MAGANet: Achieving Combinatorial Generalization by Modeling a Group Action

Geonho Hwang [1] **Jaewoong Choi** [1] **Hyunsoo Cho** [2] **Myungjoo Kang** [2]

Abstract

Combinatorial generalization refers to the ability to collect and assemble various attributes from diverse data to generate novel unexperienced data. This ability is considered a necessary passing point for achieving human-level intelligence. To achieve this ability, previous unsupervised approaches mainly focused on learning the disentangled representation, such as the variational autoencoder. However, recent studies discovered that the disentangled representation is insufficient for combinatorial generalization and is not even correlated. In this regard, we propose a novel framework for data generation that can robustly generalize under these distribution shift situations. Instead of representing each data, our model discovers the fundamental transformation between a pair of data by simulating a group action. To test the combinatorial generalizability, we evaluated our model in two settings: Recombination-to-Element and Recombination-to-Range. The experiments demonstrated that our method has quantitatively and qualitatively superior generalizability and generates better images than traditional models.

1. Introduction

Whether a deep learning model can generalize to different distributions is a topic that is being researched widely (Shen et al., 2021). Traditional deep learning methods rely on the overly ideal assumption that training and test data are *i.i.d.* sampled from the same distribution. Although these models accomplish superior performance in the standard setting, the models tend to overfit training data and fail severely in the test dataset with different distributions (Montero et al., 2020; Schott et al., 2021). In other words, they have low generalizability under the distribution shift situation. Generative models and unsupervised representation learning (Schott et al., 2021; Montero et al., 2020) also suffer from the same problem. Especially *combinatorial generalization* (Vankov & Bowers, 2020) is one of the crucial problems that has drawn attention recently in the unsupervised representation field. It refers to the model's capacity to combinatorially combine the properties of two different data and create novel data that the model was not encountered through the learning process. For example, if a model, that has not experienced an image of a bearded woman while training, can generate the image by combining the attributes from an image of a bearded man and an image of a woman, we can say that the model has the combinatorial generalization capability. Because humans are innately capable of these sorts of tasks (Processing, 1986), the ability of deep learning models to freely extract and combine more abstract concepts is essential to achieve human-level capacities.

A generative model pursuing *disentangled representation* has long been regarded as one of the breakthroughs in solving the problem. A representation is called disentangled if the underlying generative factors of the data and the axis of latent representation encoded by the model have a correspondence (Eastwood & Williams, 2018); that is, a data variation occurred by a change of one generative factor should affect only one axis of latent representation and vice versa. Since a perfectly disentangled representation enables one to change each property independently by definition, it has been considered that a well-disentangled representation would accompany good combinatorial generalization capabilities. Unfortunately, as Montero et al. (2020) argued, it turns out that there is little correlation between the disentanglement score and the combinatorial generalization capacity. The model tended to have a high disentanglement score and low reconstruction error only on the training data.

On the other hand, various attempts have been made to disentangle models using the concept of group action (Yang et al., 2021; Quessard et al., 2020). Higgins et al. (2018) aim to derive the definition of disentangled representations using the correspondence relationship of group structures between data and representations. From this perspective, the

[1]Korea Institute for Advanced Study, Seoul, South Korea [2]Seoul National University, Seoul, South Korea. Correspondence to: Myungjoo Kang <mkang@snu.ac.kr>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

disentangled model means that a group acting in the latent space is decomposed as a direct product of subgroups so that each subgroup acts confined to corresponding axis of the latent space. However, most models assume the specific structure in which data are created by selecting one fixed data point, called a ***pivot***, and applying group actions to this pivot data. This model structure, encoding individual data to a latent variable, makes the model vulnerable to the distribution shift.

In this regard, we propose a novel generative framework MAGANet (Modeling A Group Action Network) to handle the Combinatorial generalization problem. Following the gist of group-based disentanglement (Higgins et al., 2018), we concentrate on the correspondence between the transformations and the symmetry groups. Unlike other research representing each data point to the latent space (Yang et al., 2021; Quessard et al., 2020), we focus on modeling the *transformation* itself between data. To model the transformation, we jointly train the encoder and the decoder like ordinary autoencoders and VAEs. The difference is that our encoder takes a pair of data as input, and encodes a difference to the latent variable representing an element of a group. The decoder simulates the group action acting on the data space. It takes data and an element of a latent group and transfers the input data to the target data following the group action learned through training. The decoder structure is designed to simulate the true transformation derived from the group action and be more flexible to the distribution shift of the dataset. This ability to simulate group actions is realized by hard-constraining the equivariance of group actions into the network structure. We use an invertible network as a building block to impart group equivariance to the network. We evaluated our model on two distribution shift settings, Recombination-to-Element and Recombination-to-Range. Quantitative and qualitative experiments show that our proposed method performs better combinatorial generalization. Our contributions are as follows:

- We proposed an approach called MAGANet, simulating the group structure and the group action. It is a novel generative framework that models a transformation between data.
- We proposed a group equivariant network structure that can learn a sufficiently wide range of nonlinear group actions. We utilized this network as a decoder for MAGANet.
- We quantitatively and qualitatively proved that the MAGANet achieved a substantially better combinatorial generalization capacity than VAE-based models in the experiments.
- We demonstrated that our model is robust in the selection of the pivot data, which proves the strong generalizability of the model.

2. Related Works

Disentangled Representation Recently, various attempts have been made to obtain a disentangled representation. Typical examples are variational autoencoders (VAE) (Kingma & Welling, 2013) and their variants. These methods, using a specific prior and KL-divergence term for the encoded latent variable, were considered one of the most effective ways to obtain a disentangled representation for several years. β-VAE (Higgins et al., 2016) added a coefficient to the KL divergence term to enhance the disentanglement effect. FactorVAE (Kim & Mnih, 2018) attempted to obtain better disentanglement by giving direct independence between latent codes using total correlation. In addition, several methods for evaluating disentanglement have been proposed. Eastwood & Williams (2018) proposed a DCI metric that measures disentanglement based on the degree to which latent variables explain generative factors. Chen et al. (2018) presented a disentanglement metric MIG that measures the gap in the value of mutual information between latent variables and generative factors with the largest mutual information and other latent variables.

However, VAE-based methods have limitations in that it is based on the statistical independence of latent codes. Locatello et al. (2020a) provided the theoretical results that any prior calculated as the Cartesian product of the function of each coordinate is not identifiable with respect to the rotation, so it is impossible to get disentangled representation without some inductive bias. Based on this result, several studies investigated the condition of weak supervision where the model can get the disentangled representation. Shu et al. (2019) showed that restricted labeling, match pairing, and rank pairing are sufficient conditions for disentangled representation. Locatello et al. (2020b) also demonstrated that the training with paired data whose latent factors differ only by a few generative factors ensures the identifiability of the model.

Group Based Disentanglement Higgins et al. (2018) reestablished the definition of disentanglement as a homomorphic relationship and correspondence between subgroups of a group and generative factors of the data. Accordingly, several follow-up papers were presented. Yang et al. (2021) presented a general method to grouptify VAE models using a dihedral group. Quessard et al. (2020) parametrized a special orthogonal group and utilized it as a structure of the latent space to model more expansive data space.

Combinatorial Generalization Vankov & Bowers (2020) first provided a concept of combinatorial generalization. Here, disentangled representation is considered a critical

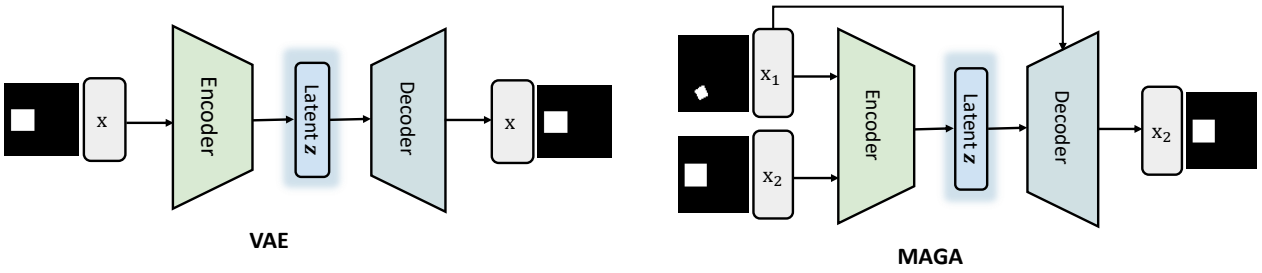


Figure 1. **Difference between standard VAE and Our Model.** The encoder of the standard VAE encodes the data itself, and the encoder of our model encodes the difference between the two data, while the vanilla VAE encoder takes a single data as the input. Likewise, the decoder of the standard VAE decodes the data from the latent variable, and the decoder of our model decodes the transformation from the input data to the output data, represented by the latent variable.

factor in achieving it. However, Montero et al. (2020) experimentally showed that disentanglement and combinatorial generalization have low correlation, and the model with even perfect disentanglement could have poor generalizability. Similarly, Schott et al. (2021) conducted more extensive experiments and showed that any model could not understand the underlying mechanism.

The concept of counterfactual synthesis exists as a very similar task or a task with a different name to combinatorial generalization. It is a task that generates realistic data that may not exist in the real world. The Structural Causal Model (SCM) (Kocaoglu et al., 2017; Thiagarajan et al., 2021; Sauer & Geiger, 2021) is one proposed way to achieve the goal using a causal mechanism. However, the methods suffer from the inflexibility of the prior SCM and the expensive cost of identifying all the causalities in the data. To overcome the limitation, Feng et al. (2022) devised the method using the pre-trained generative model and the distribution of the target attributes. However, the model still has limitations in that it requires a pre-trained model and attribute classifier. To overcome this, we presented a model that efficiently performs combinatorial generalization in a fully unsupervised setting.

3. Backgrounds

We will briefly introduce the preliminaries in this section. From now on, we denote the data space, such as the set of images as $\mathcal{X}$ and its latent representation space as $\mathcal{Z}$. In this paper, we treat the image space $\mathcal{X} = \mathbb{R}^{C\times W\times H}$ and the Euclidean space $\mathcal{Z} = \mathbb{R}^d$.

Variational Autoencoder VAE (Kingma & Welling, 2013) is a representation learning method based on likelihood maximization. VAE mainly consists of two parts, an encoder and a decoder. The encoder takes input x from the data space $\mathcal{X}$ and maps it to a distribution $q(z|x)$ on the latent space $\mathcal{Z}$. The decoder takes input from $z \in \mathcal{Z}$ and matches it to the original data x. The entire process is trained via maximizing Evidence Lower Bound(ELBO).

Group and Group Action A group is one of the most fundamental and ubiquitous structures in all areas. Mathematically, a group $(G, \cdot)$ is a set G equipped with a binary operation $\cdot$ following three axioms (Lang, 2012).

(Identity) $\exists\, e \in G$ such that $\forall g \in G, g \cdot e = e \cdot g = g$

(Inverse) $\forall g \in G, \exists\, g^{-1} \in G$ such that $g \cdot g^{-1} = g^{-1} \cdot g = e$

(Associativity) $\forall g_1, g_2, g_3 \in G,\ (g_1 \cdot g_2) \cdot g_3 = g_1 \cdot (g_2 \cdot g_3)$

A group can act on a space $\mathcal{X}$ with a function $\alpha : G \times \mathcal{X} \to \mathcal{X}$. An action of an element g of the group G on the set $\mathcal{X}$ is the transformation, $g \cdot : \mathcal{X} \to \mathcal{X}$, defined as $g \cdot x := \alpha(g, x)$. Group action must satisfy the homomorphic relation between the group and the group of transformations, that is,

$$\forall g, h \in G,\ \forall x \in \mathcal{X},\ g \cdot (h \cdot x) = (g \cdot h) \cdot x. \qquad (1)$$

Group action, resembling the transformation properties of the world, is considered that it has significant importance in learning disentangled representation (Higgins et al., 2018). In terms of group and group action, we can interpret the latent space $\mathcal{Z}$ of the standard VAE as a group, and the data x is generated by the group action $g \cdot x_0$ for some $g \in \mathcal{Z}$ and the fixed pivot data point $x_0 \in \mathcal{X}$. The encoder and the decoder also can be interpreted as a function that maps data x to the corresponding group element g and vice versa.

4. Methods

Motivation Previous unsupervised representation learning methods tend to consider that the latent space has a one-to-one correspondence to the data space. For example, autoencoder and VAE encode the data to a latent variable

and decode it to the same data again. If the latent space has a certain group structure, it is equivalent to assuming that the data x is generated as $x = g \cdot x_0$, for some x_0 common for all data. However, this approach is structurally vulnerable to the out-of-distribution data because the model directly matches the data x to the element g in the underlying group structure. If the model could not access the data, the model has no chance to learn the corresponding group element; hence the decoder also could not generate the correct reconstruction.

Transformation Encoding To overcome the problem and achieve combinatorial generalization, we present the method which learns the *transformation* of the data, not the data itself. Unlike the existing model, we map a pair of data x, x' to an element of a latent group g, which satisfies the group action relation $x' = g \cdot x$. We will call x' as the template data. And we also let the model learn how g acts in the data space. For example, consider a dataset with two generative factors A and B, which can have two values, 0 and 1, respectively. If the model has access to the training data $(\text{FactorA} = 0, \text{FactorB} = 0)$, $(\text{FactorA} = 0, \text{FactorB} = 1)$, $(\text{FactorA} = 1, \text{FactorB} = 0)$, it can acquire the group action to make the first component larger by pairing $(\text{FactorA} = 0, \text{FactorB} = 0)$ and $(\text{FactorA} = 1, \text{FactorB} = 0)$. Then, by applying the group action to $(\text{FactorA} = 0, \text{FactorB} = 1)$, we can get $(\text{FactorA} = 1, \text{FactorB} = 1)$ that the model has not seen before.

The framework is embodied by combining an encoder and decoder with a special structure. For data space $\mathcal{X}$ and latent space $\mathcal{Z}$, the encoder $E : \mathcal{X} \times \mathcal{X} \to \mathcal{Z}$ takes a pair of data $(x_1, x_2) \in \mathcal{X} \times \mathcal{X}$ and outputs a latent variable $z \in \mathcal{Z}$. The output latent variable z is considered an element of group structure and is supposed to represent the transformation that changes x_1 to x_2:

$$x_2 = z \cdot x_1 \quad \text{where} \quad z = E(x_1, x_2). \tag{2}$$

We also suppose that data space $\mathcal{X}$ is generated by group action $\alpha : G \times X \to X$ transitively and freely; in that case, for an arbitrary pair of a data point (x_1, x_2), there exists a *unique* $g \in G$ such that $g \cdot x_1 = x_2$ and the encoder is supposed to finds such an element z corresponding to g. The decoder $D : \mathcal{Z} \times \mathcal{X} \to \mathcal{X}$ learns the group action $\alpha : \mathcal{Z} \times \mathcal{X} \to \mathcal{X}$, so it takes a latent variable $z \in \mathcal{Z}$ and data $x \in \mathcal{X}$ as input.

4.1. Model Architecture

Encoder The encoder takes a pair of data as input. For the ordinary encoder $\overline{E} : \mathcal{X} \to \mathcal{Z}$ used in the VAE, $\overline{E}$ takes an data as input and outputs a sample from a Gaussian distribution with a mean and a variance parametrized by the encoder. $\overline{E}(x) = \mu + \epsilon\sigma$ where $\epsilon \sim \mathcal{N}(0, I)$. For the

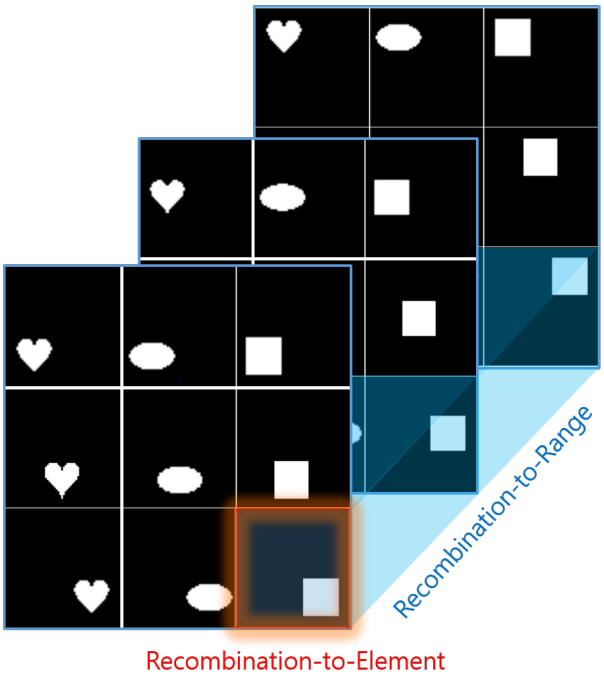


Figure 2. **Concept of the Recombination-to-Element(Red) and the Recombination-to-Range(Blue)** in dSprites. The test dataset of the Recombination-to-Range setting contains all square images on the right side of the image, regardless of other generative factors(position-y of the sprites in the figure). On the other hand, the test dataset of the Recombination-to-Element contains images with a square located in the lower right corner because it contains only one combination of all generative factors.

$x_1, x_2 \in \mathcal{X}$, let $\overline{E}(x_1) = z_1 = \mu_1 + \epsilon_1\sigma_1$ and $\overline{E}(x_2) = z_2 = \mu_2 + \epsilon_2\sigma_2$. We define the entire encoder $E : \mathcal{X} \times \mathcal{X} \to \mathcal{Z}$ as $E(x_1, x_2) := z_2 - z_1$. We used the same architecture in Burgess et al. (2018) for $\overline{E}$, but any other model can be used. Detailed architectures are in Appendix A.

The encoder defined in this way satisfies all axioms of the group. For example, in the case of inverse axiom, assume that x_2 is equal to $g \cdot x_1$. Then, because $g^{-1} \cdot x_2 = x_1$, encoded values with input (x_1, x_2) and (x_2, x_1) should be in an inverse relation. As $E(x_1, x_2) = \overline{E}(x_2) - \overline{E}(x_1) = -\left(\overline{E}(x_1) - \overline{E}(x_2)\right) = -E(x_2, x_1)$, the relation is satisfied, and a similar argument holds for all other axioms.

Decoder The decoder takes data and a latent variable as input. And, it must satisfies the property (Eq 1) of the group action. To implement this, we used an invertible neural network. An invertible neural network is an neural network which is invertible as a function. In other words, if f is an invertible neural network, the inverse f^{-1} exists, so both $f \circ f^{-1}$ and $f^{-1} \circ f$ become the identity function. Typically, normalizing flow (Rezende & Mohamed, 2015) commonly uses an invertible neural network to match data distribution

and prior.

For an invertible function $f : \mathcal{X} \to \mathcal{X}'$, we let the decoder $D : \mathcal{Z} \times \mathcal{X} \to \mathcal{X}$ be as follows:

$$D\left(z, x\right) := f^{-1} \circ \rho_{\mathcal{X}'}\left(z\right) \circ f\left(x\right), \tag{3}$$

where $\rho_{\mathcal{X}'}(z)$ is a group action acting on $\mathcal{X}'$. Then, we can check that the defined decoder satisfies the homomorphic property of the group action (Eq 1):

$$D\left(z_2, D\left(z_1, x\right)\right) = D\left(z_2 \cdot z_1, x\right). \tag{4}$$

This is because f and f^{-1} are canceled out, and the following equation holds.

$$\begin{aligned} D(z_2, D\left(z_1, x\right)) &= \left(f^{-1} \circ \rho_{\mathcal{X}'}(z_2) \circ f\right)\left(D\left(z_1, x\right)\right) && (5)\\ &= f^{-1} \circ \rho_{\mathcal{X}'}\left(z_2\right) \circ f \circ f^{-1} \circ \rho_{\mathcal{X}'}\left(z_1\right) \circ f\left(x\right) && (6)\\ &= f^{-1} \circ \rho_{\mathcal{X}'}\left(z_2\right) \circ \rho_{\mathcal{X}'}\left(z_1\right) \circ f\left(x\right) && (7)\\ &= f^{-1} \circ \rho_{\mathcal{X}'}\left(z_2 \cdot z_1\right) \circ f\left(x\right) = D\left(z_2 \cdot z_1, x\right). && (8) \end{aligned}$$

Because z is in the Euclidean space $\mathcal{Z} = \mathbb{R}^d$ in our setting, we implement the group action $\rho_{\mathcal{X}'}(z) : \mathcal{X}' \to \mathcal{X}'$ as an affine transformation:

$$\rho_{\mathcal{X}'}\left(z\right)\left(x'\right) = x' + Mz, \tag{9}$$

where M is the matrix $\mathbb{R}^{n \times d}$ and Mz is the matrix multiplication. Here, n is the dimension of the $\mathcal{X}$ and $\mathcal{X}'$, and M is set to be trainable. Although we only deal with the Euclidean space as the group, all groups with parametrizable group actions, such as a special orthogonal group with matrix multiplication, can be utilized.

To train this network, we need to be able to differentiate the function in both forward and backward directions. We adapt some network structures of Glow (Kingma & Dhariwal, 2018) as the invertible function f described above. It consists of ActNorms, invertible 1×1 convolutions, and affine coupling layers. Detailed architecture can be found in Appendix A.

Our decoder $D(z, \cdot)$ has a group-equivariant structure between $\mathcal{Z}$ and $\mathcal{X}$. In other words, $\mathcal{Z}$ acts on $\mathcal{Z}$ by $\cdot$ operation and on $\mathcal{X}$ by $D(z, \cdot)$. There are several previous studies (Winter et al., 2022) that suggest neural network architectures with group equivariance. However, our model has advantanges that differentiate it from existing networks. Existing studies have mainly aimed at constructing an equivariant network for group representation. In these approaches, the constructed equivariance is limited to the group actions that act linearly in spaces. On the other hand, the network using our invertible network can approximate any group action, guaranteed by the universality of invertible networks (Teshima et al., 2020).

4.2. Losses

VAE Loss Like an VAE framework, the encoder and the decoder should be adjusted so that the encoded latent variable should reconstruct the data again. In our framework, it is interpreted as the encoder E first estimate a group element g as $g \approx E(x_1, x_2)$, and then the decoder D simulates the group action so as to map $g \cdot x_1 \approx D(g, x_1) = D(E(x_1, x_2), x_1)$ to x_2 again. Regarding this, we give the reconstruction constraint $\mathcal{L}_{\text{recon}}$ as follows:

$$\mathcal{L}_{\text{recon}} = l_{\mathcal{X}}\left(D\left(E\left(x_1, x_2\right), x_1\right), x_2\right), \tag{10}$$

where $l_{\mathcal{X}}$ is the loss in the image space. We use the binary cross entropy loss as $l_{\mathcal{X}}$ in this paper. For the encoded mean and variance, we also calculate the KL-divergence $\mathcal{L}_{KL}$ with the normal isotropic Gaussian prior.

Latent Reconstruction Loss Unlike the ordinary autoencoder framework, the loss $\mathcal{L}_{\text{recon}} = l_{\mathcal{X}}(D(E(x_1, x_2), x_1), x_2)$ is insufficient to induce the autoencoder to learn grouptified representations. The encoder and the decoder can bypass the grouptified representation by ignoring x_1 and treating only x_2 the way usual autoencoders use it. That is, if the encoder $E' : \mathcal{X} \to \mathcal{Z}$ and the decoder $D' : \mathcal{Z} \to \mathcal{X}$ satisfy the equation $D'(E'(x)) = x$, the encoder $E(x_1, x_2) = E'(x_2)$ and the decoder $D(z, x_1) = D'(z)$ also satisfy the equation $D(E(x_1, x_2), x_1) = x_2$. To prevent this problem and guarantee the injectivity of the simulated group action, we impose the following natural restriction $\mathcal{L}_{\text{recon\_latent}}$ on the model.

$$\mathcal{L}_{\text{recon\_latent}} = d_{\mathcal{Z}}\left(E\left(x, D\left(z, x\right)\right), z\right), \tag{11}$$

where $d_{\mathcal{Z}}$ denotes the distance for measuring the difference in the latent space. For an arbitrary latent variable z and data x, $D\left(z, x\right)$ denotes a sample generated by acting z on x. Then, $E\left(x, D\left(z, x\right)\right)$ should represent the group element, which is required to transform x to $D\left(z, x\right)$. Therefore, $\mathcal{L}_{\text{recon\_latent}}$ should be close to zero naturally. We use the L_1 norm as $d_{\mathcal{Z}}$ in this paper.

In summary, the final loss becomes

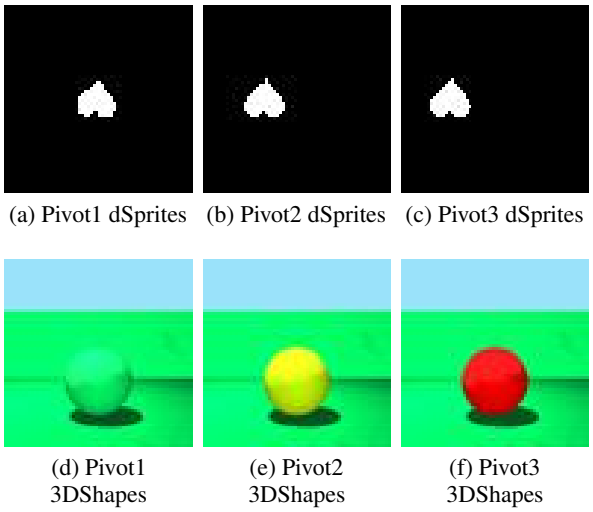
$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \beta_{KL}\mathcal{L}_{KL} + \beta_{\text{recon\_latent}}\mathcal{L}_{\text{recon\_latent}}. \tag{12}$$

Here, βs are the coefficients deciding the strength of regularization.

5. Experiments

5.1. Dataset

We evaluate our model on two datasets, the dSprites dataset (Matthey et al., 2017) and the 3D Shapes dataset (Burgess & Kim, 2018). The dSprites dataset consists of gray-scale



(a) Pivot1 dSprites (b) Pivot2 dSprites (c) Pivot3 dSprites

(d) Pivot1 3DShapes (e) Pivot2 3DShapes (f) Pivot3 3DShapes

Figure 3. **Pivot images of the datasets.** The pivot data from the dSprites have the generative factors [shape=*heart*, position-x $= \alpha$, position-y $= 0.48$, rotation $= 180°$, scale $= 0.7$], where α is (a) 0.48, (b) 0.16, and (c) 0.02. The pivot data from the 3D shapes have generative factors [floor-hue $= 0.4$, wall-hue $= 0.4$, object-hue $= \alpha$, object-shape$=$ sphere, object-scale$=1$, object-orientation$=0$], where α is (d) 0.4, (e) 0.2, and (f) 0.0.

sprite images. Each image is constructed with five generative factors: shape, scale, orientation, position-x, and position-y. The dataset has every combination of the five attributes, so the entire number of images is $3 \times 6 \times 40 \times 64 \times 64 = 737,280$. The 3D Shapes dataset is the dataset of the color images depicting the three-dimensional arrangement of the object. Each image is constructed with six generative factors: floor-hue, wall-hue, object-hue, object-shape, object-scale, and object-orientation.

5.2. Combinatorial Generalization

To test the combinatorial generalization property of the model, we measure the reconstruction error while separating the training and test data. Following the evaluation protocol of Montero et al. (2020), we evaluate the combinatorial generalization under two settings, the *Recombination-to-Element* and the *Recombination-to-Range*. As we can see in Fig 2, both settings mutually exclusively split the entire dataset into training and test dataset to conduct the evaluation of specific generalization tasks. A model is trained with the training dataset and is evaluated with the test dataset. Because the model did not experience the data of the test dataset, various generalization abilities can be evaluated depending on how the data is divided.

The Recombination-to-Element is the setting where all training data is available to the model while training except the only one combination of all generative factors. In the dSprites case, all data is in the training dataset except the case [shape=*ellipse*, position-x ≥ 0.6 , position-y ≥ 0.6, $120° \leq$ rotation $\leq 240°$, scale < 0.6]. Recombination-to-Element is the easiest of the two settings. Next, the Recombination-to-Range excludes a combination of the two generative factors regardless of other factors. In dSprites case, training data is all the data except the case, [shape=*square*, position-x> 0.5]. The Recombination-to-Range is a much more difficult task than the Recombination-to-Element in the sense that the model can not access the specific combination of the two generative factors at all. Performing well in the Recombination-to-Range setting is essential for combinatorial generalization.

Similar to dSprites, the Recombination-to-Element setting of 3D shapes has a test dataset with generative factors [floor-hue > 0.5, wall-hue > 0.5, object-hue > 0.5, object-shape=cylinder, object-scale$= 1$, object-orientation$= 0$], and the Recombination-to-Range setting has a test dataset with generative factors [object-hue ≥ 0.5 (cyan), object-shape $=$ oblong] .

Table 1. **BCE Reconstruction Error(↓) on dSprites.** Our method demonstrated a significantly better performance than other models in the Recombination-to-Range(R2Range) setting and similar or better performance in the Recombination-to-Element(R2Element) setting.

Method	R2Element	R2Range
VAE	**8.05**	200.35
β-VAE ($\beta = 8$)	24.62	215.95
β-VAE ($\beta = 12$)	24.91	154.90
Factor-VAE ($\gamma = 20$)	24.62	130.56
Factor-VAE ($\gamma = 50$)	22.58	153.98
Factor-VAE ($\gamma = 100$)	24.88	100.60
MAGANet(Ours)	8.25	**49.74**

Table 2. **BCE Reconstruction Error(↓) on 3D shapes.** Our method demonstrated a significantly better performance than other models in the Recombination-to-Element(R2Element) and the Recombination-to-Range(R2Range) setting.

Method	R2Element	R2Range
VAE	3,923	4,294
β-VAE ($\beta = 8$)	3,927	4,482
β-VAE ($\beta = 12$)	3,940	5,077
Factor-VAE ($\gamma = 20$)	3,935	4,602
Factor-VAE ($\gamma = 50$)	3,943	5,275
Factor-VAE ($\gamma = 100$)	3,958	5,095
MAGANet(Ours)	**3,902**	**3,582**

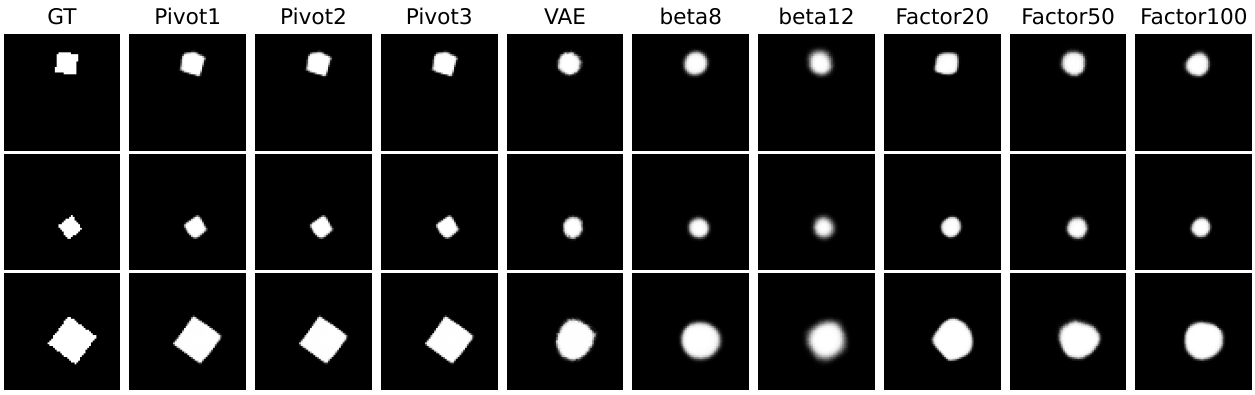


Figure 4. **Reconstruction images on the Recombination-to-Range setting in dSprites.** Ground truth images are sampled from the test dataset with generative factors [shape=*square*, position-x> 0.5]. VAE-based models tend to generate blob whenever exposed to an unseen data situation. On the other hand, our method generates an exact square sprites image located on the right side of the image regardless of the selection of the pivot data.

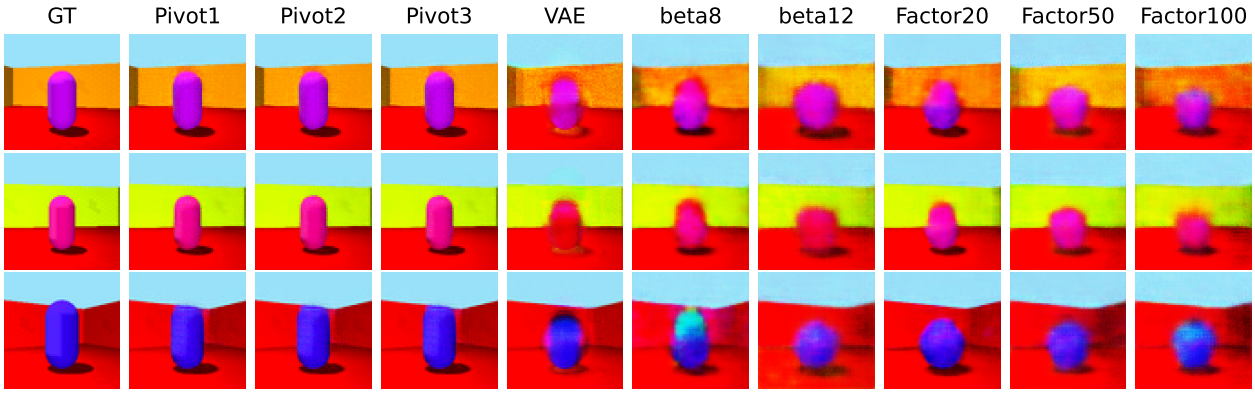


Figure 5. **Reconstruction images on the Recombination-to-Range setting in 3D Shapes.** Ground truth images are sampled from the test dataset with generative factors [object-hue ≥ 0.5 (cyan), object-shape $=$ oblong]. Similar to the dSprites dataset, VAE-based models can not generate the combination of the oblong shape and the object hue that was not provided in the training dataset. On the other hand, our method generates exact oblong shape images.

Pivot Data Unlike the previous VAE models, our decoder of the model takes a pair of images as the input. For a fair evaluation, we need to make a pair of the pivot image and a test image. The pivot image is the fixed image from the training dataset and is paired to all data from the test dataset so that the comparison with the existing method can be conducted. Because the decoder model always takes the same input from the pivot image as a template, the results are can be influenced by the selection of the pivot data. Therefore, we made the best possible effort to fairly select pivot data. The generative factors, that show the same distribution on train and test datasets, are selected near the median value of the range of values. It includes generative factors such as position-y in the dSprites and wall hue in the 3D shapes. For the generative factors that serve as criteria for dividing data, we conducted the ablation study on the selecting the value for pivot data.

For the dSprites dataset, the pivot image is the image with the generative factors [shape=*heart*, position-x $= \alpha$, position-y $= 0.48$, rotation $= 180^\circ$, scale $= 0.7$]. We select the generative factors except for the shape and the position-x as the value near to median because the generalization tasks are defined according to the combination of shape and position-x. Position-x is chosen from $\alpha \in [0.02, 0.16, 0.48]$, and unless otherwise noted, α is 0.16. Similarly, the pivot for the 3D Shape is set to [floor-hue $= 0.4$, wall-hue $= 0.4$, object-hue $= \alpha$, object-shape= sphere, object-scale= 1, object-orientation= 0]. Object hue is chosen from $\alpha \in [0.0, 0.2, 0.4]$ and unless otherwise noted α is 0.2. The pivot data for both datasets can be found in Figure 3.

5.3. Experiment Settings

We adopted the experiment setting from the one from VAE for both datasets. The optimizer is Adam, with a learning rate of 0.0005. The dimension of the latent variable is set to 10 and the batch size is 64. For the regularizing coefficients β, we set the value to $\beta_{\text{recon\_latent}} = 300$, and $\beta_{\text{KL}} = 300$. We trained 100 epochs for both datasets three times and took the model with the best binary cross entropy loss model.

The results were compared with the six models listed in Montero et al. (2020), VAE, β-VAE with $\beta = 8$, β-VAE with $\beta = 12$, Factor-VAE with $\gamma = 20$, Factor-VAE with $\gamma = 50$, and Factor-VAE with $\gamma = 100$.

5.4. Results

We conducted the reconstruction evaluations on two datasets with each three pivot data and measured the binary cross entropy loss for the test dataset. Our results and the results of comparison group experiments from Montero et al. (2020) are summarized in Table 1 and Table 2. On dSprites, our method achieves a similar reconstruction loss to the best-performing VAE in the Recombination-to-Element, and significantly outperforms other models in the Recombination-to-Range (Table 1). On 3D shapes, our method attains the best reconstruction loss in both settings (Table 2). The result implies that the models successfully reconstruct the data in the test dataset.

For the qualitative result, we plot the reconstruction of the test dataset of the dSprites dataset in Fig 4 and the 3D shapes dataset in Fig 5. We observed that VAE-based models tend to generate blob near the generated image, ignoring rotation, scale, and shape in the dSprites data reconstruction. On the other hand, our method manages to generate exact square sprites images with almost the same shape as the ground truth data. For the 3D shapes dataset, the previous models severely failed in generating the exact shapes of the object, leading to a significantly large reconstruction loss, as opposed to our model restoring both hue and shape successfully.

5.5. Robustness for the Pivot Data Selection

To test the robustness under the pivot data variation, we selected three pivot images for each dataset and conducted the Recombination-to-Element and the Recombination-to-Range experiments for each pivot. Pivot data are different in position-x in the dSprites dataset and object-hue in the 3D shapes dataset. These generative factors are criteria for splitting training and test dataset, respectively. We remark that it is natural to think that the farther the pivot data is from the test dataset, the more the reconstruction becomes difficult. In this respect, reconstructing the test dataset with pivot3 is more challenging than reconstructing with pivot1.

Table 3. **Reconstruction Error(↓) on dSprites for different pivot images.** We can observe that the difference derived from selecting the pivot image is insignificant.

dSprites		
Piovt	**Recomb2Element**	**Recomb2Range**
Pivot1	7.61	47.68
Pivot2	7.11	48.41
Pivot3	8.26	49.74
3D Shapes		
Piovt	**Recomb2Element**	**Recomb2Range**
Pivot1	3,900	3,588
Pivot2	3,900	3,573
Pivot3	3,901	3,574

The test dataset has the generative factors of [shape=*square*, position-x>0.5]. Hence, the pivot3 with position-x=0.02 is much farther from the test dataset than the pivot1 with position-x=0.48. Nevertheless, Table 3 demonstrates that the overall reconstruction loss is similar for all the pivot data regardless of the value of position-x. This result implies the strong generalizability of our model.

5.6. Validity of the Encoder

We conducted the experiment to check whether the encoded latent variables represent *the transformation* between data. If the encoder represents the transformation, then when the same group action is applied to different data, they should be encoded into identical latent vector. In other words, the equation $E(x, gx) = E(x', gx')$ should be satisfied for distinct x and x'.

The experiment settings are as follows. Let x_1 be an image with generative factor [position-x = 0.5 , position-y = 0.5], and x_2 be the image with generative factor [position-x = 0.6 , position-y = 0.5], and all other factors of x_2 are the same as x_1. Similarly, x_3 is the image with [position-x = 0.7 , position-y = 0.5], and x_2' is the image with [position-x = 0.5 , position-y = 0.6]. We encoded pairs of images to latent variables as $z_{12} = E(x_1, x_2)$, $z_{23} = E(x_2, x_3)$, and $z'_{12} = E(x_1, x_2')$. Note that while x_2 and x_3 differ in position-x from x_1, x_2' has a different position-y from x_1.

If E encodes the transformation between a pair of images regardless of the initial image, z_{12} would be far more different from z'_{12} than z_{23}. In this regard, we compared the cosine similarity of (z_{12}, z'_{12}) and (z_{12}, z_{23}). We evaluated the coherence for all combinations of other generative factors, i.e., shape, scale, and orientation. Specifically, there are 720 pairs of z_{12} and z_{23} representing the same x-translation from different initial images. We measured the cosine simi-

larity for each pair of (z_{12}, z_{23}) and computed the average similarity of all pairs. The overall average cosine similarity between z_{12} and z_{23} is 0.921. In contrast, the average cosine similarity between z_{12} and z'_{12} is 0.061, which is significantly lower than 0.921. This result demonstrates a strong coherence of latent variables that encodes the same transformation.

6. Conclusion

In this paper, we proposed the novel generative framework MAGANet capable of the combinatorial generalization task. It was confirmed that MAGANet stably showed significantly better performance than the existing models qualitatively and quantitatively. While our primary focus in this paper was not explicitly on the disentanglement property of the model, we strongly believe that the framework possesses substantial potential for achieving a robust disentanglement property. This is because disentanglement is a concept inherently connected to the transformation of data, rather than merely the embedding of individual data instances. To the best of our knowledge, MAGANet is the first attempt to bridge the gap between combinatorial generalization and data interpretation by leveraging the group action theory. It is important to note that our model evaluation is currently limited to a toy dataset. Nevertheless, we anticipate that our study will serve as a valuable baseline for future research in this domain.

Acknowledgement

This work was supported by a KIAS Individual Grant [AP087501, AP092801] via the Center for AI and Natural Sciences at Korea Institute for Advanced Study, the NRF grant[2012R1A2C3010887], and the MSIT/IITP([1711117093], [2021-0-00077], [No. 2021-0-01343, Artificial Intelligence Graduate School Program(SNU)]).

References

Burgess, C. and Kim, H. 3d shapes dataset. https://github.com/deepmind/3dshapes-dataset/, 2018.

Burgess, C. P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G., and Lerchner, A. Understanding disentangling in β-vae. *arXiv preprint arXiv:1804.03599*, 2018.

Chen, R. T., Li, X., Grosse, R. B., and Duvenaud, D. K. Isolating sources of disentanglement in variational autoencoders. *Advances in neural information processing systems*, 31, 2018.

Eastwood, C. and Williams, C. K. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*, 2018.

Feng, R., Xiao, J., Zheng, K., Zhao, D., Zhou, J., Sun, Q., and Zha, Z.-J. Principled knowledge extrapolation with gans. *arXiv preprint arXiv:2205.13444*, 2022.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016.

Higgins, I., Amos, D., Pfau, D., Racaniere, S., Matthey, L., Rezende, D., and Lerchner, A. Towards a definition of disentangled representations. *arXiv preprint arXiv:1812.02230*, 2018.

Kim, H. and Mnih, A. Disentangling by factorising. In *International Conference on Machine Learning*, pp. 2649–2658. PMLR, 2018.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kocaoglu, M., Snyder, C., Dimakis, A. G., and Vishwanath, S. Causalgan: Learning causal implicit generative models with adversarial training. *arXiv preprint arXiv:1709.02023*, 2017.

Lang, S. *Algebra*, volume 211. Springer Science & Business Media, 2012.

Locatello, F., Bauer, S., Lucic, M., Rätsch, G., Gelly, S., Schölkopf, B., and Bachem, O. A sober look at the unsupervised learning of disentangled representations and their evaluation. *arXiv preprint arXiv:2010.14766*, 2020a.

Locatello, F., Poole, B., Rätsch, G., Schölkopf, B., Bachem, O., and Tschannen, M. Weakly-supervised disentanglement without compromises. In *International Conference on Machine Learning*, pp. 6348–6359. PMLR, 2020b.

Matthey, L., Higgins, I., Hassabis, D., and Lerchner, A. dsprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/, 2017.

Montero, M. L., Ludwig, C. J., Costa, R. P., Malhotra, G., and Bowers, J. The role of disentanglement in generalisation. In *International Conference on Learning Representations*, 2020.

Processing, P. D. Explorations in the microstructure of cognition. *Volume 1: Foundations*, 1986.

Quessard, R., Barrett, T., and Clements, W. Learning disentangled representations and group structure of dynamical environments. *Advances in Neural Information Processing Systems*, 33:19727–19737, 2020.

Rezende, D. and Mohamed, S. Variational inference with normalizing flows. In *International conference on machine learning*, pp. 1530–1538. PMLR, 2015.

Sauer, A. and Geiger, A. Counterfactual generative networks. *arXiv preprint arXiv:2101.06046*, 2021.

Schott, L., Von Kügelgen, J., Träuble, F., Gehler, P., Russell, C., Bethge, M., Schölkopf, B., Locatello, F., and Brendel, W. Visual representation learning does not generalize strongly within the same domain. *arXiv preprint arXiv:2107.08221*, 2021.

Shen, Z., Liu, J., He, Y., Zhang, X., Xu, R., Yu, H., and Cui, P. Towards out-of-distribution generalization: A survey. *arXiv preprint arXiv:2108.13624*, 2021.

Shu, R., Chen, Y., Kumar, A., Ermon, S., and Poole, B. Weakly supervised disentanglement with guarantees. *arXiv preprint arXiv:1910.09772*, 2019.

Teshima, T., Ishikawa, I., Tojo, K., Oono, K., Ikeda, M., and Sugiyama, M. Coupling-based invertible neural networks are universal diffeomorphism approximators. *Advances in Neural Information Processing Systems*, 33:3362–3373, 2020.

Thiagarajan, J., Narayanaswamy, V. S., Rajan, D., Liang, J., Chaudhari, A., and Spanias, A. Designing counterfactual generators using deep model inversion. *Advances in Neural Information Processing Systems*, 34:16873–16884, 2021.

Vankov, I. I. and Bowers, J. S. Training neural networks to encode symbols enables combinatorial generalization. *Philosophical Transactions of the Royal Society B*, 375 (1791):20190309, 2020.

Winter, R., Bertolini, M., Le, T., Noe, F., and Clevert, D.-A. Unsupervised learning of group invariant and equivariant representations. In *Advances in Neural Information Processing Systems*, 2022.

Yang, T., Ren, X., Wang, Y., Zeng, W., and Zheng, N. Towards building a group-based unsupervised representation disentanglement framework. In *International Conference on Learning Representations*, 2021.

A. Architecture

We use the architecture from Burgess et al. (2018) as the encoder. The structure is as follows.

Table 4. **The Encoder Architecture.**

4×4 convolution with 32 channels
ReLU
4×4 convolution with 32 channels
ReLU
4×4 convolution with 32 channels
ReLU
4×4 convolution with 32 channels
ReLU
Fully connected layer with 256 nodes
ReLU
Fully connected layer with 256 nodes
ReLU
Fully connected layer with d nodes

We use the architecture from the Glow (Kingma & Dhariwal, 2018) as the decoder. The structure is as follows. First, the FlowStep module is defined as the building block. Then, we pile three layers of FlowStep modules with squeeze layer

Table 5. **The FlowStep Architecture.**

Layers
ActNorm
1×1 convolution without LU Decomposition
Additive Coupling Layer

attached above to construct the Flow module.

Table 6. **The Decoder Architecture.**

Flow module
Squeeze Layer with factor 2
FlowStep
FlowStep
FlowStep

Finally, three Flow modules are build to make the entire invertible network.

Table 7. **The FlowNet Architecture.**

FlowNet
Flow module
Flow module
Flow module