# MAGA: MODELING A GROUP ACTION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Combinatorial generalization, an ability to collect various attributes from diverse data and assemble them to generate novel unexperienced data, is considered an essential traversal point to achieve human-level intelligence. 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 proposed a novel framework of data generation that can robustly generalize under these distribution shift situations. The model, simulating the group action, carries out combinatorial generalization by discovering the fundamental transformation between the data. We conducted experiments on the 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 over traditional models.

## 1 INTRODUCTION

Whether a deep learning model can generalize to the distribution different from the training data is a topic that is being researched widely (Shen et al., 2021). Based on the overly ideal assumption that training and test data are *i.i.d.* sampled from the same distribution, traditional deep learning methods tend to overfit training data and fail severely in the test dataset (Montero et al., 2020; Schott et al., 2021), even if it accomplishes superior performance in the naive setting. 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 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 sort 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 calculated 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 (Yang et al., 2021; Quessard et al., 2020) disentangle models using the concept of group action. Higgins et al. (2018) aims to clarify the definition of disentangled representations using the homomorphic relationship of group structures between data and representations. However, most models assume the specific structure in which data

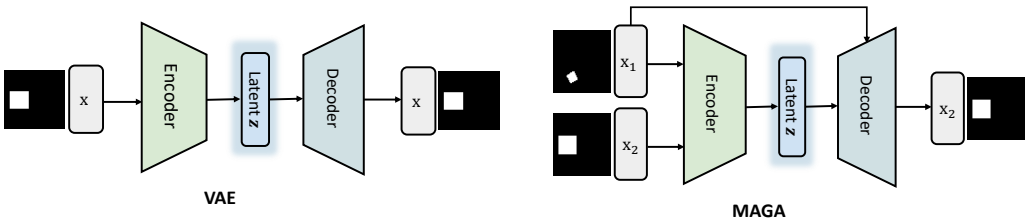



Figure 1: **Overview of VAE and Our Models.** The encoder of the standard VAE encodes the data itself, and the encoder of our model encodes the difference between the two data. Likewise, the decoder of the standard VAE decodes the data from the latent variable, and the decoder of our model decodes the transformation from the input data to the output data represented by the latent variable.

are created by defining one fixed data point, called pivot, and applying group actions to the 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 MAGA to handle the 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 papers focusing on a more general group structure (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 grouptified latent variable. Encoded values are regularized to follow the group axioms. 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. We lay out the entire structure so that the decoder can approximate the true transformation induced by the group action and be more flexible to the distribution shift of the dataset. Quantitative and qualitative experiments show that our proposed method performs better combinatorial generalization. Our contributions are as followings:

- We proposed a approaches simulating the group structure and the group action. It is the novel generative framework that models a transformation between data.
- We quantitatively and qualitatively proved that the methods showed a substantially better combinatorial generalization capacity than the VAE-based models in the experiments.
- We also 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. $\beta$-VAE (Higgins et al., 2016) adds 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 the latent variable 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. Accepting this result,

several studies investigated under what weakly supervised setting the model can get the disentangled representation. Shu et al. (2019) showed that restricted labeling, match pairing, and rank pairing are sufficient condition for disentangled representation. Locatello et al. (2020b) also demonstrates 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) re-established 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 the $SO(n)$ 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 and disentangled representation is considered a critical factor in achieving it. However, Montero et al. (2020) experimentally displayed 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 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 absurdly 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 an 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 an original data $x$. The entire process is trained using the loss called Evidence Lower Bound(ELBO).

**Group and Group Action** Group is one of the most fundamental and ubiquitous structures in all areas. Mathematically, 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 the 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 $\forall g, h \in G$, $\forall x \in \mathcal{X}$, $g \cdot (h \cdot x) = (g \cdot h) \cdot x$. Group action, resembling the transformation properties of the world, is considered that has significant importance in learning disentangled representation (Higgins et al., 2018). In terms of group and group action, the latent space $\mathcal{Z}$ of the standard VAE can be interpreted 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 map data $x$ to the corresponding group element $g$ and vice versa.

## 4 METHODS

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. In the case that the latent space has a certain group structure, it is the same to think 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. To overcome the problem and achieve combinatorial generalization, we provided 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 $x' = g \cdot x$. And we also let the model learn how $g$ to act 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 $(\text{Factor A} = 0, \text{Factor B} = 0)$, $(\text{Factor A} = 0, \text{Factor B} = 1)$, $(\text{Factor A} = 1, \text{Factor B} = 0)$, it can acquire the group action to make the first component larger by pairing $(\text{Factor A} = 0, \text{Factor B} = 0)$ and $(\text{Factor A} = 1, \text{Factor B} = 0)$. And by applying the group action to $(\text{Factor A} = 0, \text{Factor B} = 1)$, we can get $(\text{Factor A} = 1, \text{Factor B} = 1)$ that the model has not seen before.

The frame 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 $z \in \mathcal{Z}$. Output latent variable $z$ is considered an element of group structure and is supposed to represent the transformation that changes $x_1$ to $x_2$. Suppose 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. The decoder $D : \mathcal{Z} \times \mathcal{X} \to \mathcal{X}$ imitates 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.

**Encoder** The encoder takes a pair of data as input. For convenience, we concatenate images along the channel and treat them as double channel-sized images. For example, a batch of images with size $\mathbb{R}^{N \times C \times W \times H}$ is paired to $\mathbb{R}^{N \times 2C \times W \times H}$ and used as input to the encoder. Then, the encoder outputs a latent variable with size $\mathbb{R}^{N \times d}$. We used the same architecture in Burgess et al. (2018), but any other model can be used. Detailed architectures are in Appendix A.

**Decoder** The decoder takes data and a latent variable as input. We spread a latent variable of size $\mathbb{R}^{N \times d}$ to $\mathbb{R}^{N \times d \times W \times H}$ and concatenate it to an image of size $\mathbb{R}^{N \times C \times W \times H}$ along the channel. So the decoder becomes the function

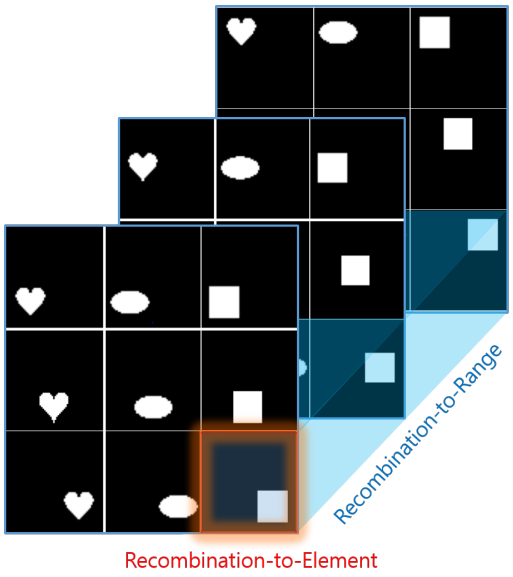



Figure 2: **Concept of the Recombination-to-Element(Red) and the Recombination-to-Range(Blue)** in dSprites. The test dataset of the recombination-to-Range setting contains all square images on the right side of the image, regardless of other generative factors(Position Y of the sprites in the figure). 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.

from $\mathbb{R}^{N\times(C+d)\times W\times H}$ to $\mathbb{R}^{N\times C\times W\times H}$. Now we can utilize the models used in the field of image transfer and choose the generator model used in Zhu et al. (2017). Detailed architecture can be found in Appendix A.

### 4.1 Loss

Now we introduce several losses that constrain the encoder and the decoder to satisfy the rules of the group and the group action.

**Reconstruction Loss** Like an ordinary autoencoder framework, the encoder and the decoder should be adjusted so that the encoded latent variable should be reconstructed to the data again. To interpret it as our framework, it means that the encoder $E$ first estimate a group element $g$ as $E(x_1, x_2)$, and then the decoder $D$ simulates the group action so as to map $(E(x_1, x_2), x_1)$ to $x_2$ again. Regarding this, we give the reconstruction constraint $\mathcal{L}_{\text{recon}} = l_{\mathcal{X}}(D(E(x_1, x_2), x_1), x_2)$, 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.

**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 for autoencoder frameworks. The encoder and the decoder can bypass the loss by ignoring $x_1$ and treating only $x_2$ the way traditional 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 on the model. For an arbitrary latent variable $z$ and data $x$, the pair of the decoded data $D(z, x)$ and the original data $x$ should be encoded to the original $z$. In other words, $\mathcal{L}_{\text{latent\_recon}} = d_{\mathcal{Z}}(E(x, D(z, x)), z)$ should be close to zero, where $d_{\mathcal{Z}}$ denotes the distance for measuring the difference in the latent space. We use the $L_1$ norm as $d_{\mathcal{Z}}$ in this paper.

**Latent Group Loss** Unlike the existing method, there is no reason our method's encoded representation should abide by the axioms of the group. To make the latent representation have a group structure, we impose the latent space on a group structure by regularizing the axioms of the group as a loss. Because there are three axioms in the group, we also defined three regularization losses, $\mathcal{L}_{\text{iden}}$, $\mathcal{L}_{\text{inv}}$, and $\mathcal{L}_{\text{assoc}}$, for identity, inverse and associativity axiom, respectively.

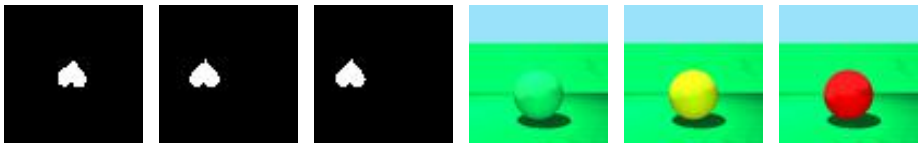
- To assert the identity axiom $E(x, x) = e$, we regularize using the loss $\mathcal{L}_{\text{iden}} = d_{\mathcal{Z}}(E(x, x), e)$.
- To assert the inverse axiom, that is, if $E(x_1, x_2) = g$, then $E(x_2, x_1) = g^{-1}$, we regularize using the loss $\mathcal{L}_{\text{inv}} = d_{\mathcal{Z}}(E(x_1, x_2) \cdot E(x_2, x_1), e)$.
- To assert the associativity axiom, that is, if $E(x_1, x_2) = g$ and $E(x_2, x_3) = h$, then $E(x_1, x_3) = h \cdot g$, we regularize using the loss $\mathcal{L}_{\text{assoc}} = d_{\mathcal{Z}}(E(x_2, x_3) \cdot E(x_1, x_2), E(x_1, x_3))$.

To evaluate the losses, we need to sample the data pair. For the batch of size $N$, $[x_1, x_2, \cdots, x_N]$, we sample the index to form a batch of pairs. For the identity case, the batch becomes $\text{Batch}_{iden} = [(x_1, x_1), (x_2, x_2), \cdots, (x_N, x_N)]$, and the loss is

$$\mathcal{L}_{\text{iden}} = \sum_{i=1}^{N} d_{\mathcal{Z}}(E(x_i, x_i), e). \tag{1}$$

For the inverse case, the two symmetric batches become $\text{Batch}_{inv1} = [(x_1, x_2), (x_3, x_4), \cdots, (x_{N-1}, x_N)]$ and $\text{Batch}_{inv2} = [(x_2, x_1), (x_4, x_3), \cdots, (x_N, x_{N-1})]$. And the loss becomes

$$\mathcal{L}_{\text{inv}} = \sum_{i=1}^{\frac{N}{2}} d_{\mathcal{Z}}(E(x_{2i}, x_{2i-1}) \cdot E(x_{2i-1}, x_{2i}), e). \tag{2}$$



(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 $\alpha$ 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 $\alpha$ is (d) 0.4, (e) 0.2, and (f) 0.0.

For the associativity case, we randomly sample the triplet $(x_{r(3i-2)}, x_{r(3i-1)}, x_{r(3i)})$ from the batch and test the associativity using the loss

$$\mathcal{L}_{\text{assoc}} = \sum_{i=1}^{\frac{N}{3}} d_{\mathcal{Z}}(E(x_{r(3i-2)}, x_{r(3i-1)}) \cdot E(x_{r(3i-1)}, x_{r(3i)}), E(x_{r(3i-2)}, x_{r(3i)})). \tag{3}$$

In all, the group loss becomes $\mathcal{L}_{\text{group}} = \mathcal{L}_{\text{iden}} + \mathcal{L}_{\text{inv}} + \mathcal{L}_{\text{assoc}}$.

In addition to the group loss, we observed that the model abused the loss to minimize the Latent Reconstruction Loss and the Latent Group Loss by collapsing all $z$ values near the zero point. To block this detour, we penalize the concentration of the batch of the latent values by forcing the variance to be larger than a certain threshold. In the formula, the loss becomes $\mathcal{L}_{\text{var}} = \max(0, 1 - \sum_{i=1}^{d} Var(z_i))$.

**Group Action Loss** Latent Group Loss has imposed the encoder to encode the difference between data as the group element. On the other hand, the decoder has not been regularized to utilize the proper group structure for modeling the group action. Applying axioms of the group similar to the latent group loss, we get the following losses.

- To regularize $D(e, x_1) = x_1$, we regularize using the identity action loss $\mathcal{L}_{\text{action\_iden}} = l_{\mathcal{X}}(D(e, x_1), x_1)$.
- To regularize $D(E(x_1, x_2)^{-1}, x_2) = x_1$, we regularize using the inverse action loss $\mathcal{L}_{\text{action\_inv}} = l_{\mathcal{X}}(D(E(x_1, x_2)^{-1}, x_2), x_1)$.
- To regularize $D(E(x_2, x_3) \cdot E(x_1, x_2), x_1) = x_3$, we regularize using the associativity action loss $\mathcal{L}_{\text{action\_assoc}} = l_{\mathcal{X}}(D(E(x_2, x_3) \cdot E(x_1, x_2), x_1), x_3)$.

To sum up, we get the group action loss $\mathcal{L}_{\text{action}} = \mathcal{L}_{\text{action\_iden}} + \mathcal{L}_{\text{action\_inv}} + \mathcal{L}_{\text{action\_assoc}}$.

All in all, the final loss becomes $\mathcal{L} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{action}} + \beta_{\text{recon\_latent}}\mathcal{L}_{\text{recon\_latent}} + \beta_{\text{group}}\mathcal{L}_{\text{group}} + \beta_{\text{var}}\mathcal{L}_{\text{var}}$. Here, $\beta$'s are the coefficients deciding the strength of group regularization.

## 5 Experiments

### 5.1 Dataset

We use two datasets in the experiments, the dSprites dataset (Matthey et al., 2017) and the 3D Shapes dataset (Burgess & Kim, 2018). The dSprites dataset consist of gray-scale 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.

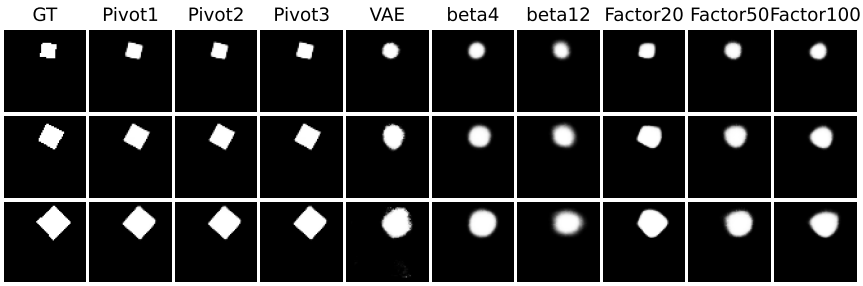



Figure 4: **Reconstruction images on the Recombination-to-Range setting in dSprites.** Ground truth images are sampled from the test dataset with generative factors [shape=*square*, position-x$> 0.5$]. VAE-based models tend to generate blob whenever exposed to an unseen data situation. On the other hand,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

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

| Method | Recomb2Element | Recomb2Range |
|---|---|---|
| VAE | **8.05** | 200.35 |
| $\beta$-VAE 8 | 24.62 | 215.95 |
| $\beta$-VAE 12 | 24.91 | 154.90 |
| Factor-VAE 20 | 24.62 | 130.56 |
| Factor-VAE 50 | 22.58 | 153.98 |
| Factor-VAE 100 | 24.88 | 100.60 |
| MAGA(Ours) | 10.54 | **62.56** |

### 5.2 Combinatorial Generalization

To test the combinatorial generalization property of the model, we evaluate 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. 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. For example, in the dSprites case, all data is in the training dataset except the case [shape=*ellipsis*, position-x $\geq 0.6$ , position-y $\geq 0.6$, $120^\circ \leq$ rotation $\leq 240^\circ$ , 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. For example, 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't access the specific combination of the two generative factors at all. Performing well in tbe 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 $\geq 0.5$ (cyan), object-shape $=$ oblong] .

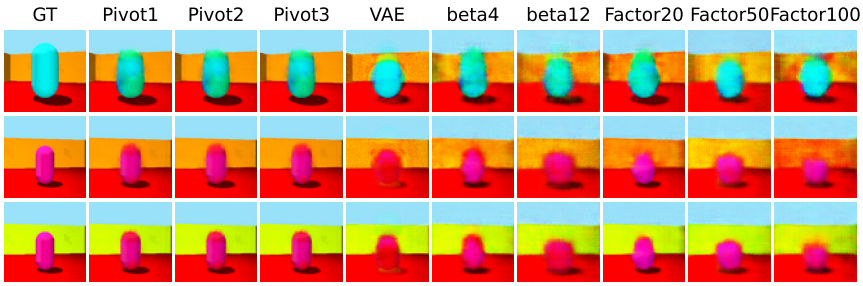



Figure 5: **Reconstruction images on the Recombination-to-Range setting in 3D Shapes.** Ground truth images are sampled from the test dataset with generative factors [object-hue $\geq 0.5$ (cyan), object-shape $=$ oblong]. Similar to the dSprites dataset, VAE-based models can not generate the combination 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.

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

| Method | Recomb2Element | Recomb2Range |
|---|---|---|
| VAE | 3,923 | 4,294 |
| $\beta$-VAE 8 | 3,927 | 4,482 |
| $\beta$-VAE 12 | 3,940 | 5,077 |
| Factor-VAE 20 | 3,935 | 4,602 |
| Factor-VAE 50 | 3,943 | 5,275 |
| Factor-VAE 100 | 3,958 | 5,095 |
| MAGA(Ours) | **3,902** | **3,959** |

**Pivot Data** Unlike the previous VAE models, our decoder of the model takes a pair of images as the input. To fairly evaluate our models to the existing models, we need to make up a pair using the pivot image and an image from the test dataset. 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 severely influenced by the selection of the pivot data. Therefore, we made the best possible effort in selecting pivot data. The generative factor that is not directly connected to the partition of the dataset, such as position Y in the dSprites and wall hue in the 3D shapes, are selected near the median value of the range of values. 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 $= 180°$ , scale $= 0.7$]. We select the generative factors except for the shape and the position-X as the value near to median. Position X is chosen from $\alpha \in [0.02, 0.16, 0.48]$, and unless otherwise noted, $\alpha$ 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 $\alpha$ 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 $\beta$, we set the value to $\beta_{\text{recon\_latent}} = 100$, $\beta_{\text{group}} = 100$, and $\beta_{\text{var}} = 300$. We trained $100$ epochs for both datasets three times and took the model with the best binary cross entropy loss model.

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

| **Datasets** | **dSprites** | | **3D Shapes** | |
|---|---|---|---|---|
| **Method** | **Recomb2Element** | **Recomb2Range** | **Recomb2Element** | **Recomb2Range** |
| Pivot1 | 11.80 | 60.45 | 3,902 | 3,941 |
| Pivot2 | 10.54 | 63.24 | 3,905 | 3,959 |
| Pivot3 | 12.91 | 66.68 | 3,907 | 3,971 |

The results were compared with the six models listed in Montero et al. (2020), VAE, $\beta$-VAE with $\beta = 6$, $\beta$-VAE with $\beta = 6$, 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. We observed that our methods showed significantly better reconstruction loss in the Recombination-to-Range setting.

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 significant reconstruction loss, as opposed to our model restoring both hue and shape successfully.

### 5.5 Ablation Study on the Pivot data

For proof of robustness of the model to the variation of the pivot data of our model, 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, which is both one of the criterion generative factors for splitting training and test dataset. 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. For the dSprites case, because the test dataset has the generative factors [shape=*square*, position-x$> 0.5$], the pivot1 with Position X value $0.02$ would have more difficulty in the reconstruction than the pivot3 with value $0.48$. Nevertheless, the result in the Table 3 demonstrates that the overall reconstruction loss is all similar for the pivot data regardless of the value of position X. This result implies the strong generalizability of our model.

## 6 Conclusion

In this paper, we proposed the novel generative framework MAGA capable of the combinatorial generalization task. It was confirmed that MAGA stably showed significantly better performance than the existing models qualitatively and quantitatively. Among the two main flows, combinatorial generalization and disentanglement, we did not explicitly concern about the disentanglement property of the model in this paper. Because we did not impose any constraint that the model should represent the disentangled representation, we can not say that our model aims at the disentanglement property right now. However, we firmly believe that the framework has the potential for solid disentanglement property because disentanglement is the concept fundamentally related to the transformation, not the embedding of individual data itself. We will further research the disentanglement property of the model.

## REFERENCES


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

Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in $\beta$-vae. *arXiv preprint arXiv:1804.03599*, 2018.

Ricky TQ Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. *Advances in neural information processing systems*, 31, 2018.

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

Ruili Feng, Jie Xiao, Kecheng Zheng, Deli Zhao, Jingren Zhou, Qibin Sun, and Zheng-Jun Zha. Principled knowledge extrapolation with gans. *arXiv preprint arXiv:2205.13444*, 2022.

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

Irina Higgins, David Amos, David Pfau, Sebastien Racaniere, Loic Matthey, Danilo Rezende, and Alexander Lerchner. Towards a definition of disentangled representations. *arXiv preprint arXiv:1812.02230*, 2018.

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

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

Murat Kocaoglu, Christopher Snyder, Alexandros G Dimakis, and Sriram Vishwanath. Causalgan: Learning causal implicit generative models with adversarial training. *arXiv preprint arXiv:1709.02023*, 2017.

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

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. A sober look at the unsupervised learning of disentangled representations and their evaluation. *arXiv preprint arXiv:2010.14766*, 2020a.

Francesco Locatello, Ben Poole, Gunnar Rätsch, Bernhard Schölkopf, Olivier Bachem, and Michael Tschannen. Weakly-supervised disentanglement without compromises. In *International Conference on Machine Learning*, pp. 6348–6359. PMLR, 2020b.

Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/, 2017.

Milton Llera Montero, Casimir JH Ludwig, Rui Ponte Costa, Gaurav Malhotra, and Jeffrey Bowers. The role of disentanglement in generalisation. In *International Conference on Learning Representations*, 2020.

Parallel Distributed Processing. Explorations in the microstructure of cognition. *Volume 1: Foundations*, 1986.

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

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

Lukas Schott, Julius Von Kügelgen, Frederik Träuble, Peter Gehler, Chris Russell, Matthias Bethge, Bernhard Schölkopf, Francesco Locatello, and Wieland Brendel. Visual representation learning does not generalize strongly within the same domain. *arXiv preprint arXiv:2107.08221*, 2021.

Zheyan Shen, Jiashuo Liu, Yue He, Xingxuan Zhang, Renzhe Xu, Han Yu, and Peng Cui. Towards out-of-distribution generalization: A survey. *arXiv preprint arXiv:2108.13624*, 2021.

Rui Shu, Yining Chen, Abhishek Kumar, Stefano Ermon, and Ben Poole. Weakly supervised disentanglement with guarantees. *arXiv preprint arXiv:1910.09772*, 2019.

Jayaraman Thiagarajan, Vivek Sivaraman Narayanaswamy, Deepta Rajan, Jia Liang, Akshay Chaudhari, and Andreas Spanias. Designing counterfactual generators using deep model inversion. *Advances in Neural Information Processing Systems*, 34:16873–16884, 2021.

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

Tao Yang, Xuanchi Ren, Yuwang Wang, Wenjun Zeng, and Nanning Zheng. Towards building a group-based unsupervised representation disentanglement framework. In *International Conference on Learning Representations*, 2021.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

## A ARCHITECTURE

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

Table 4: **The Encoder Architecture.**

| **Ratio** |
|---|
| $4 \times 4$ convolution with 32 channels |
| ReLU |
| $4 \times 4$ convolution with 32 channels |
| ReLU |
| $4 \times 4$ convolution with 32 channels |
| ReLU |
| $4 \times 4$ 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 CycleGAN (Zhu et al., 2017) as the decoder. The structure is as followings.

Table 5: **The Decoder Architecture.**

| **Layers** |
|---|
| $7 \times 7$ Convolution-InstanceNorm-ReLU with 64 channels |
| $3 \times 3$ Convolution-InstanceNorm-ReLU layer with 128 channels |
| $3 \times 3$ Convolution-InstanceNorm-ReLU layer with 256 channels |
| Residual Block with 256 channels |
| Residual Block with 256 channels |
| Residual Block with 256 channels |
| Residual Block with 256 channels |
| Residual Block with 256 channels |
| Residual Block with 256 channels |
| $3 \times 3$ fractional-strided-Convolution-InstanceNorm-ReLU with 128 channels |
| $3 \times 3$fractional-strided-Convolution-InstanceNorm-ReLU with 64 channels |
| $7 \times 7$ Convolutionn-InstanceNorm-ReLU with C channels |