서울과학고등학교 졸업논문

신경망에서 대칭성 기반 매개변수 공유의 Inductive Bias로의 작용에 대한 실증적 연구

2026년 7월

과학영재학교

서울과학고등학교

2 4 0 3 1

김 주 환

요 약

본 연구에서는 task에 내재된 대칭성을 기반으로 한 매개변수 공유가 신경망에서 inductive bias로 작용할 수 있는지를 실증적으로 분석한다. 일반적인 Multi-Layer Perceptron(MLP)은 높은 표현력을 가지지만, 입력 순서나 구조에 대한 불필요한 의존성을 학습할 수 있다는 한계를 가진다. 이러한 문제를 완화하기 위해 기존에는 data augmentation이나 구조적으로 제약된 모델을 활용하여 일반화 성능 향상을 유도해 왔다. 이에 본 연구에서는 task의 대칭성을 기반으로 일부 가중치를 강제로 공유함으로써 함수 공간을 제한하고, 이를 통해 구조적 inductive bias를 형성하는 방법을 제안한다. 해당 방법론의 효과를 검증하기 위하여 세 가지 toy task에 대해 일반 MLP, 데이터 증강 기반 접근, 그리고 대칭성 기반 매개변수 공유 모델을 비교하였으며, symmetric robustness, sample efficiency, parameter efficiency 및 일반화 성능 관점에서 분석을 수행하였다. 실험 결과, 대칭성을 반영한 매개변수 공유 모델은 적은 데이터 및 제한된 파라미터 환경에서 보다 안정적인 성능과 향상된 일반화 특성을 보였다. 본 연구는 parameter sharing이 단순한 모델 경량화 기법을 넘어, 신경망의 일반화를 유도하는 구조적 inductive bias로 작용할 수 있음을 시사한다.

주요어 : 대칭성, 파라미터 공유, inductive bias, robustness, efficiency, Neural Network
학번 : 24031

제 1장 Introduction

제 1 절 문제 상황

Multi-Layer Perceptron(MLP)은 Universal Approximation Theorem에 따라 임의의 연속 함수를 근사할 수 있으나(Hornik 1991), 이 범용성은 동시에 귀납적 편향(inductive bias)의 부재를 낳는다. Task에 부적합한 함수도 가설 공간에 포함되어 최적화가 비효율적으로 일어나며 (Battaglia et al. 2018), 특화된 아키텍처들이 더 효과적임이 밝혀지며 널리 채택되고 있는 것도 이러한 이유에서이다(Schmidhuber 2015; LeCun, Bengio, and Hinton 2015).

이러한 문제를 해결하기 위한 자연스러운 방법은 task에 내재된 대칭성을 이용해 가설 공간을 제한하는 것이다. 실제로 많은 task가 입력의 특정 변환에 대한 invariance 또는 equivariance를 내재적으로 요구한다. 집합에 대한 함수는 원소의 순서와 무관해야 하므로 permutation invariance를 가지며, image segmentation은 이미지의 평행 이동에 대해 equivariant해야 한다. 이러한 대칭성을 모델에 내재시키는 대표적인 방법은 매개변수 공유(parameter sharing)로, 특정 가중치들을 동일하게 강제함으로써 함수가 해당 대칭성을 만족하도록 한다. CNN이 공간적 위치에서 가중치를 공유함으로써 translational equivariance를 달성하는 것이 대표적인 예이다. 최근에는 이 관계를 이론적으로 형식화하는 연구가 활발히 이루어지고 있다. Ravanbakhsh 등은 permutation group에 대한 parameter sharing 구조로부터 equivariant 신경망을 구성할 수 있음을 보였으며 (Ravanbakhsh, Schneider, and Poczos 2017), Finzi 등은 이를 임의의 matrix group으로 일반화하였다(Finzi, Welling, and Wilson 2021).

제 2 절 연구 목적

그러나 핵심적인 실증적 질문은 충분히 탐구되지 않았다. 기존 연구들은 equivariant architecture의 설계나 이론적 성질 증명에 초점을 두는 경우가 많았으며, parameter sharing의 효과를 data augmentation이나 model capacity의 영향과 분리하여 분석한 연구는 상대적으로 드물다. Parameter sharing은 종종 모델 경량화 수단으로 이해되기도 하나, 그 근본적인 역할은 가설 공간을 task와 양립 가능한 대칭성 하의 함수들로 구조적으로 제한하는 inductive bias를 제공하는 것이다. 따라서 parameter sharing이 실제로 일반화 성능 향상에 기여하는지, 혹은 단순히 파라미터 수 감소에 따른 정규화 효과에 불과한지는 아직 명확하지 않다.

본 연구는 이러한 질문에 답하기 위해 대칭 구조가 명확하게 정의된 세 가지 toy task를 설계하고, 서로 다른 귀납적 편향을 갖는 네 종류의 모델을 비교한다. 비교 대상은 구조적 제약이 없는 vanilla MLP, data augmentation을 적용한 MLP, parameter sharing을 통해 대칭성이 직접 강제된 MLP, 그리고 두 방법을 모두 적용한 MLP이다. 과제 구조와 데이터셋 규모를 체계적으로 통제함으로써, parameter sharing이 일반화 성능, 데이터 효율성, 대칭성 보존에 미치는 영향을 정량적으로 분석한다. 이를 통해 parameter sharing이 단순한 파라미터 절감 기법을 넘어, 대칭성을 반영하는 효과적인 inductive bias로 작용할 수 있는지 실증적으로 검증하고자 한다.

제 2장 Theoretical Background

제 1 절 Symmetry as Inductive Bias

유한한 훈련 데이터만으로는 가능한 모든 함수의 집합 $\mathcal{F}$ 안에서 올바른 가설을 유일하게 결정할 수 없다. No Free Lunch Theorem (Wolpert 1996)이 보여주듯, 특정 task에 대한 사전 가정, 즉 inductive bias 없이는 어떤 알고리즘도 훈련 데이터 이상으로 일반화할 수 없다 (Mitchell 1980). Inductive bias는 탐색 범위를 $\mathcal{F}$의 부분집합인 가설 공간 $\mathcal{H} \subsetneq \mathcal{F}$로 제한함으로써 작용하며, 핵심적인 문제는 편향의 유무가 아니라 *어떤* 편향을 채택하느냐이다.

본 연구에서 주목하는 inductive bias는 task에 내재된 *대칭성*이다. 많은 task는 그 정의로부터 입력의 특정 변환에 대해 출력이 일정한 방식으로 유지되는 성질을 요구한다. 이를 형식화하기 위해, 집합 G와 이항 연산 $\cdot$의 쌍 $(G, \cdot)$이 결합법칙, 항등원, 역원의 존재성을 만족할 때 이를 군(Group)이라 한다. 또한 조건 $g_1 \cdot (g_2 \cdot x) = (g_1 \cdot g_2) \cdot x$, $e \cdot x = x$를 만족하는 사상 $G \times \mathcal{X} \to \mathcal{X}$를 군 G의 $\mathcal{X}$ 위에서의 작용(Group Action)이라 하며, g의 x에 대한 작용을 $g \cdot x$로 표기한다.

군 G가 입력 공간 $\mathcal{X}$와 출력 공간 $\mathcal{Y}$ 위에서 각각 작용할 때, 함수 $f : \mathcal{X} \to \mathcal{Y}$의 대칭성은 다음과 같이 정의된다.

Definition 1 (Equivariance / Invariance)**.** 함수 $f : \mathcal{X} \to \mathcal{Y}$가 군 G에 대해 **equivariant**하다는 것은 다음을 만족함을 의미한다 (Kondor and Trivedi 2018).

$$f(g \cdot x) = g \cdot f(x), \quad \forall g \in G,\ \forall x \in \mathcal{X} \tag{1}$$

출력 공간에서의 군 작용이 항등 변환인 특수한 경우, 즉 $f(g \cdot x) = f(x)$가 성립할 때 f를 G에 대해 **invariant**하다고 한다. Invariance는 Equivariance의 특별한 경우이다.

많은 task가 이러한 대칭성을 내재적으로 요구한다. 집합에 대한 함수는 원소의 순서와 무관해야 하므로 symmetric group S_n에 대해 invariant해야 하며, image segmentation은 이미지가 평행 이동하면 segmentation 결과도 동일하게 이동해야 하므로 평행 이동군에 대해 equivariant해야 한다. CNN이 translational equivariance를 (Cohen and Welling 2016), Transformer가 특정 군에 대한 equivariance를 구현한 것으로 해석될 수 있는 것도 (Bronstein et al. 2021) 이러한 맥락에서이다. 오직 해당 대칭성을 만족하는 함수로만 $\mathcal{H}$를 제한하는 것이 task에 부합하는 자연스러운 inductive bias이며, 본 연구는 이를 MLP에서 parameter sharing을 통해 구현하는 방법을 다음 절에서 형식화한다.

제 2 절 Equivariant MLP via Parameter Sharing

군 G의 표현(Representation)이란 G에서 가역 선형 변환들의 군 $GL(V)$로의 준동형사상 $\rho : G \to GL(V)$이다 (Serre et al. 1977). 표현을 이용하면 군의 작용을 행렬의 언어로 다룰 수 있으며, 이를 통해 equivariance 조건을 선형 레이어 수준에서 명시적으로 유도할 수 있다.

선형 레이어의 Equivariance 조건

입력 공간 $\mathbb{R}^n$과 출력 공간 $\mathbb{R}^m$ 위에서 군 G가 각각 표현 $\rho_{\text{in}} : G \to GL(\mathbb{R}^n)$과 $\rho_{\text{out}} : G \to GL(\mathbb{R}^m)$으로 작용한다고 하자. 선형 레이어 $f(x) = Wx$가 G에 대해 equivariant하려면, 식 (1)에 $f(x) = Wx$를

대입하고 임의의 x에 대해 성립해야 한다는 조건으로부터 다음을 얻는다.

$$W\rho_{\text{in}}(g) = \rho_{\text{out}}(g)\,W, \quad \forall g \in G \tag{2}$$

이 조건을 *Equivariance 조건*이라 하며, 이를 만족하는 가중치 행렬의 집합을 다음과 같이 정의한다.

$$\mathcal{W}_G = \left\{ W \in \mathbb{R}^{m\times n} \;\middle|\; W\rho_{\text{in}}(g) = \rho_{\text{out}}(g)\,W,\ \forall g \in G \right\} \tag{3}$$

Parameter Sharing에 의한 $\mathcal{W}_G$의 구현

Equivariance 조건은 W에 대한 선형 동차 방정식이므로, $\mathcal{W}_G$는 $\mathbb{R}^{m\times n}$의 선형 부분공간을 이룬다. ρ_{in}, ρ_{out}이 치환 표현인 경우, Appendix 1의 증명 (Ravanbakhsh, Schneider, and Poczos 2017)에 의해 $\mathcal{W}_G$의 기저는 인덱스 쌍 (i,j)에 대한 G-작용의 동치류 $\{O_1,\dots,O_k\}$에 대응하는 행렬

$$B_l = \sum_{(i,j)\in O_l} E_{ij}, \quad l = 1,\dots,k \tag{4}$$

로 구성된다. 동치류들이 $\{1,\dots,m\}\times\{1,\dots,n\}$를 서로소로 분할하므로, 임의의 $W \in \mathcal{W}_G$는

$$W = \sum_{l=1}^{k} w_l B_l, \quad w_l \in \mathbb{R} \tag{5}$$

로 유일하게 표현된다. 즉, 같은 동치류에 속하는 위치의 가중치는 하나의 자유 파라미터를 공유하며, 이것이 곧 Parameter Sharing이다. Figure 1가 이 과정을 개괄적으로 나타낸다. 연속 군의 경우 (Finzi, Welling, and Wilson 2021)은 동치류 열거 대신 Lie algebra 생성원으로부터 유도되는 선형 시스템을 수치적으로 풀어 동일한 구성을 일반화한다.

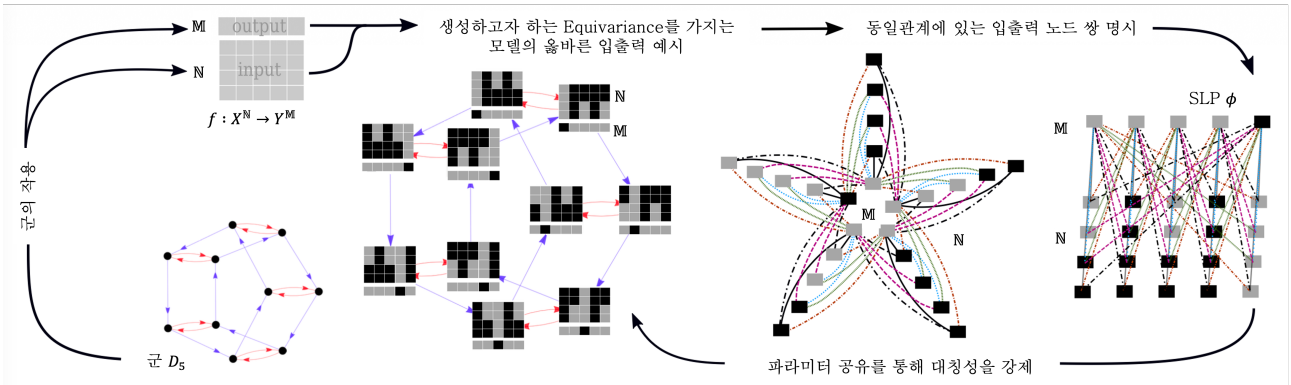

Figure 1: 파라미터 공유를 통한 Equivariance 보장의 개요. 군 G의 작용으로부터 equivariance 조건을 도출하고, 동일한 궤도 관계에 있는 입출력 노드 쌍의 가중치를 공유함으로써 G에 대해 equivariant한 SLP(Single Layer Perceptron)를 구성한다.
(좌측) $G = D_5$, 입력 4×5 이미지, 출력 크기 5의 벡터.
(좌측 중앙) G 작용에 따른 입력 변화.
(우측 중앙) 동일 궤도 관계에 있는 입출력 쌍의 그래프 표현.
(우측) 가중치 공유 관계를 명시한 이분 그래프 형태의 SLP.
허가를 받아 재인용한 (Ravanbakhsh, Schneider, and Poczos 2017)의 Figure이다.

Equivariant MLP 구성 및 가설 공간의 제한

단일 선형 레이어의 equivariance를 MLP 전체로 확장하는 것은 자명하다. Elementwise 활성화 함수 σ는 $\rho_{\text{out}}(g)$가 치환 행렬일 때 $\sigma(\rho_{\text{out}}(g)\, z) = \rho_{\text{out}}(g)\, \sigma(z)$를 만족하므로, equivariant한 선형 레이어와 elementwise 활성화 함수로 구성된 각 레이어 ϕ_l이 G에 대해 equivariant하면, 이들의 합성 $\Phi = \phi_L \circ \cdots \circ \phi_1$ 역시 equivariant하다 (Ravanbakhsh, Schneider, and Poczos 2017). Invariant MLP는 equivariant MLP의 출력에 합(sum), 평균(mean) 등의 permutation-invariant 집계 함수 Agg를 적용하여 구성한다 (Zaheer et al. 2017).

$$f_{\text{inv}}(x) = \text{Agg}(\Phi(x)) \tag{6}$$

결과적으로 $\mathcal{W}_G$는 $\mathbb{R}^{m \times n}$의 진부분공간이며, 일반적으로 $\dim(\mathcal{W}_G) = k < mn$이 성립한다. Parameter Sharing이 적용된 MLP의 가설 공간

$$\mathcal{H}_G = \{f_W \in \mathcal{H} \mid W \in \mathcal{W}_G\} \subsetneq \mathcal{H} \tag{7}$$

는 대칭성 조건을 만족하는 함수들로 구조적으로 제한되며, 이것이 곧 parameter sharing이 제공하는 inductive bias이다.

제 3장 Experimental Design

제 1 절 Experimental Goals and Comparison Setup

본 실험은 parameter sharing이 제공하는 symmetry-based inductive bias가 신경망의 일반화 성능에 미치는 영향을 실증적으로 검증하는 것을 목적으로 한다. 구체적으로 다음 세 가지 질문에 답하고자 한다. 첫째, symmetry를 모델 구조에 직접 반영한 parameter sharing이 일반화 성능 향상에 기여하는가. 둘째, 이러한 효과는 data augmentation과 어떠한 차이를 보이는가. 셋째, parameter sharing의 효과는 이상화된 문제에 국한되는가, 아니면 보다 복잡하고 현실적인 문제에서도 유지되는가.

Task selection

세 가지 질문에 답하기 위해 구조적 복잡성과 현실성이 서로 다른 세 task를 선정하였다. 세 task는 모두 permutation symmetry를 포함하되, symmetry가 문제를 규정하는 정도와 현실적 일반성을 달리 설계하였다.
Task 1은 permutation invariant set function 문제이다. 입력 원소의 순서가 본질적으로 의미를 가지지 않으므로 symmetry가 문제의 정의 자체를 직접 규정한다. Symmetry-based inductive bias의 효과를 가장 순수한 형태로 관찰하기 위한 기준 실험에 해당한다.
Task 2는 graph classification 문제이다. 노드의 번호가 임의적으로 부여되므로 node relabeling에 대한 invariance가 요구되지만, 단순한 집합 함수와 달리 graph structure에 의해 정의되는 relational information을 포함한다. Symmetry의 효과가 보다 복합적인 구조 속에서 나타나는 중간 단계의 문제이다.
Task 3은 대칭성을 가지는 물리 시스템의 상태 전이 예측 문제이다. 물리 법칙은 입자의 순서에 의존하지 않으므로 symmetry가 중요한 제약으로 작용하지만, 실제 예측 성능은 동역학적 상호작용, 비선형성 등 다양한 요소에 의해 함께 결정된다. 현실적 적용 가능성과 과학적 일반성이 가장 큰 문제이다.
Task 1에서 Task 3으로 갈수록 symmetry가 문제를 규정하는 정도는 감소하고 현실성은 증가한다. 이러한 배열을 통해 parameter sharing의 효과가 단순한 toy problem에 국한되는지, 보다 복잡한 문제에서도 일관된 inductive bias로 기능하는지를 비교한다.

Comparison Setup

모든 task에서 동일한 비교 구도를 유지하였다. 비교 대상은 다음 네 모델이다.

- **Vanilla MLP**: 구조적 제약이 없는 기준선
- **MLP + Aug**: data augmentation으로 symmetry를 경험적으로 학습하게 하는 접근
- **MLP + Share**: parameter sharing으로 symmetry를 모델 구조에 직접 강제하는 접근
- **MLP + Aug + Share**: 두 방법을 동시에 적용한 모델

이 구성을 통해 symmetry를 반영하는 방식의 차이, 즉 데이터 수준에서의 유도와 구조 수준에서의 강제가 일반화 성능에 미치는 영향을 분리하여 분석한다.

제 2 절 Evaluation Metrics

Parameter sharing의 효과를 다면적으로 분석하기 위해 네 가지 지표를 사용한다.

Generalization Performance

회귀 문제의 경우 test loss를, 분류 문제의 경우 test accuracy와 test loss를 함께 측정한다. Parameter sharing이 효과적인 inductive bias로 작용한다면, 동일하거나 더 적은 수의 parameter로도 test set에서 더 안정적인 성능을 보여야 한다.

Symmetry Error

본 연구에서 가장 핵심적인 지표이다. 입력에 task가 요구하는 permutation transformation을 가하였을 때, 모델 출력이 이론적으로 기대되는 방식으로 유지되는지를 측정한다. L을 오차함수라 하면 다음과 같이 정의된다.

$$\text{Symmetry Error}(x; f) = \mathbb{E}_{g \in G} \left[L(g \cdot f(x),\, f(g \cdot x)) \right] \tag{8}$$

Parameter sharing을 적용한 모델은 구조적으로 symmetry를 만족하도록 설계되므로 symmetry error가 0에 수렴해야 하며, 이 지표는 모델이 단순히 데이터를 맞추는 수준을 넘어 task의 구조 자체를 반영하는지를 평가한다.

Sample Efficiency

적절한 inductive bias는 적은 데이터로도 더 나은 일반화를 가능하게 한다. 각 task에서 training set size를 변화시키며 동일한 비교를 반복하고, 조건별 test performance 변화를 비교함으로써 sample efficiency를 평가한다.

Parameter Efficiency

Parameter sharing은 자연스럽게 trainable parameter 수의 감소를 수반한다. 각 모델의 parameter 수를 함께 기록하고, 유사하거나 더 적은 수의 parameter로 어느 정도의 generalization performance와 symmetry preservation을 달성하는지를 비교한다. 이를 통해 parameter sharing의 효과가 단순한 모델 압축인지, 구조적으로 유의미한 inductive bias인지를 구분한다.

제 3 절 Task 1: Permutation-Invariant Set Function

Task Definition

n개의 실수로 이루어진 입력 $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$으로부터 하나의 스칼라 값을 예측하는 regression problem이다. 입력 원소의 순서는 의미를 가지지 않으므로, 학습해야 하는 함수 $f : \mathbb{R}^n \to \mathbb{R}$는 임의의 $\pi \in S_n$에 대해 다음 조건을 만족해야 한다.

$$f(\pi \cdot x) = f(x), \quad \forall \pi \in S_n,\ \forall x \in \mathbb{R}^n \tag{9}$$

Parameter Sharing Construction

Appendix 2의 유도에 따라, S_n-equivariance 조건을 만족하는 선형 레이어는 hidden representation h^τ를 n_{block}개의 block으로 분할할 때 다음 형태로 유일하게 결정된다.

$$h_i^{\tau+1} = \sigma\left(Ah_i^\tau + B\left(\sum_{j\neq i} h_j^\tau\right) + b \right), \quad i = 1, \ldots, n_{\text{block}} \tag{10}$$

행렬 A는 각 block의 local transformation, B는 나머지 block들의 합에 적용되는 global transformation에 해당하며, 모든 block이 동일한 A, B, b를 공유한다. 이를 block matrix로 표기하면($n_{\text{block}} = 3$으로 가정) 다음과 같다.

$$h^{\tau+1} = \sigma\left(\begin{bmatrix} A & B & B \\ B & A & B \\ B & B & A \end{bmatrix} h^\tau \right) \tag{11}$$

최종적으로 equivariant layer를 여러 층 쌓은 뒤 block 방향 평균을 취함으로써 permutation-invariant한 출력을 얻는다. 전체 구조는 Figure 2에 나타냈다.

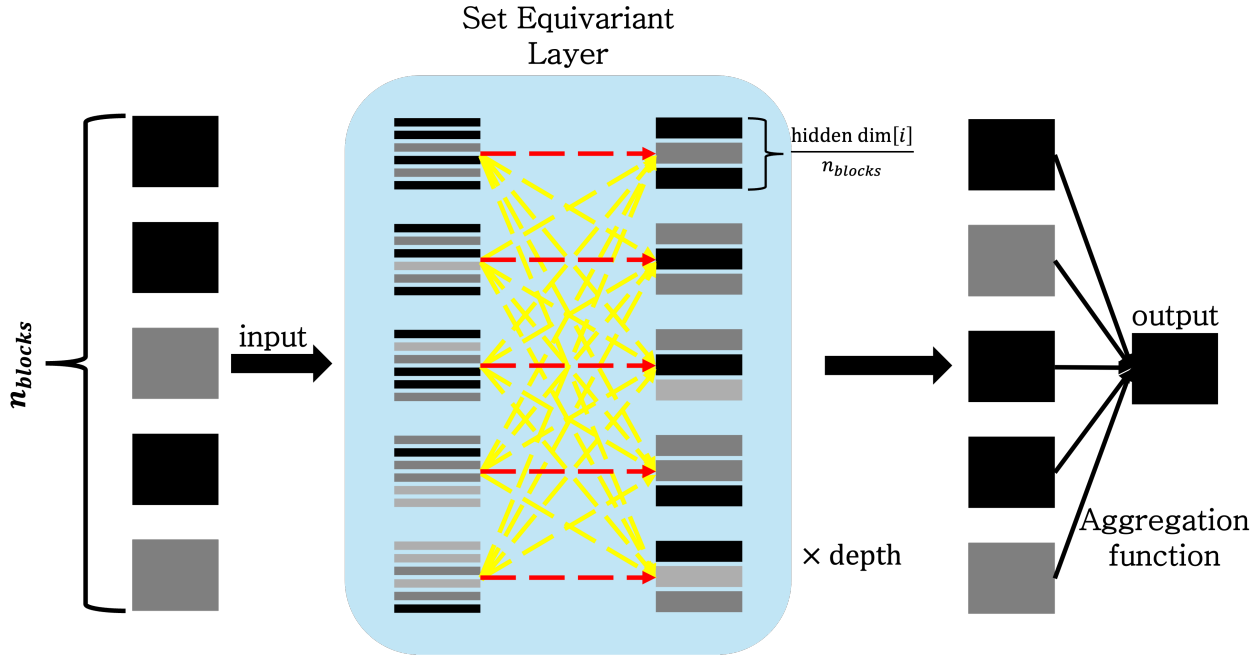

Figure 2: Task 1의 shared MLP 구조도. 같은 색의 간선은 동일한 가중치를 공유하며, block 방향 aggregation을 통해 최종적으로 invariant한 출력을 얻는다.

제 4 절 Task 2: Graph Classification

Task Definition

인접행렬 $A \in \mathbb{R}^{n\times n}$와 node feature 행렬 $X \in \mathbb{R}^{n\times d_{\text{in}}}$로부터 graph-level class label을 예측하는 함수 $f : (A, X) \mapsto y$를 학습하는 문제이다. 노드의 번호는 본질적 의미를 가지지 않으므로, 임의의 permutation

matrix P에 대해 다음 조건을 만족해야 한다.

$$f(PAP^\top, PX) = f(A, X) \tag{12}$$

최종 출력은 invariant해야 하지만 중간 representation은 각 노드에 대응되므로, 레이어 수준에서 permutation-equivariant한 변환을 구성하고 마지막에 graph-level pooling을 적용하는 방식으로 구성하는 것이 자연스럽다.

Parameter Sharing Construction

Appendix 3의 유도에 따라, node relabeling에 대해 equivariant한 선형 graph layer는 정확히 다섯 개의 basis operator의 선형결합으로 표현된다.

$$B_1 = \mathrm{diag}(A) \odot X \tag{13}$$

$$B_2 = AX - B_1 \tag{14}$$

$$B_3 = (\deg(A) - \mathrm{diag}(A)) \odot X \tag{15}$$

$$B_4 = \left(\textstyle\sum_j \mathrm{diag}(A)_j X_j\right) - B_1 \tag{16}$$

$$B_5 = (\textstyle\sum_i (AX)_i) - B_1 - B_2 - B_3 - B_4 \tag{17}$$

여기서 $\odot$는 원소별 곱, $\deg(A)$는 각 노드의 degree를 모은 벡터이며, B_4와 B_5는 전역 집계값을 각 노드로 broadcast한 항이다. 한 레이어의 출력은 다음과 같다.

$$Y = \sum_{l=1}^{5} W_l(B_l) + b \tag{18}$$

여러 equivariant layer를 통과한 node representation에 graph-level pooling을 적용하여 최종적으로 permutation-invariant한 graph representation을 얻는다. 전체 구조는 Figure 3에 나타냈다.

제 5 절 Task 3: Symmetric Physical System Prediction

Task Definition

2차원 공간에서 운동하는 세 입자의 계를 고려한다. 각 입자의 상태(위치 및 속도)를 $x_i \in \mathbb{R}^4$라 하면 전체 상태는 $x = (x_1, x_2, x_3) \in \mathbb{R}^{12}$이며, 현재 상태 x_t로부터 일정 시간 이후의 상태 $x_{t+\Delta t}$를 예측하는 함수 $f : \mathbb{R}^{12} \to \mathbb{R}^{12}$를 학습하는 문제이다.
첫 번째와 두 번째 입자는 동일한 질량을 가지므로, 생성원 g의 작용 $g \cdot (x_1, x_2, x_3) = (x_2, x_1, x_3)$으로 정의되는 S_2에 대해 다음 equivariance 조건을 만족해야 한다.

$$f(g \cdot x) = g \cdot f(x), \quad \forall x \in \mathbb{R}^{12} \tag{19}$$

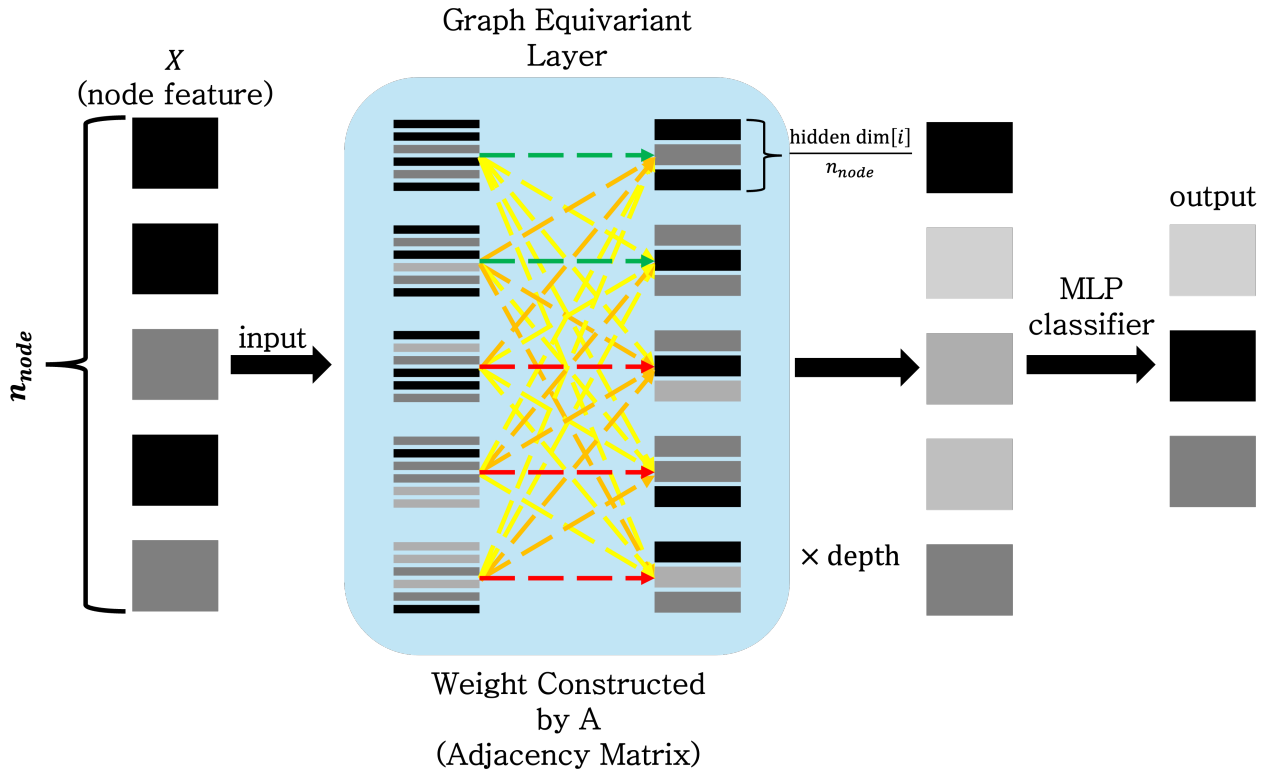

Figure 3: Task 2의 shared graph MLP 구조도. 같은 색의 연결은 동일한 가중치를 공유하며, graph-level pooling을 통해 permutation-invariant한 graph representation을 얻는다.

Parameter Sharing Construction

Hidden representation을 $h^\tau = (h_1^\tau, h_2^\tau, h_3^\tau)$으로 분할하면, Appendix 4의 유도에 따라 S_2-equivariance 조건을 만족하는 선형 레이어는 다음 block matrix 형태로 유일하게 결정된다.

$$h^{\tau+1} = \sigma\left(\begin{bmatrix} A & B & C \\ B & A & C \\ D & D & E \end{bmatrix} h^\tau + \begin{bmatrix} b_{\text{sym}} \\ b_{\text{sym}} \\ b_3 \end{bmatrix}\right) \tag{20}$$

첫 번째와 두 번째 block은 대칭적 구조(A, B, C, b_{sym} 공유), 세 번째 block은 구별되는 입자로 취급된다 (D, E, b_3 독립).

제 4장 Methods

본 장에서는 각 task의 구현 방식과 학습 조건을 정리한다. 앞 장이 실험의 목적과 비교 구도를 서술하였다면, 본 장은 실제 코드에 반영된 데이터 생성 규칙, 모델 구조, 하이퍼파라미터, 조작 변수를 명시하는 데 초점을 둔다.

제 1 절 Common Setting

모든 실험은 PyTorch로 구현하였다. 회귀 문제인 Task 1과 Task 3에서는 MSE loss를, 분류 문제인 Task 2에서는 CrossEntropy loss를 사용하였다. 최적화는 모두 Adam optimizer로 수행하였으며, 학습률은 전 실험에서 10^{-3}로 고정하였다. 대칭성 보존 정도는 입력에 군 작용을 가한 뒤 출력의 이론적 관계가 얼마나 유지되는지를 평균 제곱 오차로 측정하였다. Symmetry Error를 계산할 시, 무작위로 5 개의 $g \in G$에 대한 평균값을 이용하였다.

항목	Task 1, 3	Task 2
Framework	PyTorch	PyTorch, PyTorch Geometric
Optimizer	Adam	Adam
Learning rate	10^{-3}	10^{-3}
Loss	MSE	CrossEntropy
Symmetry metric	permutation MSE	logit MSE under permutation

Table 1: 전체 실험의 공통 설정

Data augmentation을 진행하는 경우, 학습 시 매 batch마다 임의의 permutation을 가하는 방식으로 구현하였다.

제 2 절 Task 1: Permutation-Invariant Set Function

Data

입력은 $x = (x_1, \ldots, x_n) \in [-2, 2]^n$이며, 각 원소는 균등분포에서 독립적으로 추출하였다. 목표 함수는

$$f(x) = \sum_{i=1}^{n} \phi(x_i), \qquad \phi(x) = \sin x + \exp x + \log |x| \tag{21}$$

로 정의하였다.

Model and Hyperparameters

Vanilla MLP의 경우, fully connected 구조를 이용하였으며 활성화 함수로는 LeakyReLU(0.01)을 이용하였다. Shared MLP(MLP + Share)도 동일한 구조를 이용하였으며, aggregation function으로는 평균값을 이용하였다. 모델의 shape의 경우, 각 은닉층에서 순서대로 block당 노드 수가 $8, 16, 8, 4, 2$가 되도록 고정하였다.
그 외의 Epoch는 최대 250으로 학습시켰으나, patience가 15인 Early stopping을 함께 사용하였다.

Controlled Variation

Task 1에서는 block의 수(n_{block})와 training data size를 변화시켰다. Table 2와 같이 두 값을 변화시키며 관찰하였다. training data size를 변화시킬 때, validation data size, test data size, batch size도 같이 변화시켰다.

하이퍼파라미터 종류						
n_{blocks}	1	2	5	10	20	
Training data size	40	80	400	800	4000	8000
Validation data size	10	20	100	200	1000	2000
Test data size	10	20	100	200	1000	2000
Batch Size	4	8	16	32	128	256

Table 2: Task 1에서 이용한 조작 변인

제 3 절 Task 2: Graph Classification

Data

그래프 분류 실험에는 TUDataset의 화학 분자 데이터셋인 NCI1을 이용하여, 분자의 독성 여부에 대한 이진 Classification을 실시하였다. 각 노드별로 원소가 one-hot-encoding된 크기 37의 feature를 가졌다. 최대 노드 개수가 50개 이하인 그래프만을 이용하였다.
각 graph는 최대 50개의 node로 제한하였으며, Vanilla MLP에서는 adjacency matrix 50×50과 node feature 50×37을 평탄화하여 총 4350차원 입력으로 사용하였다. 데이터 크기 조건은 500, 1500, 4110으로 두었고, 각 조건마다 $8:1:1$ 비율로 train, validation, test set을 나누었다.

Model and Hyperparameters

Shared MLP는 Figure 3와 같이 symmetric layer와 graph pooling를 거친 후 MLP 2층 classifier를 거친 모델을 이용하였다. 활성화 함수로는 LeakyReLU(0.01)을 이용하였다. Vanilla MLP도 Shared MLP와 동일한 은닉층 구성을 가졌으며, 인접행렬과 node featrues를 모두 입력으로 사용하여 일반 linear layer에 통과시켰다. 이후 attention pooling을 거친 후 Shared MLP와 같은 크기의 classifier를 이용하였다. training, test, validation data는 모두 $8:1:1$의 비율로 분할하였다.
그 외의 Epoch는 최대 250으로 학습시켰으나, patience가 15인 Early stopping을 함께 사용하였다

Controlled Variation

Task 2에서는 데이터 크기와 은닉층 개수를 변화시켰다. Table 3와 같이 두 값을 각각 변화시키며 관찰하였다. training data size를 변화시킬 때, validation data size, test data size, batch size도 같이 변화시켰다.

하이퍼파라미터 종류			
Training data size	500	1500	4110
은닉층 크기	$[64, 64, 16]$	$[256, 256, 128, 64, 16]$	$[256, 256, 256, 128, 128, 64, 16]$

Table 3: Task 2에서 사용한 조작 변인

제 4 절 Task 3: Symmetric Physical System Prediction

Data

식 (22)와 같은 힘이 입자 사이에 가정할 때의 입자 운동을 시뮬레이션한 데이터를 이용하였다.

$$\vec{F} = -\frac{m_1 m_2}{r^2 + \epsilon^2}\hat{r} \tag{22}$$

$\epsilon = 0.5$을 이용하였으며, 질량이 $(10, 10, 20)$인 세 입자가 운동하는 상황을 가정하였으며, 시간 간격 $dt = 0.01$을 두어 시뮬레이션을 진행하였다. 초기 위치와 속도는 표준정규분포에서 추출하였으며, 속도는 평균을 제거하여 전체 계의 병진 운동이 생기지 않는 데이터를 이용하였다. 각각 5000 step동안 전개하였고, 매 10 step마다 상태를 저장하여 현재 상태 x_t와 10 step 이후의 상태 x_{t+10}를 하나의 학습 쌍으로 사용하였다.

Model and Hyperparameters

Vanilla MLP는 fully connected 구조를 이용하였고, 활성화 함수로는 ReLU를 사용하였다. Shared MLP는 앞서 제시한 S_2에 대한 equivariant linear layer를 이용하여 구성하였으며, 역시 활성화 함수로 ReLU를 사용하였다.
그 외의 Epoch는 최대 400으로 학습시켰으나, patience가 20인 Early stopping을 함께 사용하였다

Controlled Variation

Task 3에서는 데이터 크기와 모델 크기를 변화시켰다. 데이터 크기는 1000, 5000, 20000, 40000의 네 조건으로 두었고, 각 조건에서 8 : 2 비율로 training data와 validation data를 분할하였다. 또한 모델 크기도 Table 4와 같이 세 가지에 대해 관찰하였다.

하이퍼파라미터 종류				
Train data size	800	4000	16000	32000
Validation data size	200	1000	4000	8000
Test data size	24950	24950	24950	24950
Batch size	128	256	512	1024
은닉층 크기	$[12, 48, 48, 12]$	$[12, 48, 96, 96, 48, 12]$	$[12, 96, 192, 48, 12]$	

Table 4: Task 3에서 사용한 조작 변인

제 5장 Results and Discussion

본 장에서는 generalization performance, symmetry preservation, sample efficiency, parameter efficiency의 네 관점에서 결과를 정리한다. 각 task에 대해 핵심 figure를 제시하고, 이어서 전체 정량 결과를 표로 정리하였다. 같은 조건에서 더 좋은 값은 굵게 표시하였다. Loss와 symmetry error는 작을수록, accuracy는 클수록, parameter 수는 적을수록 더 바람직한 값으로 해석하였다.

제 1 절 Results

Task 1: Permutation-Invariant Set Function

Task 1에서는 실제 training data size를 기준으로 결과를 다시 집계하였다. Figure 4는 train size와 block 수에 따라 vanilla MLP와 shared MLP의 test loss 및 symmetry error가 어떻게 달라지는지를 보여준다.

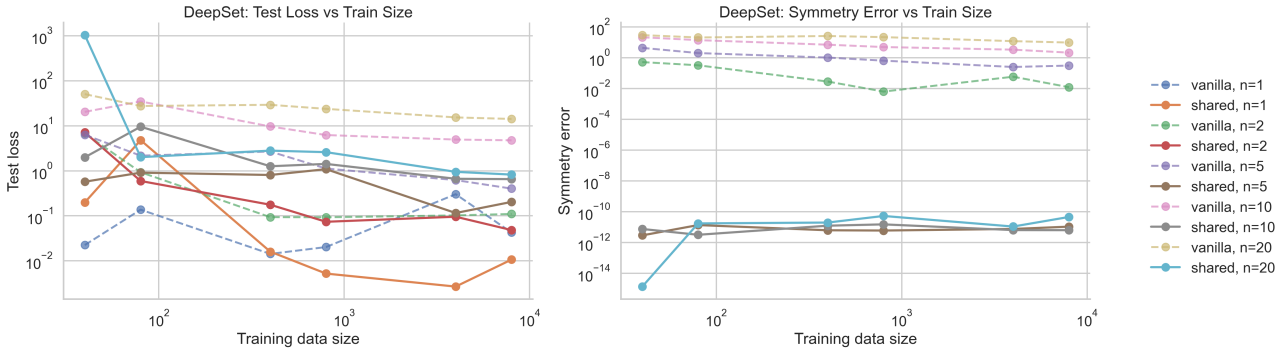

Figure 4: Task 1의 test loss 및 symmetry error 비교. 가로축은 실제 training data size이며, 각 곡선은 block 수 조건을 나타낸다. Shared MLP는 대부분의 조건에서 symmetry error를 거의 0으로 유지하며, block 수가 증가할수록 vanilla MLP 대비 더 안정적인 일반화 성능을 보인다.

Shared MLP는 $n_{\text{block}} = 2, 5, 10, 20$의 전 조건에서 symmetry error를 거의 0으로 유지하였다. 반면 vanilla MLP의 symmetry error는 block 수가 증가할수록 빠르게 커졌다. 일반화 성능 역시 block 수가 충분히 클 때 shared 구조의 이점을 분명히 드러냈다. 특히 $n_{\text{block}} = 5, 10, 20$에서는 대부분의 조건에서 shared 모델이 더 낮은 test loss를 보였다.

Figure 5가 보여주듯이, shared MLP의 파라미터 수는 모든 조건에서 651개로 일정하였고, vanilla MLP는 block 수에 따라 345개에서 122401개까지 증가하였다.

n_{block}	Train size	Vanilla loss	Shared loss	Vanilla sym.	Shared sym.	Vanilla param.	Shared param.
1	40	**0.022**	0.198	**0**	**0**	**345**	651
1	80	**0.136**	4.768	**0**	**0**	**345**	651
1	400	**0.014**	0.016	**0**	**0**	**345**	651
1	800	0.020	**0.005**	**0**	**0**	**345**	651
1	4000	0.300	**0.003**	**0**	**0**	**345**	651
1	8000	0.042	**0.011**	**0**	**0**	**345**	651
2	40	**6.708**	7.084	5.16e-01	**0**	1297	**651**
2	80	0.925	**0.591**	3.20e-01	**0**	1297	**651**
2	400	**0.093**	0.175	2.81e-02	**0**	1297	**651**
2	800	0.092	**0.073**	6.27e-03	**0**	1297	**651**

n_{block}	Train size	Vanilla loss	Shared loss	Vanilla sym.	Shared sym.	Vanilla param.	Shared param.
2	4000	0.102	**0.095**	5.79e-02	**0**	1297	**651**
2	8000	0.109	**0.048**	1.21e-02	**0**	1297	**651**
5	40	6.277	**0.569**	4.33e+00	**2.84e-12**	7801	**651**
5	80	2.157	**0.906**	1.99e+00	**1.32e-11**	7801	**651**
5	400	2.668	**0.803**	1.02e+00	**6.07e-12**	7801	**651**
5	800	1.116	**1.082**	6.40e-01	**5.84e-12**	7801	**651**
5	4000	0.619	**0.115**	2.49e-01	**7.41e-12**	7801	**651**
5	8000	0.400	**0.202**	3.06e-01	**1.04e-11**	7801	**651**
10	40	20.317	**1.987**	2.12e+01	**7.37e-12**	30801	**651**
10	80	34.558	**9.483**	1.38e+01	**3.09e-12**	30801	**651**
10	400	9.615	**1.255**	6.98e+00	**1.21e-11**	30801	**651**
10	800	6.196	**1.415**	4.91e+00	**1.47e-11**	30801	**651**
10	4000	4.919	**0.661**	3.31e+00	**6.25e-12**	30801	**651**
10	8000	4.732	**0.650**	2.12e+00	**6.17e-12**	30801	**651**
20	40	**50.251**	1029.625	2.93e+01	**1.33e-15**	122401	**651**
20	80	27.391	**2.002**	2.05e+01	**1.70e-11**	122401	**651**
20	400	29.094	**2.791**	2.56e+01	**1.93e-11**	122401	**651**
20	800	23.660	**2.581**	2.17e+01	**5.17e-11**	122401	**651**
20	4000	15.272	**0.944**	1.19e+01	**1.06e-11**	122401	**651**
20	8000	14.120	**0.819**	9.61e+00	**4.46e-11**	122401	**651**

Table 5: Task 1의 전체 정량 결과. 같은 조건에서 더 낮은 loss, 더 낮은 symmetry error, 더 적은 parameter를 굵게 표시하였다.

Task 2: Graph Classification

Task 2에서는 vanilla, vanilla + augmentation, shared, shared + augmentation의 네 모델을 비교하였다. Figure 6는 hidden setting과 data size에 따른 accuracy 변화를 보여준다.

정확도 기준으로 보면 shared 구조는 대부분의 조건에서 vanilla 구조보다 우세하였다. 특히 augmentation이 없는 직접 비교에서 shared 모델은 모든 hidden setting과 data size에서 vanilla보다 높은 accuracy를 기록하였다. Augmentation은 vanilla 모델을 개선하였으나, exact equivariance를 제공하는 shared 구조를 완전히 대체하지는 못하였다.

Hidden setting	Data size	Model	Loss	Accuracy	Symmetry error
3	500	Share	0.632	0.660	**3.34e-15**
3	500	Share + Aug	**0.557**	**0.720**	4.35e-15
3	500	Vanilla	1.081	0.360	3.41e-01
3	500	Vanilla + Aug	0.693	0.440	6.67e-03
3	1500	Share	0.671	**0.647**	**5.66e-15**
3	1500	Share + Aug	**0.648**	0.640	5.75e-15
3	1500	Vanilla	2.123	0.560	9.56e+00
3	1500	Vanilla + Aug	0.660	0.633	2.25e-02
3	4110	Share	**0.604**	0.659	9.27e-15
3	4110	Share + Aug	0.606	**0.669**	**6.81e-15**
3	4110	Vanilla	3.549	0.523	1.96e+01
3	4110	Vanilla + Aug	0.612	0.664	2.81e-02
5	500	Share	**0.671**	**0.580**	**9.48e-16**
5	500	Share + Aug	0.690	0.540	1.94e-15
5	500	Vanilla	2.550	0.440	1.89e+01

Hidden setting	Data size	Model	Loss	Accuracy	Symmetry error
5	500	Vanilla + Aug	0.686	0.460	2.14e-03
5	1500	Share	**0.636**	0.613	2.31e-15
5	1500	Share + Aug	0.645	**0.620**	**1.31e-15**
5	1500	Vanilla	6.077	0.527	3.67e+01
5	1500	Vanilla + Aug	0.662	0.560	3.08e-02
5	4110	Share	0.604	0.633	**3.86e-15**
5	4110	Share + Aug	**0.593**	**0.664**	4.31e-15
5	4110	Vanilla	9.236	0.521	1.61e+02
5	4110	Vanilla + Aug	0.624	0.613	5.29e-02
7	500	Share	**0.628**	**0.680**	1.12e-15
7	500	Share + Aug	0.714	0.400	**4.73e-16**
7	500	Vanilla	5.401	0.340	2.54e+01
7	500	Vanilla + Aug	0.725	0.460	2.19e-02
7	1500	Share	0.666	0.567	**1.13e-15**
7	1500	Share + Aug	**0.657**	**0.660**	1.93e-15
7	1500	Vanilla	10.659	0.520	1.35e+02
7	1500	Vanilla + Aug	0.748	0.547	7.86e-02
7	4110	Share	0.614	**0.662**	**3.01e-15**
7	4110	Share + Aug	**0.607**	0.655	3.75e-15
7	4110	Vanilla	8.615	0.538	1.83e+02
7	4110	Vanilla + Aug	0.625	0.637	4.18e-02

Table 6: Task 2의 전체 정량 결과. 같은 hidden setting과 data size에서 더 낮은 loss, 더 높은 accuracy, 더 낮은 symmetry error를 굵게 표시하였다.

Task 3: Symmetric Physical System Prediction

Task 3에서는 데이터 크기, 모델 크기, augmentation 적용 여부를 함께 변화시키며 네 모델을 비교하였다. Figure 8는 small, medium, large의 각 모델 크기에서 data size에 따른 test loss와 symmetry error 변화를 보여주고, Figure 9은 각 family에서 가장 낮은 test loss를 기록한 대표 모델의 rollout 예시를 보여준다. 정량적으로는 모든 모델 크기와 data size 조건에서 vanilla 계열이 더 낮은 test loss를 기록하였다. 반면 shared 계열은 모든 조건에서 symmetry error를 10^{-12} 수준으로 유지하였으며, small, medium, large 조건에서 각각 1992, 7528, 16552개의 파라미터만으로 동일한 대칭성을 안정적으로 보존하였다. 가장 낮은 전체 test loss는 large vanilla 모델의 data size 40000 조건에서 0.032였고, shared 계열의 최저 test loss는 large shared + augmentation 모델의 동일 조건에서 0.053이었다.

Model size	Data size	Model	Test loss	Symmetry error	Parameters
Small	1000	Share	0.823	7.02e-13	**1992**
Small	1000	Share + Aug	0.880	**6.64e-13**	**1992**
Small	1000	Vanilla	0.783	2.705	3564
Small	1000	Vanilla + Aug	**0.734**	0.182	3564
Small	5000	Share	0.615	**9.23e-13**	**1992**
Small	5000	Share + Aug	0.618	9.24e-13	**1992**
Small	5000	Vanilla	**0.408**	2.673	3564
Small	5000	Vanilla + Aug	0.648	0.073	3564
Small	20000	Share	0.387	**1.60e-12**	**1992**
Small	20000	Share + Aug	0.447	1.61e-12	**1992**
Small	20000	Vanilla	**0.158**	2.262	3564
Small	20000	Vanilla + Aug	0.369	0.108	3564

Model size	Data size	Model	Test loss	Symmetry error	Parameters
Small	40000	Share	0.307	2.93e-12	**1992**
Small	40000	Share + Aug	0.313	**2.72e-12**	**1992**
Small	40000	Vanilla	**0.092**	1.366	3564
Small	40000	Vanilla + Aug	0.288	0.130	3564
Medium	1000	Share	0.901	2.78e-12	**7528**
Medium	1000	Share + Aug	0.907	**2.24e-12**	**7528**
Medium	1000	Vanilla	0.811	2.586	19884
Medium	1000	Vanilla + Aug	**0.752**	0.219	19884
Medium	5000	Share	0.646	**2.27e-12**	**7528**
Medium	5000	Share + Aug	0.676	2.47e-12	**7528**
Medium	5000	Vanilla	**0.244**	3.236	19884
Medium	5000	Vanilla + Aug	0.513	0.166	19884
Medium	20000	Share	0.236	**3.45e-12**	**7528**
Medium	20000	Share + Aug	0.272	4.46e-12	**7528**
Medium	20000	Vanilla	**0.083**	2.691	19884
Medium	20000	Vanilla + Aug	0.281	0.180	19884
Medium	40000	Share	0.132	5.28e-12	**7528**
Medium	40000	Share + Aug	0.154	**4.78e-12**	**7528**
Medium	40000	Vanilla	**0.084**	2.559	19884
Medium	40000	Vanilla + Aug	0.193	0.177	19884
Large	1000	Share	0.801	**2.16e-12**	**16552**
Large	1000	Share + Aug	0.808	2.18e-12	**16552**
Large	1000	Vanilla	0.753	2.147	29724
Large	1000	Vanilla + Aug	**0.747**	0.137	29724
Large	5000	Share	0.566	**2.01e-12**	**16552**
Large	5000	Share + Aug	0.551	2.10e-12	**16552**
Large	5000	Vanilla	**0.141**	2.602	29724
Large	5000	Vanilla + Aug	0.405	0.209	29724
Large	20000	Share	0.228	3.23e-12	**16552**
Large	20000	Share + Aug	0.194	**2.79e-12**	**16552**
Large	20000	Vanilla	**0.049**	1.688	29724
Large	20000	Vanilla + Aug	0.157	0.127	29724
Large	40000	Share	0.058	**3.29e-12**	**16552**
Large	40000	Share + Aug	0.053	3.49e-12	**16552**
Large	40000	Vanilla	**0.032**	1.435	29724
Large	40000	Vanilla + Aug	0.092	0.110	29724

Table 7: Task 3의 전체 정량 결과. 같은 모델 크기와 data size 조건에서 더 낮은 loss, 더 낮은 symmetry error, 더 적은 parameter를 굵게 표시하였다.

제 2 절 Discussion

본 절에서는 실험 설계에서 제시한 세 가지 연구 질문에 순서대로 답한다.

Q1. Parameter sharing은 일반화 성능 향상에 기여하는가?

Task 1과 Task 2의 결과는 이 질문에 긍정적으로 답한다. Task 1에서 symmetry가 문제 정의 자체를 이루는 조건, 즉 $n_{\text{block}} \geq 5$이상에서 shared 모델은 거의 모든 데이터 크기 조건에서 vanilla보다 낮은 test loss를 기록하였다. 특히 651개의 파라미터만으로 최대 122401개의 파라미터를 사용하는 vanilla 대비 더 나은 일반화 성능을 달성하였다는 점은, 이 효과가 단순한 모델 용량의 차이로 설명되지 않음을 보여준다. Task 2에서도 모든 hidden setting과 data size 조건에서 shared 모델이 augmentation 없는 vanilla보다

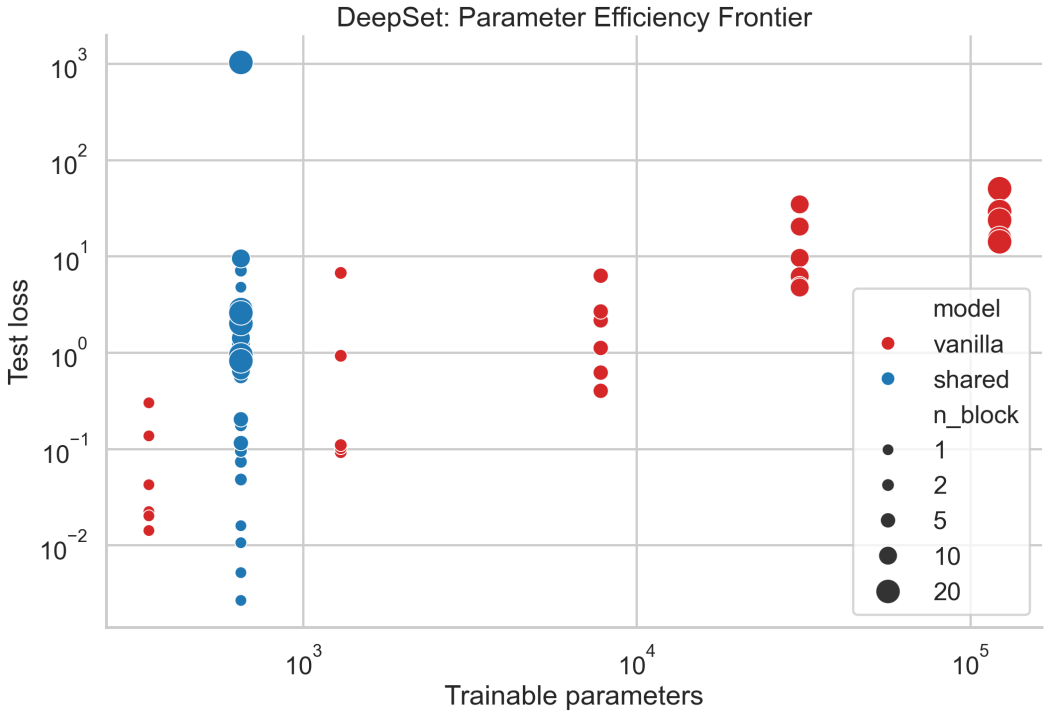

Figure 5: Task 1의 parameter efficiency 비교. Shared MLP는 block 수와 무관하게 파라미터 수가 거의 일정한 반면, vanilla MLP는 block 수 증가에 따라 파라미터 수가 빠르게 증가한다.

Figure 6: Task 2의 accuracy 비교. Shared 계열 모델은 대부분의 조건에서 vanilla 계열보다 높은 정확도를 보이며, augmentation은 특히 vanilla 모델의 성능을 개선한다.

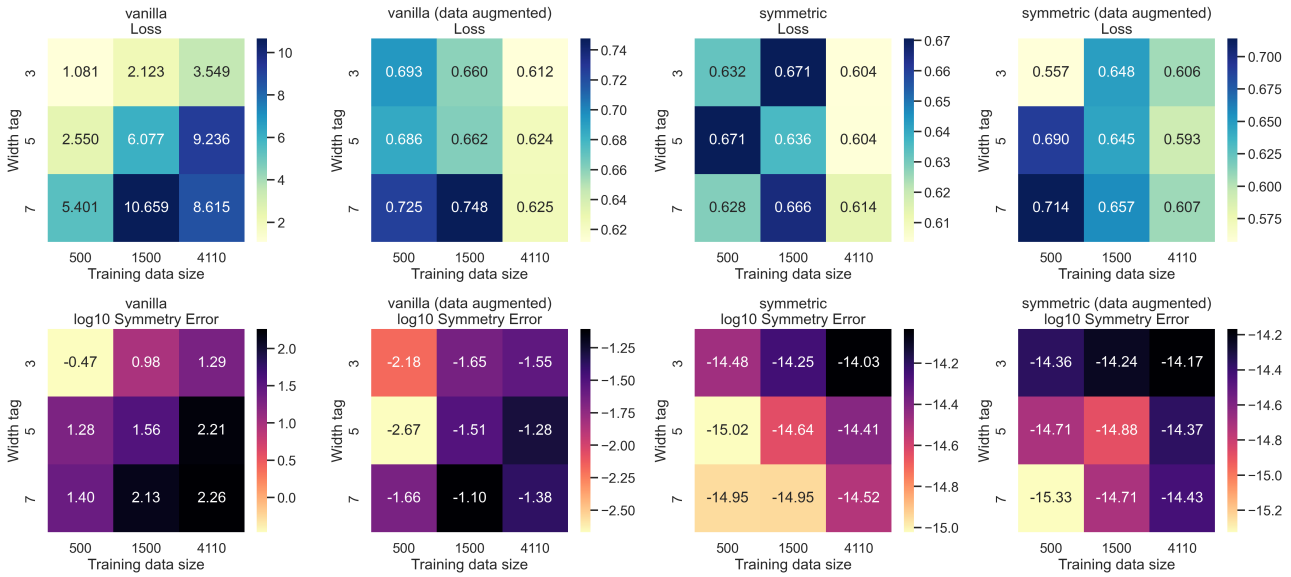

Figure 7: Task 2의 test loss 및 symmetry error heatmap. Shared 계열 모델은 거의 0에 가까운 symmetry error를 유지하며, augmentation은 vanilla 모델의 symmetry violation을 줄이지만 exact equivariance에는 도달하지 못한다.

Figure 8: Task 3의 정량적 비교. 모든 모델 크기에서 data size가 증가할수록 test loss는 감소하였다. Shared 계열 모델은 전 구간에서 symmetry error를 거의 0으로 유지하였고, augmentation은 vanilla 계열의 symmetry violation을 줄이는 데 기여하였다.

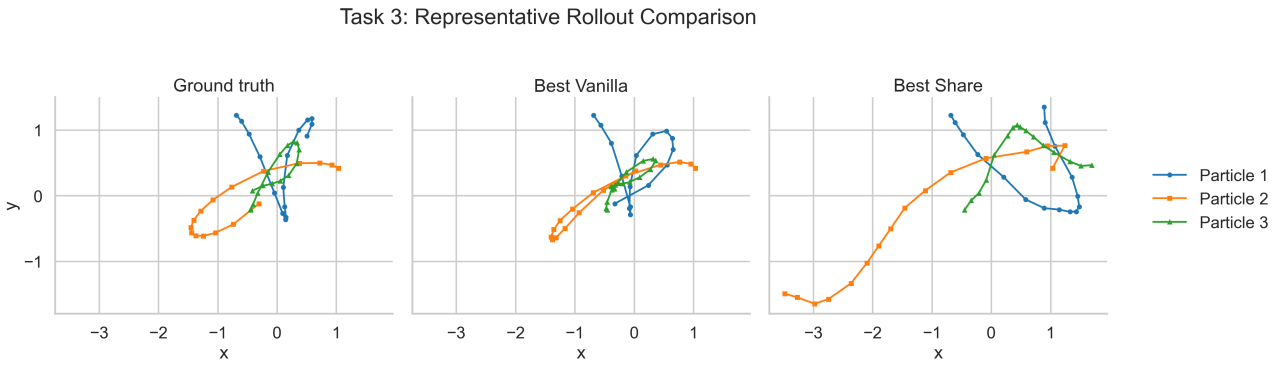

Figure 9: Task 3의 representative rollout 비교. 각 family에서 test loss가 가장 낮은 모델을 선택하여 비교하였다. Shared 계열 모델은 궤적 전체에서 두 대칭 입자에 대한 교환 구조를 일관되게 유지한다.

높은 accuracy를 기록하였다.

다만 Task 1의 $n_{\text{block}} = 1$ 조건과 같이 symmetry가 task를 규정하는 정도가 약하거나 문제 규모가 작아 vanilla 모델도 충분한 정보를 얻을 수 있는 경우에는 shared 구조의 이점이 나타나지 않았다. 이는 parameter sharing의 효과가 task에서 symmetry가 차지하는 역할의 크기에 비례한다는 점을 시사한다.

Q2. Parameter sharing의 효과는 data augmentation과 어떠한 차이를 보이는가?

두 접근의 차이는 Task 2에서 가장 분명하게 드러난다. Augmentation은 vanilla 모델의 symmetry error를 수 십에서 10^{-2} 수준으로 크게 줄이고 accuracy도 유의미하게 개선하였다. 그러나 shared 구조는 symmetry error를 10^{-15} 수준으로 유지하였으며, augmentation 없는 shared 모델이 augmentation을 적용한 vanilla보다 일관되게 높은 accuracy를 기록하였다.

이 차이는 두 방법의 근본적인 성격 차이에서 비롯된다. Data augmentation은 모델이 데이터를 통해 symmetry를 *경험적으로 학습*하도록 유도하는 접근이다. 충분한 데이터가 주어지면 효과적이지만, 학습 후에도 symmetry가 구조적으로 보장되지는 않는다. 반면 parameter sharing은 symmetry를 위반하는 함수를 가설 공간에서 *구조적으로 배제*한다. 따라서 데이터 양과 무관하게 exact equivariance가 보장되며, symmetry가 강한 제약으로 작용하는 task에서는 더 효율적인 학습이 가능하다.

MLP + Aug + Share 조합은 두 방법을 결합하여 Task 2에서 가장 높은 accuracy를 보이는 경우도 있었으나, shared 단독 대비 일관된 개선은 나타나지 않았다. 이는 모델이 이미 구조적으로 symmetry를 만족하는 상태에서 augmentation이 추가적으로 제공하는 정보가 제한적임을 의미한다.

Q3. Parameter sharing의 효과는 이상화된 문제에 국한되는가?

Task 3의 결과는 이 질문에 대해 보다 구체적인 답을 제공한다. 데이터 크기와 모델 크기를 함께 변화시킨 모든 조건에서, shared 계열 모델은 symmetry error를 10^{-12} 수준으로 유지하였다. 이는 물리계 상태 전이 예측과 같이 입력 구조와 동역학이 함께 작용하는 문제에서도 parameter sharing이 여전히 강한 구조적 제약으로 작동함을 보여준다.

반면 일반화 성능의 관점에서는 symmetry prior만으로 충분하지 않았다. 모든 조건에서 test loss의 최저값은 vanilla 계열에서 나타났고, 가장 좋은 결과는 large vanilla 모델의 data size 40000 조건에서 얻어졌다. Augmentation은 vanilla 계열의 symmetry violation을 크게 줄였지만, shared 계열이 제공하는 exact equivariance에는 도달하지 못하였다.

따라서 Q3에 대해서는 "parameter sharing의 효과는 이상화된 toy problem에 국한되지 않으며, 복잡한 물리계에서도 symmetry preservation과 parameter efficiency의 장점은 유지된다"고 답할 수 있다. 다만 예측 오차 자체를 항상 최소화하는 것은 아니므로, 복잡한 task에서는 symmetry prior와 충분한 모델 용량을 함께 고려할 필요가 있다.

종합

세 질문에 걸친 결과를 종합하면, parameter sharing은 단순한 모델 경량화 기법이 아니라 **task에 부합하는 hypothesis space만을 남기도록 하는 구조적 inductive bias**로 해석하는 것이 더 적절하다. 그 효과는 하나의 단일 지표로 환원되지 않으며, task에서 symmetry가 차지하는 역할에 따라 generalization performance, symmetry preservation, parameter efficiency, physical consistency 중 서로 다른 측면에서 드러난다. 특히 symmetry가 문제를 강하게 규정하는 조건일수록, data augmentation으로는 달성할 수 없는 구조적 이점이 뚜렷하게 나타난다.

참고문헌

[1] Peter W Battaglia et al. "Relational inductive biases, deep learning, and graph networks". In: *arXiv preprint arXiv:1806.01261* (2018).

[2] Michael M Bronstein et al. "Geometric deep learning: Grids, groups, graphs, geodesics, and gauges". In: *arXiv preprint arXiv:2104.13478* (2021).

[3] Taco Cohen and Max Welling. "Group equivariant convolutional networks". In: *International conference on machine learning*. PMLR. 2016, pp. 2990–2999.

[4] Marc Finzi, Max Welling, and Andrew Gordon Wilson. "A practical method for constructing equivariant multilayer perceptrons for arbitrary matrix groups". In: *International conference on machine learning*. PMLR. 2021, pp. 3318–3328.

[5] Kurt Hornik. "Approximation capabilities of multilayer feedforward networks". In: *Neural networks* 4.2 (1991), pp. 251–257.

[6] Risi Kondor and Shubhendu Trivedi. "On the generalization of equivariance and convolution in neural networks to the action of compact groups". In: *International conference on machine learning*. PMLR. 2018, pp. 2747–2755.

[7] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *nature* 521.7553 (2015), pp. 436–444.

[8] Tom M Mitchell. "The need for biases in learning generalizations". In: (1980).

[9] Siamak Ravanbakhsh, Jeff Schneider, and Barnabas Poczos. "Equivariance through parameter-sharing". In: *International conference on machine learning*. PMLR. 2017, pp. 2892–2901.

[10] Jürgen Schmidhuber. "Deep learning in neural networks: An overview". In: *Neural networks* 61 (2015), pp. 85–117.

[11] Jean-Pierre Serre et al. *Linear representations of finite groups*. Vol. 42. Springer, 1977.

[12] David H Wolpert. "The lack of a priori distinctions between learning algorithms". In: *Neural computation* 8.7 (1996), pp. 1341–1390.

[13] Manzil Zaheer et al. "Deep sets". In: *Advances in neural information processing systems* 30 (2017).

Abstract

An Empirical Study of Symmetry-Induced Parameter Sharing as an Inductive Bias in Neural Networks

Ju Hwan Kim

Seoul Science High School

This study empirically investigates whether parameter sharing based on task-inherent symmetry can serve as an inductive bias in neural networks. Although standard Multi-Layer Perceptrons (MLPs) are highly expressive, they are also prone to learning unnecessary dependencies on input order or structure. To address this issue, prior work has typically relied on data augmentation or structurally constrained architectures to improve generalization. In this work, we propose a method that restricts the function space by enforcing weight sharing according to task symmetry, thereby inducing a structural inductive bias. To evaluate the effectiveness of this approach, we compare standard MLPs, data augmentation-based methods, and symmetry-aware parameter-sharing models on three toy tasks, analyzing their performance in terms of symmetric robustness, sample efficiency, parameter efficiency, and generalization. Experimental results show that models with symmetry-based parameter sharing achieve more stable performance and improved generalization, particularly in settings with limited data and restricted model capacity. These findings suggest that parameter sharing is not merely a model compression technique, but can also function as a structural inductive bias that promotes generalization in neural networks.

Keywords : symmetry, parameter sharing, inductive bias, robustness, efficiency, neural networks
Student Number : **24031**

Appendix 1 Proof: Basis of $\mathcal{W}_G$

군 G가 입력 공간 $\mathbb{R}^n$과 출력 공간 $\mathbb{R}^m$ 위에서 각각 치환 표현 $\rho_{\text{in}} : G \to S_n$, $\rho_{\text{out}} : G \to S_m$으로 작용한다고 하자. Equivariance 조건을 만족하는 가중치 행렬의 집합을

$$\mathcal{W}_G = \left\{ W \in \mathbb{R}^{m \times n} \mid W\rho_{\text{in}}(g) = \rho_{\text{out}}(g)W,\ \forall g \in G \right\} \tag{23}$$

로 정의한다. 또한 인덱스 쌍 $(i, j) \in \{1, \dots, m\} \times \{1, \dots, n\}$ 위의 G-작용을

$$g \cdot (i, j) \ := \ \big(\rho_{\text{out}}(g)(i),\ \rho_{\text{in}}(g)(j)\big) \tag{24}$$

로 정의하고, 이 작용에 의한 orbit 분해를 $\{1, \dots, m\} \times \{1, \dots, n\} = O_1 \sqcup \cdots \sqcup O_k$로 표기한다. 각 orbit O_l에 대해 basis 행렬을

$$B_l = \sum_{(i,j) \in O_l} E_{ij}, \qquad l = 1, \dots, k \tag{25}$$

로 정의한다. 여기서 E_{ij}는 (i, j) 위치만 1인 표준 기저 행렬이다.

Theorem 1 ((Ravanbakhsh, Schneider, and Poczos 2017))**.** $\mathcal{W}_G$는 $\mathbb{R}^{m \times n}$의 선형 부분공간이며, $\{B_1, \dots, B_k\}$는 $\mathcal{W}_G$의 기저를 이룬다. 따라서 임의의 $W \in \mathcal{W}_G$는

$$W = \sum_{l=1}^{k} w_l B_l, \qquad w_l \in \mathbb{R} \tag{26}$$

로 유일하게 표현된다.

Proof. **(선형 부분공간)** Equivariance 조건은 W에 대해 선형이므로, $\mathcal{W}_G$는 덧셈과 스칼라 곱에 닫혀 있다.
(Orbit 내 값 공유) $W \in \mathcal{W}_G$의 (i, j) 원소를 비교하면, 치환 행렬의 성질에 의해

$$\big[W\rho_{\text{in}}(g)\big]_{ij} = W_{i,\,\rho_{\text{in}}(g)^{-1}(j)}, \qquad \big[\rho_{\text{out}}(g)W\big]_{ij} = W_{\rho_{\text{out}}(g)^{-1}(i),\,j} \tag{27}$$

이다. 이 둘이 같아야 하므로 g를 g^{-1}으로 치환하면

$$W_{i,\,\rho_{\text{in}}(g)(j)} = W_{\rho_{\text{out}}(g)(i),\,j}, \qquad \forall g \in G,\ \forall i, j. \tag{28}$$

즉, (i, j)와 $g \cdot (i, j)$에서의 W 값이 같으므로, 같은 orbit O_l 안의 모든 인덱스 쌍은 동일한 값 w_l을 가진다.
(Spanning) $W_{ij} = w_l$ $(\forall (i, j) \in O_l)$이므로 $W = \sum_{l=1}^{k} w_l B_l$이다.
(선형 독립) $\sum_{l=1}^{k} c_l B_l = 0$이라 하자. Orbit들이 서로소이므로, 임의의 $(i, j) \in O_l$에 대해 $\big[\sum_{l'} c_{l'} B_{l'}\big]_{ij} = c_l = 0$이다. l이 임의적이므로 $c_l = 0$ $(\forall l)$.
따라서 $\{B_1, \dots, B_k\}$는 $\mathcal{W}_G$의 기저이고, $\dim(\mathcal{W}_G) = k$이다. □ □

Appendix 2 Task 1: S_n-Equivariant Layer 유도

입력을 $h = (h_1, \ldots, h_n) \in (\mathbb{R}^d)^n$으로 두고, $\pi \in S_n$의 작용을 $\rho(\pi)h = (h_{\pi^{-1}(1)}, \ldots, h_{\pi^{-1}(n)})$으로 정의한다. 선형 레이어 $L : (\mathbb{R}^d)^n \to (\mathbb{R}^d)^n$을 $(L(h))_i = \sum_j W_{ij} h_j$, $W_{ij} \in \mathbb{R}^{d\times d}$로 쓰면, equivariance 조건 $L \circ \rho(\pi) = \rho(\pi) \circ L$은

$$W_{\pi(i),\pi(j)} = W_{ij}, \quad \forall \pi \in S_n, \ \forall i, j \tag{29}$$

를 요구한다. 즉, W_{ij}는 인덱스 쌍 (i, j)의 S_n-orbit 위에서 상수여야 한다 (Ravanbakhsh, Schneider, and Poczos 2017).

S_n은 $\{1, \ldots, n\}^2$ 위의 인덱스 쌍을 정확히 두 orbit으로 분류한다: $O_{\text{diag}} = \{(i, i)\}$와 $O_{\text{off}} = \{(i, j) \mid i \neq j\}$. 따라서

$$W_{ij} = \begin{cases} A, & i = j \\ B, & i \neq j \end{cases} \qquad A, B \in \mathbb{R}^{d\times d} \tag{30}$$

이 S_n-equivariant linear layer의 유일한 형태이며, 대입하면

$$(L(h))_i = Ah_i + B\sum_{j\neq i} h_j \tag{31}$$

를 얻는다. Bias를 포함하면, 모든 block에 공통인 $b \in \mathbb{R}^d$만이 equivariance를 보존하므로

$$h'_i = Ah_i + B\sum_{j\neq i} h_j + b, \quad i = 1, \ldots, n. \tag{32}$$

Equivariance 검증 $\tilde{h} = \rho(\pi)h$에 대해 $\tilde{h}_i = h_{\pi^{-1}(i)}$이고, $\sum_{j\neq i} h_{\pi^{-1}(j)} = \sum_{j\neq \pi^{-1}(i)} h_j$이므로

$$(L(\tilde{h}))_i = Ah_{\pi^{-1}(i)} + B\sum_{j\neq i} h_{\pi^{-1}(j)} + b = (L(h))_{\pi^{-1}(i)} = (\rho(\pi)L(h))_i. \quad \square \tag{33}$$

□

Invariance는 마지막 equivariant hidden representation에 permutation-invariant aggregation $y = \frac{1}{n}\sum_i h'_i$을 적용함으로써 얻는다 (Zaheer et al. 2017).

Appendix 3 Task 2: Permutation-Equivariant Graph Layer 유도

인접행렬 $A \in \mathbb{R}^{n\times n}$과 node feature $X \in \mathbb{R}^{n\times d_{\text{in}}}$를 입력으로 받아 $Y \in \mathbb{R}^{n\times d_{\text{out}}}$를 출력하는 graph layer를 고려한다. Node relabeling $P \in S_n$의 작용은 $(A, X) \mapsto (PAP^\top, PX)$이며, equivariance 조건은

$$g(PAP^\top, PX) = P\, g(A, X), \quad \forall P \in S_n. \tag{34}$$

가장 일반적인 선형 graph layer는

$$Y_i = \sum_{j,k} T_{ijk}\, A_{jk} X_k \tag{35}$$

로 쓸 수 있다. 이를 equivariance 조건에 대입하면, T_{ijk}가 만족해야 하는 조건은

$$T_{\pi(i)\,\pi(j)\,\pi(k)} = T_{ijk}, \quad \forall \pi \in S_n \tag{36}$$

이다. 즉, T_{ijk}는 index triple (i, j, k)의 S_n-orbit 위에서 상수여야 한다 (Ravanbakhsh, Schneider, and Poczos 2017).
세 인덱스의 equality pattern에 의한 orbit 분류는 Bell number $B_3 = 5$개이다:

$$O_1 : i = j = k, \quad O_2 : i = j \neq k, \quad O_3 : i = k \neq j, \quad O_4 : j = k \neq i, \quad O_5 : i, j, k \text{ pairwise distinct.} \tag{37}$$

각 orbit에 대응하는 basis operator를 구성하면

$$B_1 = \operatorname{diag}(A) \odot X \tag{38}$$

$$B_2 = AX - B_1 \tag{39}$$

$$B_3 = \big(\deg(A) - \operatorname{diag}(A)\big) \odot X \tag{40}$$

$$B_4 = \Big(\sum_j A_{jj} X_j\Big)\mathbf{1}_n^\top - B_1 \tag{41}$$

$$B_5 = \Big(\sum_i (AX)_i\Big)\mathbf{1}_n^\top - B_1 - B_2 - B_3 - B_4 \tag{42}$$

이고, 임의의 permutation-equivariant linear graph layer는 이들의 선형결합으로 유일하게 표현된다:

$$Y = \sum_{\ell=1}^{5} W_\ell(B_\ell) + b, \qquad W_\ell \in \mathbb{R}^{d_{\text{in}}\times d_{\text{out}}}. \tag{43}$$

Equivariance 검증 각 B_ℓ은 index triple의 equality pattern으로만 정의되므로 $B_\ell(PAP^\top, PX) = PB_\ell(A, X)$이 성립한다. W_ℓ은 feature dimension에만 작용하여 permutation과 교환하므로

$$Y(PAP^\top, PX) = \sum_\ell W_\ell(PB_\ell) + b = P\left(\sum_\ell W_\ell(B_\ell) + b\right) = PY(A, X). \quad \square \tag{44}$$

□

이후 permutation-invariant graph pooling을 적용하여 graph-level representation을 얻는다.

Appendix 4 Task 3: S_2-Equivariant Layer 유도

세 입자의 hidden state를 $h = (h_1, h_2, h_3) \in (\mathbb{R}^d)^3$으로 두자. 첫 번째와 두 번째 입자가 동일한 질량을 가지므로, 대칭군은 $S_2 = \{e, g\}$이며 생성원 g의 작용은

$$g \cdot (h_1, h_2, h_3) = (h_2, h_1, h_3) \tag{45}$$

으로 정의된다. 선형 레이어 $L(h) = Wh + b$를 3×3 block matrix로 쓰면, g의 작용은

$$\rho(g) = \begin{bmatrix} 0 & I & 0 \\ I & 0 & 0 \\ 0 & 0 & I \end{bmatrix} \tag{46}$$

이고, equivariance 조건 $L\rho(g) = \rho(g)L$은 양변을 전개하면

$$M_{11} = M_{22}, \quad M_{12} = M_{21}, \quad M_{13} = M_{23}, \quad M_{31} = M_{32} \tag{47}$$

를 요구한다 (M_{33}에는 제약 없음). $A := M_{11}$, $B := M_{12}$, $C := M_{13}$, $D := M_{31}$, $E := M_{33}$으로 두면, S_2-equivariant affine layer의 유일한 형태는

$$h' = \begin{bmatrix} A & B & C \\ B & A & C \\ D & D & E \end{bmatrix} h + \begin{bmatrix} b_{\text{sym}} \\ b_{\text{sym}} \\ b_3 \end{bmatrix} \tag{48}$$

이다. Bias 항 역시 $\rho(g)b = b$를 만족해야 하므로 첫 두 block은 동일한 b_{sym}을 공유한다.

Equivariance 검증 $L(g \cdot h)$의 각 block을 계산하면

$$\begin{aligned} &\text{Block 1: } Ah_2 + Bh_1 + Ch_3 + b_{\text{sym}}, \\ &\text{Block 2: } Bh_2 + Ah_1 + Ch_3 + b_{\text{sym}}, \\ &\text{Block 3: } Dh_2 + Dh_1 + Eh_3 + b_3. \end{aligned}$$

한편 $g \cdot L(h)$는 $L(h)$의 Block 1과 2를 교환하므로

$$\begin{aligned}
&\text{Block 1: } Bh_1 + Ah_2 + Ch_3 + b_{\text{sym}},\\
&\text{Block 2: } Ah_1 + Bh_2 + Ch_3 + b_{\text{sym}},\\
&\text{Block 3: } Dh_1 + Dh_2 + Eh_3 + b_3.
\end{aligned}$$

두 식이 성분별로 일치하므로 $L(g \cdot h) = g \cdot L(h)$가 성립한다. □ □

Pointwise nonlinearity σ는 $\sigma(\rho(g)z) = \rho(g)\sigma(z)$를 만족하므로, activation을 포함한 hidden layer $h^{\tau+1} = \sigma(Wh^{\tau} + b)$ 역시 S_2-equivariant하다 (Ravanbakhsh, Schneider, and Poczos 2017).