Appendix

Table of Contents

Appendix A. Proofs

A.1. Proof of Lemma 3

Lemma 3 *Let $\mathcal{G}$ be the space of smooth invertible functions with smooth inverse (i.e., a diffeomorphism) (Locatello et al., 2020) that map $\mathcal{Z}$ to $\mathcal{X}$, and $h : \mathcal{Z} \to \mathcal{Z}$ is a smooth invertible function. Then, any function $g \in \mathcal{G}$ can be represented as $g = g^* \circ h$, where $g^* : \mathcal{Z} \to \mathcal{X}$ is the assumed true disentangled generative model in the function space $\mathcal{G}$. Formally, we have $g \sim_h g^*, \forall g \in \mathcal{G}$ and the model g is h-identifiable.*

Proof By definition, $g : \mathcal{Z} \to \mathcal{X}$ and $g^* \circ h : \mathcal{Z} \to \mathcal{X}$ are equal if their domain and codomain are the same and $g(\boldsymbol{z}) = g^*(h(\boldsymbol{z}))$ for any $\boldsymbol{z} \in \mathcal{Z}$. For $g \in \mathcal{G}$ and $g^* \circ h$, their domain and codomain are defined to be the same. Then, we could let h to be a permutation function, which is invertible, that satisfies $g(\boldsymbol{z}) = g^*(h(\boldsymbol{z})), \forall \boldsymbol{z} \in \mathcal{Z}$. First, we choose a $\boldsymbol{z} \in \mathcal{Z}$ and let $\boldsymbol{x} = g(\boldsymbol{z})$. Then, we can construct $h(\boldsymbol{z}) = \boldsymbol{z}^*$ such that $g^*(\boldsymbol{z}^*) = g^*(h(\boldsymbol{z})) = \boldsymbol{x}$. We can apply this procedure to every $\boldsymbol{z} \in \mathcal{Z}$. This is because we assume the mapping between $\mathcal{Z}$ and $\mathcal{X}$ are bijective, which means there is no conflict. As we assume h to be smooth, which is not implicitly satisfied by the permutation function, we show the smooth assumption holds by showing the limit of h exists everywhere on $\mathcal{Z}$:

$$\begin{aligned} &\lim_{\Delta \boldsymbol{z} \to \mathbf{0}} \frac{h(\boldsymbol{z} + \Delta \boldsymbol{z}) - h(\boldsymbol{z})}{\Delta \boldsymbol{z}} \\ &= \lim_{\Delta \boldsymbol{z} \to \mathbf{0}} \frac{{g^*}^{-1}(g(\boldsymbol{z} + \Delta \boldsymbol{z})) - {g^*}^{-1}(g(\boldsymbol{z}))}{\Delta \boldsymbol{z}} \\ &= {{g^*}^{-1}}'(g(\boldsymbol{z})) \times g'(\boldsymbol{z}) \end{aligned} \tag{5}$$

As we know that both g and ${g^*}^{-1}$ are smooth, their composition, ${g^*}^{-1} \circ g$, are also smooth. Thus, the limit above exist for any $\boldsymbol{z} \in \mathcal{Z}$. By definition, h is a smooth function. Thus, we can always find a smooth invertible permutation function h for any $g \in \mathcal{G}$ such that $g = g^* \circ h, \forall g \in \mathcal{G}$. As the valid output $\boldsymbol{x}$ for g and $g^* \circ h$ identical, we have $g \sim_h g^*$ by definition. As the equivalence relation holds between any $g \in \mathcal{G}$ and g^* that admits the same marginal distribution, g is h-identifiable. ■

A.2. Proof of Theorem 4

Theorem 4 *Let $i \in \{1, 2, ..., K_G\}$ be the index of data generating mechanisms. We assume each $\boldsymbol{z}_{G_i} \in \mathcal{Z}_{G_i}$, $\boldsymbol{z}_T \in \mathcal{Z}_T$, and $\boldsymbol{z} \in \mathcal{Z} = \mathcal{Z}_{G_0} \times \mathcal{Z}_{G_1} \times ... \times \mathcal{Z}_T$. We let $\hat{\boldsymbol{z}} = h(\boldsymbol{z})$, $\hat{\boldsymbol{z}} \in \hat{\mathcal{Z}}$, and $g^* : \mathcal{Z} \to \mathcal{X}$ be the true disentangled model. Then, if there exists a smooth invertible function $h : \mathcal{Z} \to \mathcal{Z}$ such that $g = g^* \circ h$ maps $\mathcal{Z}$ to $\mathcal{X}$, then h maps each $\mathcal{Z}_{G_i}$ to $\hat{\mathcal{Z}}_{G_i}$, disjoint from $\hat{\mathcal{Z}}_{G_j}, \forall j \neq i$, as well as $\hat{\mathcal{Z}}_T$, and maps $\mathcal{Z}_T$ to $\hat{\mathcal{Z}}_T$, which is disjoint from $\hat{\mathcal{Z}}_{G_i}, \forall i \in \{1, ..., K_G\}$.*

Proof By the definition of degenerate mixture prior, we have: if $\boldsymbol{z}_{G_i} \neq \mathbf{0}$, $\boldsymbol{z}_{G_j} = \mathbf{0}$ for all $j \neq i$. We name this condition as the structure constraint. As we know that $h : \mathcal{Z} \to \mathcal{Z}$ and $g^* : \mathcal{Z} \to \mathcal{X}$, we have $\mathcal{Z} = \hat{\mathcal{Z}}$ because the valid inputs for g^* are fixed. The structure constraint is enforced on both $\mathcal{Z}$ and $\hat{\mathcal{Z}}$. If h entangles $\hat{\boldsymbol{z}}_{G_i}$ and $\hat{\boldsymbol{z}}_{G_j}$ for any $i \neq j$, which means the change of one ground-truth latent variable $\boldsymbol{z}_{G_k}$ affects two learned latent variables $\hat{\boldsymbol{z}}_{G_i}$ and $\hat{\boldsymbol{z}}_{G_j}$. Then, there exist a $\boldsymbol{z}$ such that $h(\boldsymbol{z})_{G_i} = \hat{\boldsymbol{z}}_{G_i} \neq \mathbf{0}$ and $h(\boldsymbol{z})_{G_j} = \hat{\boldsymbol{z}}_{G_i} \neq \mathbf{0}$ hold simultaneously, violating the structural constraint.

To see the violation, we can let $[\boldsymbol{z}_{G_i}, \boldsymbol{z}_{G_j}] = [[1.0, 0.0]^\top, [0.0, 0.0]^\top]$ and construct $[\hat{\boldsymbol{z}}_{G_i}, \hat{\boldsymbol{z}}_{G_j}] = h([\boldsymbol{z}_{G_i}, \boldsymbol{z}_{G_j}]) = [[0.7, 0.0]^\top, [0.5, 0.0]^\top]$. We can also let $[\boldsymbol{z}_{G_i}, \boldsymbol{z}_{G_j}] = [[1.0, 1.0]^\top, [0.0, 0.0]^\top]$ and construct $[\hat{\boldsymbol{z}}_{G_i}, \hat{\boldsymbol{z}}_{G_j}] = h([\boldsymbol{z}_{G_i}, \boldsymbol{z}_{G_j}]) = [[0.0, 1.0]^\top, [1.0, 0.0]^\top]$. The existence of such $\hat{\boldsymbol{z}}_{G_i}$ and $\hat{\boldsymbol{z}}_{G_j}$ violates the structure constraint. Thus, h need to map each $\mathcal{Z}_{G_i}$ to $\hat{\mathcal{Z}}_{G_i}$, disjoint from $\hat{\mathcal{Z}}_{G_j}, \forall i \neq j$, to make the structure constraint hold.

Additionally, if h entangles $\hat{\boldsymbol{z}}_T$ and $\hat{\boldsymbol{z}}_{G_i}$ for any valid i, there exist two cases: 1) h maps $\mathcal{Z}_T$ to $\hat{\mathcal{Z}}_T$ and $\hat{\mathcal{Z}}_{G_i}$ or 2) h maps $\mathcal{Z}_{G_i}$ to $\hat{\mathcal{Z}}_T$ and $\hat{\mathcal{Z}}_{G_i}$. For the first case, there exist a $j \neq i$ such that $\boldsymbol{z}_{G_i} = \mathbf{0}$ and $\boldsymbol{z}_{G_j} \neq \mathbf{0}$ but $h(\boldsymbol{z})_{G_i} = \hat{\boldsymbol{z}}_{G_i} \neq \mathbf{0}$ and $h(\boldsymbol{z})_{G_j} = \hat{\boldsymbol{z}}_{G_j} \neq \mathbf{0}$. This is because $\boldsymbol{z}_T$ is independent from all the $\boldsymbol{z}_{G_i}$, $\boldsymbol{z}_{G_j}$ and can push $\hat{\boldsymbol{z}}_{G_i}$ to arbitrary value when other variables are fixed. Thus, the structure constraint is violated. For the second case, we can apply the proof of the first case in the opposite direction using the assumption that h is smooth and invertible. we can find a $\hat{\boldsymbol{z}}_T$ such that $h^{-1}(\hat{\boldsymbol{z}})_{G_i} = \boldsymbol{z}_{G_i} \neq 0$. In the meanwhile, we can again find a $\hat{\boldsymbol{z}}_{G_j}$ which makes $h^{-1}(\hat{\boldsymbol{z}})_{G_j} = \boldsymbol{z}_{G_j} \neq 0$. This contradicts the structure constraint. Thus, to keep the structure constraint hold, h needs to map $\mathcal{Z}_T$ to $\hat{\mathcal{Z}}_T$, which is disjoint from all the $\hat{\mathcal{Z}}_{G_i}$ and avoids mapping $\hat{\mathcal{Z}}_{G_i}$ for any valid i to $\hat{\mathcal{Z}}_T$.

Furthermore, if h entangles multiple mechanisms, we can find two variables among the entangled latent variables to show that the structure constraint is violated. ■

Appendix B. Implementation Details

B.1. Network Architecture

We report the network architectures for 1x28x28, and 3x64x64 images in Table 2, and 3, respectively. We use the same generator/decoder and encoder (with modified output layer) for VAEs. For the colored dataset Sprites, we use a separate encoder, which takes the gray-scale image is input and predicts $\hat{\boldsymbol{z}}_C$ and $\hat{\boldsymbol{z}}_G$. Such configuration would avoid identifying mechanisms by color.

Table 2: Generator, Discriminator, and Encoder Architectures for $1 \times 28 \times 28$ Inputs

Generator	Discriminator	Encoder
Input: $\mathbb{R}^{\dim(\boldsymbol{z})}$	Input: $\mathbb{R}^{1\times28\times28}$	Input: $\mathbb{R}^{1\times28\times28}$
FC 1024 BN ReLU	4×4 conv, 64 LReLU, stride 2	4×4 conv, 64 LReLU, stride 2
FC 128×7×7 BN ReLU	4×4 conv, 128 LReLU, stride 2	4×4 conv, 128 LReLU, stride 2
4×4 upconv, stride 2, 64 BN ReLU	FC 1024 LReLU	FC 1024 LReLU
4×4 upconv, stride 2, 1 Sigmoid	FC 1	FC dim($\boldsymbol{z}$)

Table 3: Generator, Discriminator, and Encoder Architectures for $3 \times 64 \times 64$ Inputs

Generator	Discriminator	Encoder
Input: $\mathbb{R}^{\dim(\boldsymbol{z})}$	Input: $\mathbb{R}^{3\times64\times64}$	Input: $\mathbb{R}^{3\times64\times64}$
FC 1024 BN ReLU	4×4 conv, 64 LReLU, stride 2	4×4 conv, 64 LReLU, stride 2
FC 128×8×8 BN ReLU	4×4 conv, 64 LReLU, stride 2	4×4 conv, 64 LReLU, stride 2
4×4 upconv, stride 2, 64 BN ReLU	4×4 conv, 128 LReLU, stride 2	4×4 conv, 128 LReLU, stride 2
4×4 upconv, stride 2, 64 BN ReLU	FC 1024 LReLU	FC 1024 LReLU
4×4 upconv, stride 2, 3 Sigmoid	FC 1	FC dim($\boldsymbol{z}$)

B.2. Hyper-parameters

We set $[\lambda_G, \lambda_T, \lambda_C\lambda_R]$ to $[10.0, 10.0, 30.0, 0.05]$ on all the datasets and change λ_T to 50 for Sprites dataset. We train our ICM model for 200 epochs. For each β-VAE and Ada-GVAE model, we train the model for 200 epochs with $\beta = [1, 2, 4, 8, 16]$. We train each generative model three times and report the best result. We use Adam optimizer with learning rate = 0.001 and batch size = 64 across the experiments. We set $(\beta_1, \beta_2) = (0.5, 0.9)$ for ICM and set $(\beta_1, \beta_2) = (0.9, 0.999)$ for VAEs as Gulrajani et al. (2017) suggest. We train the VAEs with dimension $[10, 20, 30, 50]$. We find the VAEs with 20 dimensions have the best robustness when β is optimal. For the ICM model, we set the dimension of each $\boldsymbol{z}_{G_i}$ to 1, 2, and 1 on MNIST, FashionMNIST, and Sprites, respectively. The dimension of $\boldsymbol{z}_T$ is 4, 5, and 3 on MNIST and FashionMNIST, and Sprites, respectively.

B.3. Disentangled Representations Datasets

The disentangled representations datasets, such as 3DShapes, assume the latent explanatory factors are independent of each other. Such an assumption is not compatible with the class of system that we study. We tried to create a dataset that correlates one type of variation (e.g. floor hue) with one shape. However, such a configuration will make the data too small to use.

B.4. Choosing the Number of Mechanisms

Our method does not require users to know the precise number of mechanisms in advance because our ICM model can tolerate the discrepancy between the learned mechanisms and the true mechanisms if the number of learned mechanisms is greater or equal to the number of true mechanisms. For example, we use 15 mechanisms in the MNIST experiment. The users can easily choose the number of mechanisms by applying the following procedure: 1) Pick a random number and train the ICM model. 2) Do a latent space traversal and observe the type of variations. 3) If the data samples from the same data generating mechanism share the same type of variations, the ICM model is ready to use. Otherwise, increase the number of mechanisms and repeat steps 1)&2).

B.5. Training Setting for Sprites

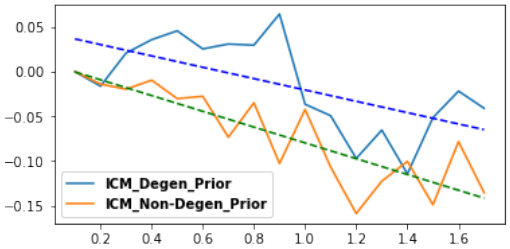
If the environment shift that affects both the ICM model and the downstream GBT model is caused by color, we do not find disentanglement improves the accuracy of the GBT model. Dittadi et al. (2021) report similar results. The possible reason is that the color shift in our experiment is not continuous, differing from width and brightness shifts. If the ICM model has never seen any blue color, it is difficult to "imagine" a blue object based on the red or green. Therefore, we let the environment shift affect the GBT model only. Such a setting is also helpful for showing the benefit of disentanglement and is called "out-of-distribution 1 setting" in a recent work (Dittadi et al., 2021).

Appendix C. More Experimental Results

C.1. Ablation Study

We conduct qualitative ablation study using a non-degenerate mixture prior $p(z_G) = \sum_{i=1}^{K_G} \frac{1}{K_G} \cdot \mathcal{N}(z_G \mid \mu_i, \sigma_i)$ in the ICM model. We find that if we want to reduce the overlap between mixture components by letting μ_i be far away from each other and letting each σ_i be small, the samples from

(*a*) Latent Space Traversal of ICM Model with Non-degenerate Mixture Prior.



(*b*) GBT Accuracy Decrease (y-axis) under Different Environment Shift Distance (x-axis).

Figure 8: (a) shows that circle size factor and circle number factor are entangled. (b) shows that ICM model with degenerate mixture prior slows down the accuracy decrease under environment shift.

clusters with large $|\mu_i|$ makes the GAN training unstable, and the small σ_i reduces the model capacity for each mixture component. Therefore, we choose a smaller K_G=3, let $\boldsymbol{\mu} = [-1.0, 0.0, 1.0]$ and $\boldsymbol{\sigma} = [0.5, 0.5, 0.5]$, and report the visualization result in Figure 8(*a*). We can see that (1) the data variability is limited, (2) the two types of variations, circle size and circle number, are entangled, and (3) interventions could change the label of the digit. (2) and (3) support our discussion in Section 2.2.

Quantitatively, we consider a binary classification task using digits 4 and 9 from the MNIST dataset. The shifted conditions are the styles of digits 4 and 9 (e.g., the circle size of digit 9). For the train set, we collect $\{\boldsymbol{x} \mid \boldsymbol{x} \in \mathcal{X}, \hat{\boldsymbol{z}}_4 = E(\boldsymbol{x})_4 \in [C_{lower}, C_{upper}) \vee \hat{\boldsymbol{z}}_9 = E(\boldsymbol{x})_9 \in [C_{lower}, C_{upper})\}$, where $[C_{lower}, C_{upper})$ is $[-1.0, 1.0)$. We collect a union of two subsets for each test set by letting $[C_{lower}, C_{upper})$ be $[-1.1, -1.0)$, $[-1.2, -1.1)$, ..., $[-3.0, -2.9)$ and $[1.0, 1.1)$, $[1.1, 1.2)$, ..., $[2.9, 3.0)$, respectively. We use the GBT model as the classifier. Figure 8(*b*) shows that as the environment shifts, the accuracy decreases faster if we use the encoder from the ICM model with a non-degenerate mixture prior. Additionally, using degenerate mixture prior yields 90.72% average accuracy, contrasting the 78.44% average accuracy from using non-degenerate mixture prior. This result complements our discussion in Section 2.2.

C.2. Qualitative Evaluation

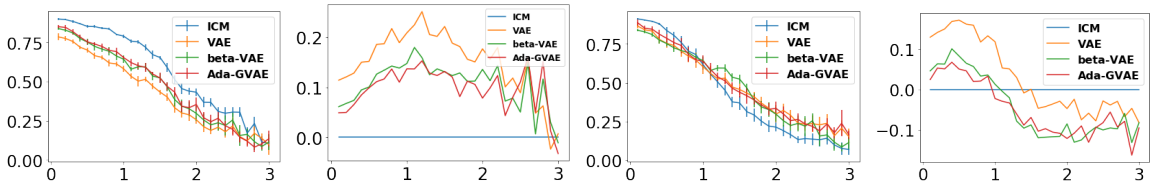
MNIST We visualize the data transforming mechanism subspace traversal in Figure 11.

FashionMNIST We visualize the traversal for each data generating mechanism subspace and the data transforming mechanism subspace in Figure 12.

Sprites We visualize the traversal for each data generating mechanism subspace and the data transforming mechanism subspace in Figure 13.

C.3. Quantitative Evaluation

We further investigate our model under MNIST (R) environment shift. Figure 9 shows the accuracy changes as the shift strength increases. Our method yields higher accuracy and slower accuracy decreases in both experiments when the shift strength is moderate. However, after a passing threshold,



(*a*) Accuracy under Width&Rotation Shift (*b*) Accuracy Differences between ICM and Others (*c*) Accuracy under Rotation Shift (*d*) Accuracy Differences between ICM and Others

Figure 9: Accuracy and accuracy differences of downstream classifiers under environment shift on the MNIST dataset. If ICM outperforms another method, the accuracy difference is positive. Note that our proposed ICM outperforms all the baselines in most settings. Exceptions include when the environment shifts too far for any model to tolerate.

our method begins to lose its advantage. Our analysis suggests two reasons: 1) The test data shifts too far from the training data, and the base generative model can not generalize to the test sets. After the test set shifts too far away, none of the methods perform well. Thus, it is hard to conclude that a model with $\sim 40\%$ accuracy is better than a model with $\sim 35\%$ accuracy. 2) The test set contains too few samples, just around tens or hundreds, making the evaluation inaccurate. As we can see from Figure 9, the larger the shift, the bigger the accuracy variations.

To eliminate this interference, we measure how much environment shift is needed to decrease the accuracy by a relative percentage. Such evaluation will put more weight on the test set that has more samples and yields reasonable accuracy. Tables 4, 5, and 6 show our method can *tolerate more environment shift* before the test sets shift too far away and decrease the accuracy by 10%, 20%, and 40%, relatively.

Table 4: Shift Distance Needed for 10% Relative Accuracy Drop under Environment Shift

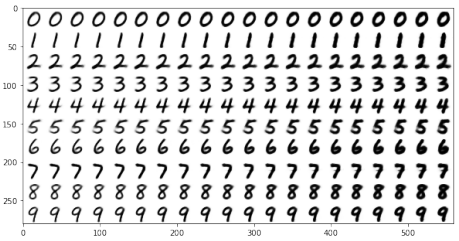
MODEL	MNIST (T)	MNIST (W&R)	MNIST (R)	FASHIONMNIST (D&W)
ICM	**1.4**	**0.8**	**0.6**	**0.4**
VAE	0.6	0.4	0.4	0.2
β-VAE	1.1	0.5	0.4	0.3
ADA	1.1	0.5	0.4	0.3

Table 5: Shift Distance Needed for 20% Relative Accuracy Drop under Environment Shift

MODEL	MNIST (T)	MNIST (W&R)	MNIST (R)	FASHIONMNIST (D&W)
ICM	**2.0**	**1.2**	**0.8**	**0.7**
VAE	1.2	0.7	0.7	0.4
β-VAE	1.7	0.8	0.7	0.5
ADA-GVAE	1.6	0.9	0.7	0.5

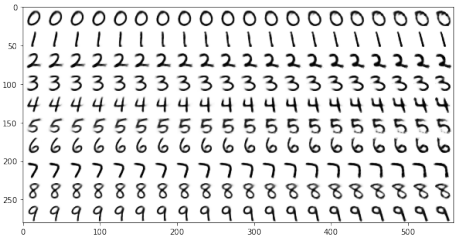
Table 6: Shift Distance Needed for 40% Relative Accuracy Drop under Environment Shift

MODEL	MNIST (T)	MNIST (W&R)	MNIST (R)	FASHIONMNIST (D&W)
ICM	**2.6**	**1.6**	1.1	**1.2**
VAE	1.9	1.3	1.1	0.8
β-VAE	2.4	1.5	**1.3**	1.1
ADA-GVAE	2.2	1.5	1.2	1.0

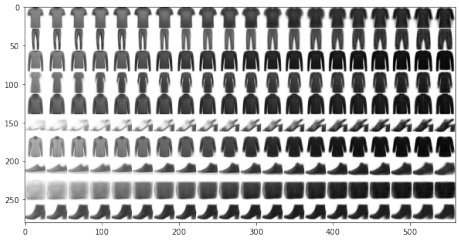


(*a*) MNIST: Stroke Thickness (T)

(*b*) MNIST: Width&Rotation (W&R)

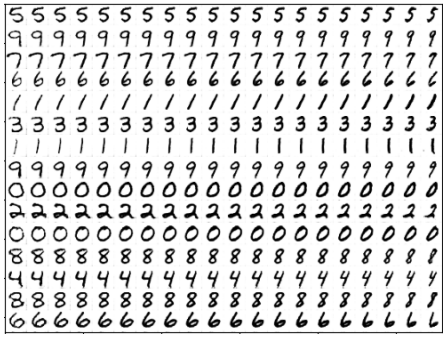


(*c*) MNIST: Rotation (R)

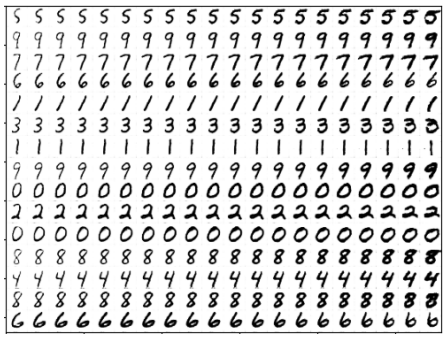


(*d*) FashionMNIST: Darkness&Width (D&W)

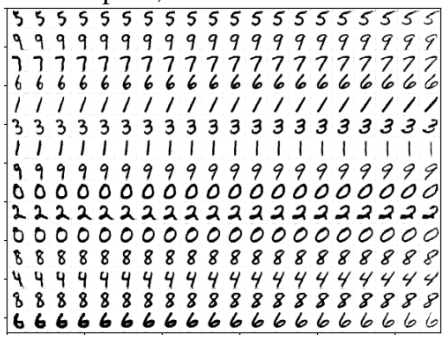
Figure 10: Conditions (Latent Variables) Used in the Environment Shift Experiment.



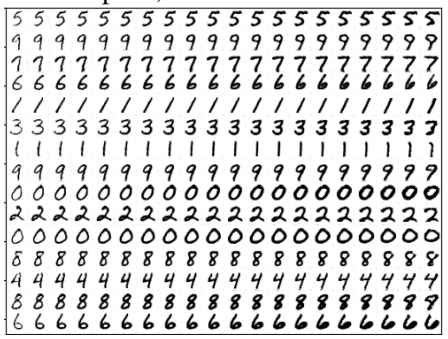
(*a*) Data Transforming Mechanism Subspace, Dimension #0



(*b*) Data Transforming Mechanism Subspace, Dimension #1

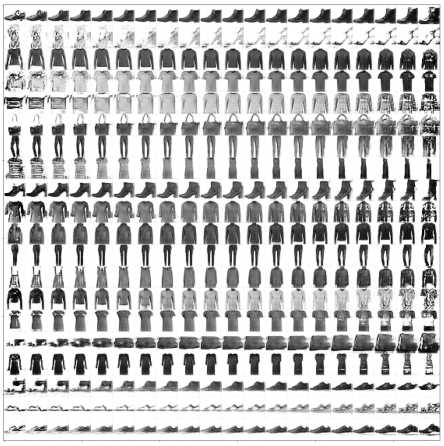


(*c*) Data Transforming Mechanism Subspace, Dimension #2

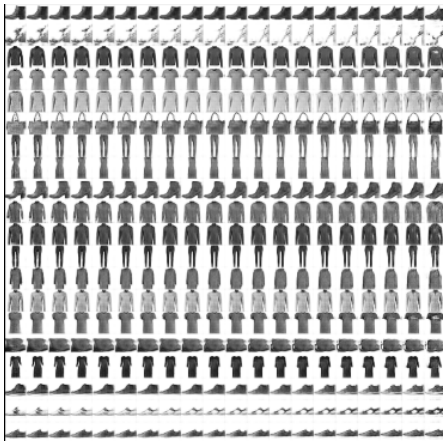


(*d*) Data Transforming Mechanism Subspace, Dimension #3

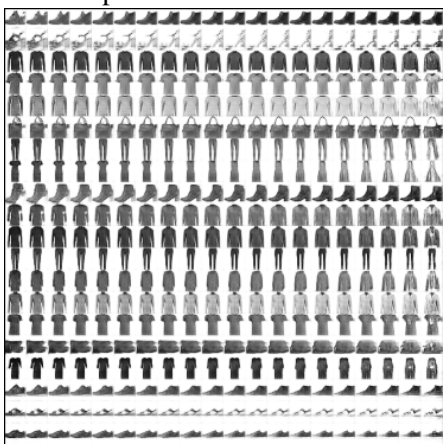
Figure 11: Data Transforming Mechanism Subspace Traversal of ICM model on MNIST



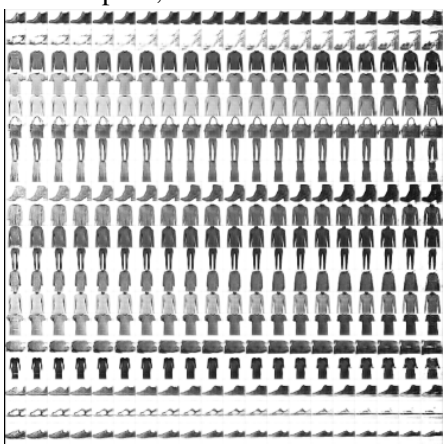
(*a*) Data Generating Mechanisms Subspaces



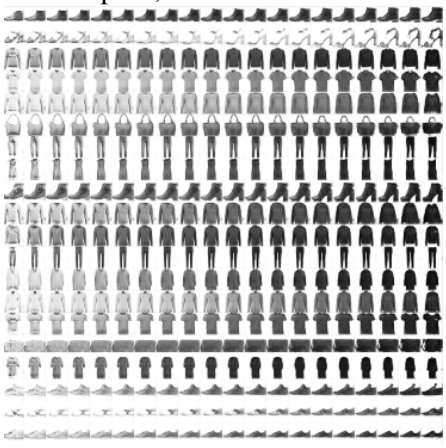
(*b*) Data Transforming Mechanism Subspace, Dimension #0



(*c*) Data Transforming Mechanism Subspace, Dimension #1



(*d*) Data Transforming Mechanism Subspace, Dimension #2

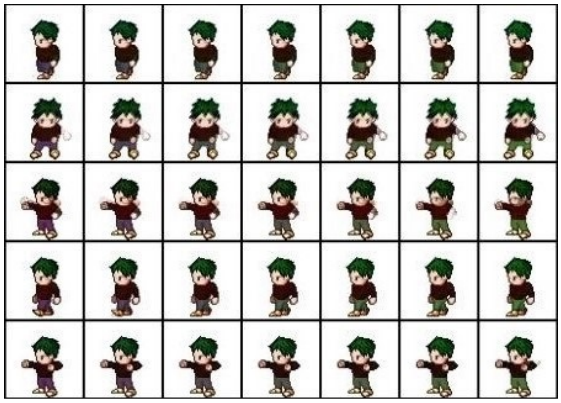


(*e*) Data Transforming Mechanism Subspace, Dimension #3

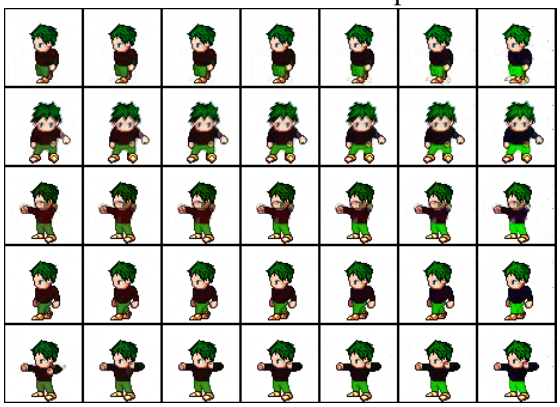
(*f*) Data Transforming Mechanism Subspace, Dimension #4

Figure 12: Latent Space Traversal of ICM model on FashionMNIST

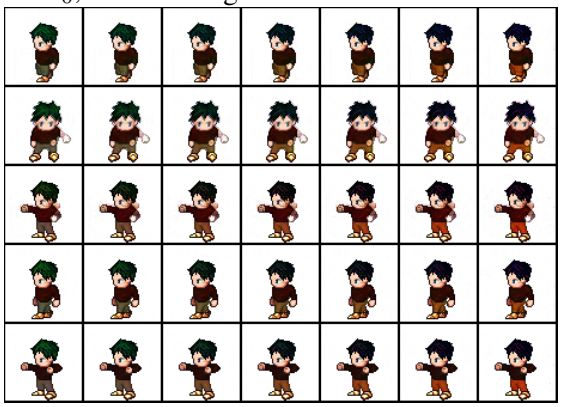
(*a*) Data Transforming Mechanism Subspace, Distinct Action for Each Subspace/Row



(*b*) Data Transforming Mechanism Subspace, z_0, Color Change



(*c*) Data Transforming Mechanism Subspace, z_1, Color Change



(*d*) Data Transforming Mechanism Subspace, z_2, Color Change

Figure 13: Latent Space Traversal of ICM model on Sprites