\boldsymbol{A}^+$ denotes the pseudo-inverse and flatten reshapes a matrix into a vector. For the upper-level task, we back-propagate through $\boldsymbol{\psi}^*$ to update $\boldsymbol{A}$. In this experiment, we use the Adam optimizer with a learning rate of 0.1.

Running experiments. We provide code to run all these experiments. Please see the code in folder `projects/ConvSharing/experiments/`. We use an NVIDIA GeForce GTX 1080 to run these experiments.

G Code Release

Please find the released code at https://github.com/raymondyeh07/equivariance_discovery.