A Reproducibility

In this section, we provide the information required to reproduce our results reported in the main text. And we commit to making the code implementation and evaluating checkpoints public. Our experiments are run on a machine with AMD Ryzen Threadripper 3970X 32-Core Processor and GeForce RTX 3090 GPU.

Contrastive Learning implementation For the implementation details of contrastive learning, please refer to Appendix A.1. The model architecture, training setups, and dataset preprocessing are all explained in detail. Our implementations are based some public and official implementations of MoCo/MoCov2 [2], BYOL/ SimSiam [3] and Barlow Twins [4].

VAE methods implementation For evaluation on synthetic datasets, i.e., dSprites, Cars3D, Small-NORB, and Shapes3D, the disentanglement score is from the original logs of DisLib Locatello et al. [5]. In the released logs, each method has different training configurations, and our reported result is from the configuration with the highest average performance overall provided random seeds. For evaluation on CelebA dataset, we follow an open-sourced implementation in Pytorch [6] and align the encoder architecture of all methods to be the same as described in Appendix A.1. For the results on Shapes3D, because DisLib does not release the pretrained checkpoints, we use the same open-sourced implementation to reproduce with the configuration indicated by DisLib. Parameters are kept as the default well-tuned version in the provided implementation. When the latent dimension is 1000, training of BetaTC VAE will collapse with the default hyperparameters, we have to decrease the β to 3.0 to work it around.

GAN methods implementation Limited by the text length, we do not include the performance of GAN methods in the main text, but we will report some in the following appendix content. It is hard to include GAN methods' performance in the benchmark as the training is not always stable and the discriminator weights are usually not provided in many public codebases. When evaluating on synthetic datasets, the FactorVAE scores of InforGAN, IB-GAN, and InfoGAN-CR are provided in the paper of Lin et al.. But the evaluation of other metrics in Lin et al. uses a not aligned settings with Locatello et al., so we check its officially release [7] to reevaluate the provided implementation and model weights under the unified evaluation setup. We perform the same evaluation process for results on the CelebA dataset.

Energy-based Model (EBM) We refer to the implementation of ICE-BeeM ([22]) for this method. We use the officially released codebase for it [8]. The encoder implementation has been aligned with our default already. The only modification we make is to use the unconditional version instead of its default conditional version in loss computation to satisfy the fully unsupervised settings. Please refer to the **runners/real_data_runner.py** file of the codebase for details.

Evaluation Protocol For MED, we first compute MI following the implementation of MIG by DisLib ([34]). Then we calculate the entropy disentanglement score in the same way as the DCI Disentanglement score in DisLib. For other disentanglement metrics evaluation, we use the implementation of DisLib. The settings of some important parameters rather than our proposed MED are provided in Appendix A.2.

A.1 Implementation of contrastive learning model

Architecture To make a fair comparison with previous methods, we follow the encoder architecture in Factor VAE ([23]). The pipeline details are shown in Table 2. After each convolutional layer in the figure, there is a ReLU activation layer and a group normalization (group number = 4) layer for BYOL. So, the encoder is a stack of (Conv-ReLU-GN) blocks. For other contrastive learning methods, we keep the default batch normalization to replace GN. By default, the final output channel

[2] https://github.com/facebookresearch/moco
[3] https://github.com/lucidrains/byol-pytorch
[4] https://github.com/facebookresearch/barlowtwins
[5] https://github.com/google-research/disentanglement_lib
[6] https://github.com/AntixK/PyTorch-VAE
[7] https://github.com/fjxmlzn/InfoGAN-CR
[8] https://github.com/ilkhem/icebeem

number is 1000, i.e, $D = 1000$. For other details of contrastive learning methods, we follow the convention in their official implementations.

Besides the representation network (encoder), BYOL also has a projector network and a predictor network. Both of them consist of a pipeline "Linear $\longrightarrow$ BN $\longrightarrow$ ReLU $\longrightarrow$ Linear". The projection dimension is 256, and the hidden dimension of the projector is 4096. The predictor keeps a 256-dimensional feature vector in its pipeline.

Table 2: The encoder architecture for our implemented contrastive learning methods on synthetic datasets. Besides, there is a ReLU activation layer and a possible normalization layer following each convolutional layer to create a stack of (Conv-ReLU-Norm) blocks.

Encoder
input: 64×64 images
pipeline:
4×4 conv, stride 2, 32-channel
4×4 conv, stride 2, 32-channel
4×4 conv, stride 2, 64-channel
4×4 conv, stride 2, 64-channel
4×4 conv, stride 2, 128-channel
1×1 conv, stride 1, D-channel

Training settings We make minor modifications to the training setting of default BYOL to apply to contrastive learning methods without negative samples. For training on all datasets, the images are resized to 64x64. For data preprocessing, we copy 1-channel images of dSprites and SmallNORB to 3-channel. During the training stage, we use such a pipeline of augmentation (in *PyTorch*-style):

1. *RandomApply(transforms.ColorJitter(0.8, 0.8, 0.8, 0.2), p=0.3)*
2. *RandomHorizontalFlip()*
3. *RandomApply(transforms.GaussianBlur((3,3), (1.0, 2.0)), p=0.2)*
4. *RandomResizeCrop(size=(64, 64), scale=(0.6,1.0))*
5. normalization.

For the normalization, the pixel value of images from dSprites and SmallNORB is uniformly normalized from [0,255] to [0,1.0]. For Cars3D, Shapes3D, and CelebA, we adopt the commonly used Imagenet-statistic normalization for preprocessing the image values.

During training, we use Adam optimizer by default, whose learning rate is $3e-4$ without weight decay. The batch size is set to be 512 without exceptional notation. For evaluation on dSprites, Shapes3D, and CelebA, we select the weights after training for 15 epochs for evaluation. We select the weights after training for 140 epochs for evaluation on Cars3D and the weights of the 200th epoch on SmallNORB considering the small scale of these two datasets.

To decrease the influence of randomness, we train each model configuration multiple times with different random seeds (seed=0, 1, 2). We report the average and standard deviation. To be precise, as our implementation is based on Pytorch, we initialize the libraries of *numpy*, *torch*, *torch.cuda*, and *random* with the same random seeds.

A.2 Evaluation Metrics

In the main text, we compare the evaluation metrics provided in the DisLib protocol with our proposed MED metric. Here we provide more details about them. Moreover, we would conduct evaluations under all of them in the next section.

BetaVAE Metrics Introduced in Higgins et al. (16), BetaVAE score assumes each dimension corresponds to one category in a linear classifier. Representations are obtained after the generated samples with only one factor fixed. Calculating the summation of the divergence between different

Table 3: The factors on all the datasets we investigate the disentanglement on.

	dSprites	Shapes3D	Cars3D	SmallNORB	CelebA
Factors (# of values)	Shape (3) Scale (6) Orientation (40) Position X (32) Position Y (32)	Floor hue (10) Wall hue (10) Object hue (10) Scale (8) Orientation (15) Shape (4)	Elevation (4) Azimuth (24) Object id (183)	category (10) Elevation (9) Azimuth (18) Lighting (6)	40 attributes (2 for each)

representations and putting this result into a linear classifier, we train a model that possibly outputs the corresponding k. The accuracy of this linear model is the value of BetaVAE metric.

FactorVAE Metrics Kim and Mnih (23) argues the BetaVAE score has the tendency to fail into a spurious disentanglement and proposes a new metric based on a majority vote classifier. Representations are obtained after the generated samples with only factor k fixed. Normalizing each dimension in representations in terms of standard deviation. Index of dimension with lowest variances of normalized representation and the factor index k is the input/output of the linear classifier. The accuracy of the classification is the FactorVAE score.

Mutual Information Gap Chen et al. (7) assumes the disentanglement model has the property that most information of one specific factor is contained in one dimension or a group of certain dimensions. The mutual information gap is the summation of the difference between the highest and second-highest normalized mutual information between a fixed factor and dimensions in representation. The formula can be illustrated as below:

$$\frac{1}{K}\sum_{k=1}^{K}\frac{1}{H_{z_k}}(I(v_{j_k}, z_k) - \max_{j\neq j_k} I(v_j, z_k)) \tag{5}$$

Where K is the overall number of ground truth factors. v is the latent representation and z_k is the factors of latent variables and $j_k = \arg\max_j I(v_j, z_k)$.

DCI disentanglement As Eastwood and Williams (11) suggests, the disentanglement is measured by the entropy of relative importance for each dimension in predicting factors. First, we have to know the importance of each dimension of the representation for predicting each factor. The importance is determined by a regressing model such as Lasso or Random Forest in the original DCI implementation (11) or Gradient Boosting Tree in DisLib implementation (34). We note the importance matrix R where R_{ij} is the importance of the i-th dimension in prediction the j-th factor. Then disentanglement score for the i-th dimension is defined as $D_i = (1 - H_K(P_i))$ where $H_K(P_i) = -\sum_{k=0}^{K-1} P_{ik} log_K P_{ik}$ denotes the entropy and $P_{ij} = R_{ij}/\sum_{k=0}^{K-1} R_{ik}$ denotes the normalized importance of i-th dimension in prediction the j-th factor. Finally the overall disentanglement score is calculated as $D = \sum_i \rho_i D_i$ where $\rho_i = \sum_j R_{ij}/\sum_{ij} R_{ij}$ is the weighting of the each dimension's informativeness in representing factors.

SAP (27) proposes the Separated Attribute Predictability (SAP) score. A score metrics is computed with classification score of predicting j^{th} factors on i^{th} dimension as the ij^{th} entry. SAP is the mean of the difference between the highest and second-highest scores for each column.

We follow the implementation provided by DisLib (34) for the evaluation protocol. Despite exceptions, the evaluation batch size is 64, the *prune_dims.threshold* is 0.06. If a classifier is required to be trained during evaluation, *num_train* is 10000, and *num_eval* is 5000. For Mutual information computation, the discretizer function is the histogram discretizer, and the number of bins in the discretization is 20. For the evaluation of MIG and SAP on dSprites, SmallNORB, Cars3D, and Shapes3D, BYOL representation vectors are reduced to 10 dimensions by PCA to be aligned with other methods. For the evaluation of MIG and SAP on CelebA, to have a fair comparison, the representation vectors of all methods are reduced to 40 dimensions. For the implementation of our proposed MED, the basic logic is the same as DCI Disentanglement, but we replace the classifier output with the mutual information based scores.

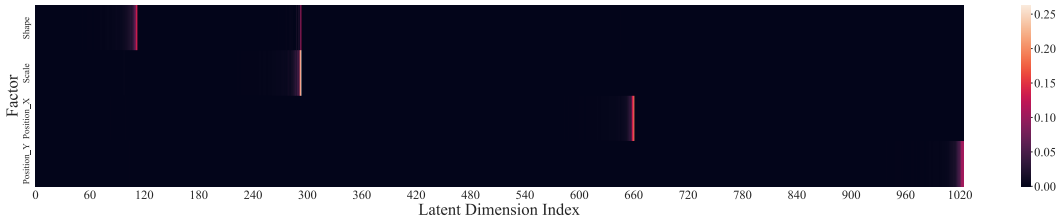
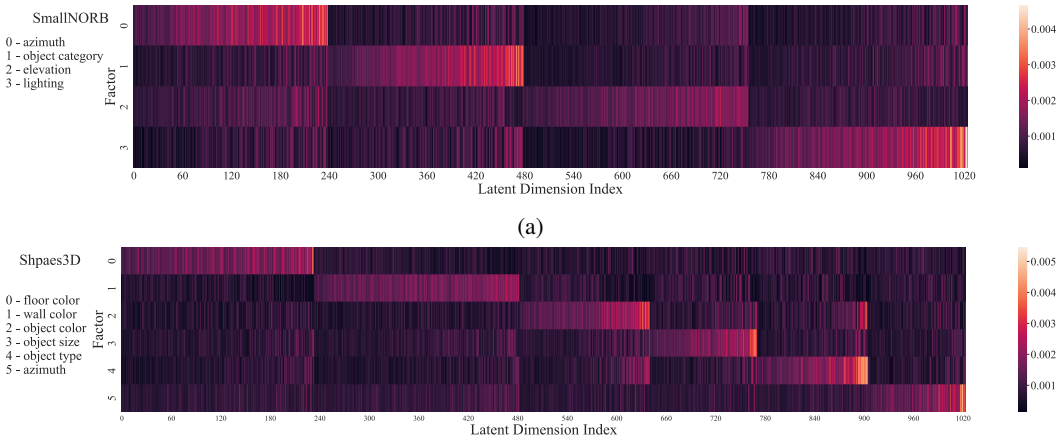


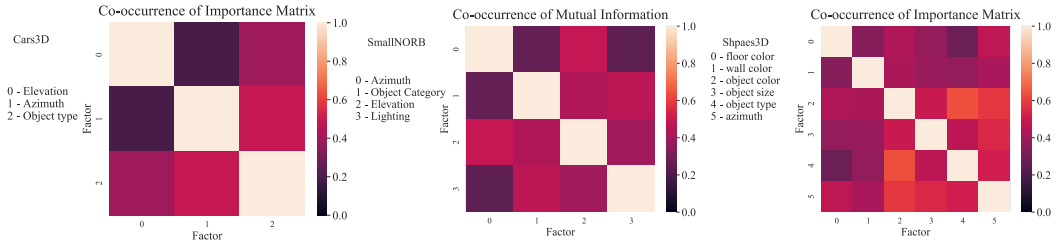
Figure 6: The importance distribution for the representation learned from BYOL on dSprites. Here, we follow the practice of DisLib to use a Gradient Boosting Tree (GBT) regressor to determine the importance matrix of each latent dimension in predicting each factor. Compared with the Mutual Information distribution shown in Figure 1a, the importance distribution is significantly more sparse. The sparsity is encouraged when constructing the GBT regressor. This makes it hard to study the true representation pattern.



(a)

(b)

Figure 7: The mutual information distribution on SmallNORB(a) and Shapes3D(b).



(a) (b) (c)

Figure 8: The co-occurrence of factors in the mutual information relationship among BYOL representations on Cars3D(a), SmallNORB(b) and Shapes3D(c).

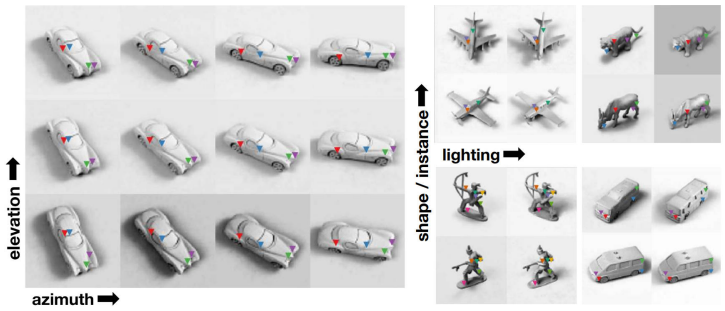


Figure 9: Some samples from SmallNORB dataset. The variance is controlled by the factor indicated on axis. The image is from Jakab et al. (20).

B More Qualitative Study

Limited by the main text length limitation, we provide more qualitative studies about the disentanglement property shown by the contrastive learning here. We still use BYOL as an example of the negative-free contrastive learning methods.

B.1 Importance Distribution by DCI

In the main text, we concisely talked about the potential variables introduced by the learnable model under some metrics. Here we show an example for the widely used DCI Disentanglement metric. We follow DisLib to use Gradient Boosting Tree to produce the Importance estimation between each factor and each latent dimension. All parameters are set the same as its default protocol. The visualization is shown in Figure 6. Compared with the mutual information distribution shown in Figure 1a, the importance distribution is obviously much more sparse. Sparsity is encouraged during constructing the GBT regressor. However, the observation can lead to the misunderstanding that the correlation between factors and latent dimensions is sparse which is not true. By using the pure measurement without involving additional adaptive models, such problem will not be raised in the proposed MED metric.

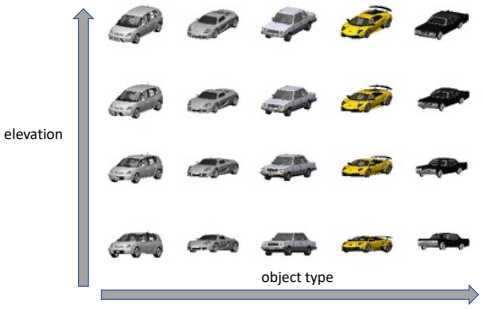


Figure 10: Samples from Cars3D. Object type and elevation are controlled. It show that the two factors are not independent.

B.2 Mutual Information Heatmaps

We compute MI between each latent dimension and each generative factor and visualize them by heatmaps, which offer us an intuitive picture of the learned representation space. For completeness, we show the MI heatmaps of SmallNORB and Shapes3D in Figure 7a and Figure 7b respectively. We can see that the disentangled pattern described in the main text still emerges. There is a group of columns brighter than others in each row, and these groups do not overlap for most rows. However, we find that some latent dimensions may emphasize more than one factor. We provide a more detailed analysis from the perspective of factor co-occurrence on this phenomenon in section B.3 below.

B.3 Co-occurrence of Factors

To understand to what extent one dimension of the learned representation would respond to more than one factor, we make the co-occurrence of mutual information to factors on more datasets here. The visualizations are shown in Figure 8b, Figure 8a and Figure 8c on SmallNORB, Cars3D, and Shapes3D respectively.

SmallNORB Though most non-diagonal entries have very low co-occurrence of mutual information, two pairs of factors show slightly higher co-occurrence. They are "azimuth-elevation" and "instance category-lighting". After investigating the dataset, we find the two pairs of factors are not fully independent. Figure 9 show some samples with corresponding factors manipulated. We could see that the elevation and azimuth are not fully independent. And the correlation between the instance category and the lighting factor is even more obvious because the lighting condition is sensibly related to the shadow around the object, whose distribution and shape is highly determined by the instance category.

Cars3D Only one pair of factors show some co-occurrence, i.e. "elevation-object type". We randomly selected samples from Cars3D by different object types and elevations, as shown in Figure 10. It shows that with the same value of elevation, samples of different object types have different visual elevation. So these two factors are not fully independent. This might explain the slightly higher co-occurrence of mutual information between this pair of factors.

Shapes3D The result shows relatively bad disentanglement. To be precise, some factor pairs show low mutual information co-occurrence as expected, such as the color factors of floor, wall, and object and the pair of "object color - azimuth". But the MI co-occurrence of "wall color - object size" and "object color - object size;" are higher than we expected as we did not recognize their high dependence. This result might relate to our model's relatively poor performance on Shapes3D.

B.4 Manipulating Factors

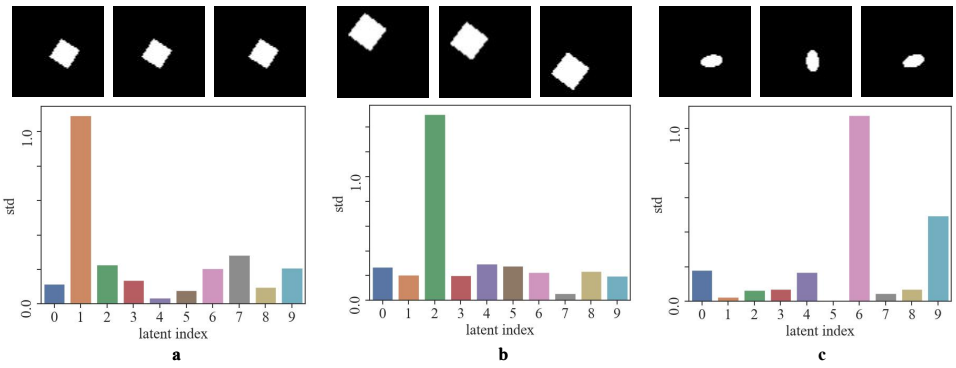


Figure 11: Representation variation when manipulating one factor only in the dimension-reduced version. In (a) and (b), *position_x* and *position_y* are manipulated respectively and only cause one dimension significantly variate. While, in (c), when manipulating the ill-defined factor *orientation*, two dimensions variate.

In the main paper, we studied the influence to representation by manipulating the factors, where the representation is reduced by selecting dimensions as in calculating Top-k MED. Here, we do the qualitative study of the influence on representation by manipulating factors in another way but still on dSprites. To make the original high-dimensional representation space more compact, we use the unsupervised dimension reduction by PCA instead, which is more general when the factor pattern is unknown. Here, we reduce the representation dimension by PCA to 10. Note that since the PCA operation mixes the original latent space with a linear combination, it might destroy the existing disentanglement property in the high dimensional space, or enhance the disentanglement if the original high dimensional space is a linear combination of the ground truth factors. But such influence is usually considered secondary to the disentanglement learned by a model. No matter which case, if the dimension-reduced representation shows disentangled properties, the original space at least captures linearly transformed ground truth factors, and the dimension reduction techniques such as PCA can make the representation more compact in a qualitative study.

Figure 11 shows the result of representation vector variation when changing only one factor at once. Given three images with only one factor's value being different, we generate the 10-dim representation vectors from them. Then, we compute the variance across the three vectors, leading to 10 scalars. The larger the variance is, the more that dimension responds to the factor change. Figure 11(a) and (b) show how reduced representation vector changes when manipulating *position_x* and *position_y*

factor respectively. It shows good disentanglement that only one representation dimension has high variation. However, in Figure 11(c) we show a failure mode of the ill-defined factor *orientation* that change of factor causes both the 6th and the 9th dimensions of reduced representation to have large variations. From the results, we observe that manipulating one well-defined independent factor causes evident variance in only one dimension. And it shows that we could make the learned representation vector more compact by unsupervised dimension reduction.

Table 4: Evaluation results on multiple datasets with different disentanglement metrics.

	Model	BetaVAE	FactorVAE	MIG	SAP	DCI	MED
dSprites	β-VAE	82.3 (7.6)	65.8 (9.2)	26.3 (11.0)	5.2 (2.7)	39.3 (13.2)	32.6 (10.0)
	β-TCVAE	86.7 (2.4)	76.6 (7.8)	23.8 (6.8)	6.9 (0.9)	36.3 (7.1)	31.8 (7.4)
	FactorVAE	84.9 (2.8)	75.3 (7.4)	18.4 (9.0)	6.8 (0.8)	28.8 (10.6)	32.5 (10.1)
	DIP-VAE-I	82.7 (3.3)	59.1 (4.8)	9.6 (5.1)	5.2 (2.6)	14.4 (4.6)	18.8 (5.6)
	DIP-VAE-II	81.5 (4.9)	58.6 (7.6)	7.4 (3.4)	3.6 (2.2)	12.3 (5.2)	14.7 (5.5)
	AnnealedVAE	86.5 (0.1)	60.1 (0.0)	**35.2 (1.3)**	7.6 (0.5)	37.9 (2.1)	**35.8 (0.8)**
	Ada-GVAE	88.0 (2.7)	73.1 (3.9)	17.3 (4.7)	6.6 (2.0)	32.3 (4.6)	–
	SlowVAE	87.0 (5.1)	75.2 (11.1)	28.3 (11.5)	4.4 (2.0)	47.7 (8.5)	–
	EBM	82.3 (2.0)	65.7 (12.5)	1.7 (0.5)	3.0 (1.2)	19.1 (1.8)	6.8 (4.0)
	InfoGAN-CR	85.5 (1.0)	88.0 (1.0)	19.8 (3.2)	6.0 (1.0)	14.0 (5.2)	–
	BYOL	**93.2 (0.4)**	**91.6 (0.8)**	29.3 (0.4)	**8.0 (0.4)**	**66.9 (0.2)**	31.3 (0.4)
Cars3D	β-VAE	**100.0 (0.0)**	89.3 (1.2)	11.7 (1.1)	1.4 (0.9)	38.7 (4.6)	29.0 (2.2)
	β-TCVAE	**100.0 (0.0)**	92.2 (2.7)	**15.5 (2.9)**	1.7 (0.3)	42.7 (3.5)	**33.0 (3.8)**
	FactorVAE	**100.0 (0.0)**	91.7 (4.1)	10.6 (2.2)	**2.0 (0.5)**	29.0 (6.7)	29.1 (3.0)
	DIP-VAE-I	**100.0 (0.0)**	90.5 (5.0)	5.9 (2.8)	1.9 (1.4)	22.6 (5.6)	19.4 (3.3)
	DIP-VAE-II	**100.0 (0.0)**	85.0 (6.1)	5.1 (2.7)	1.3 (0.8)	20.8 (5.4)	16.7 (4.1)
	AnnealedVAE	**100.0 (0.0)**	85.0 (4.3)	7.6 (1.0)	1.5 (0.5)	18.5 (4.3)	15.5 (2.5)
	SlowVAE	**100.0 (0.0)**	90.4 (0.5)	15.4 (2.2)	1.6 (0.5)	48.0 (2.4)	–
	BYOL	**100.0 (0.0)**	**95.8 (1.2)**	7.6 (0.9)	1.8 (0.7)	**48.5 (2.3)**	9.7 (0.5)
SmallNORB	β-VAE	84.1 (2.7)	60.1 (2.4)	25.0 (1.1)	11.4 (1.1)	32.6 (0.6)	24.4 (0.7)
	β-TCVAE	84.5 (2.7)	60.3 (2.3)	25.4 (0.9)	11.7 (1.1)	35.2 (0.7)	25.0 (0.9)
	FactorVAE	80.8 (3.8)	62.5 (3.6)	23.9 (2.0)	10.2 (0.9)	33.4 (1.1)	**25.9 (1.2)**
	DIP-VAE-I	84.2 (3.2)	69.8 (4.6)	24.3 (2.7)	10.2 (1.4)	30.0 (2.1)	24.5 (2.1)
	DIP-VAE-II	85.2 (1.3)	58.4 (2.1)	**25.5 (1.5)**	**14.4 (0.4)**	32.3 (0.7)	24.4 (0.7)
	AnnealedVAE	60.8 (6.2)	50.0 (9.9)	9.1 (2.2)	6.8 (0.8)	15.7 (6.4)	5.5 (3.7)
	SlowVAE	78.2 (3.8)	47.0 (2.9)	23.8 (1.8)	7.8 (1.1)	28.7 (0.7)	21.8 (1.3)
	EBM	79.0 (4.4)	57.9 (3.5)	1.7 (0.5)	1.9 (0.1)	13.9 (2.2)	2.3 (1.7)
	BYOL	**97.0 (0.8)**	**81.0 (0.5)**	3.3 (0.9)	2.2 (0.3)	**51.0 (1.0)**	7.7 (0.2)
Shapes3D	β-VAE	98.6	83.9	22.0	6.2	58.8	12.9 (3.5)
	β-TCVAE	99.8	86.8	27.1	7.9	70.9	**13.7 (0.9)**
	FactorVAE	94.2	82.5	27.0	6.1	67.2	0.7 (0.9)
	DIP-VAE-I	95.6	79.7	15.2	4.0	55.9	10.3 (0.9)
	DIP-VAE-II	97.8	88.4	18.1	6.3	41.9	–
	AnneledVAE	86.1	80.9	35.9	6.2	47.4	–
	Ada-ML-VAE	**100.0**	**100.0**	50.9	12.7	94.0	–
	Ada-GVAE	**100.0**	**100.0**	56.2	**15.3**	**94.6**	–
	SlowVAE	**100.0 (0.1)**	97.3 (4.0)	**64.4 (8.4)**	5.8 (0.9)	82.6 (4.4)	–
	EBM	75.9 (11.2)	53.2 (8.7)	5.2 (2.2)	2.8 (1.1)	21.8 (11.0)	2.1 (2.6)
	BYOL	91.5 (3.9)	82.5 (2.4)	5.2 (1.7)	2.8 (0.3)	53.1 (1.5)	6.0 (0.5)
CelebA	VAE	21.5 (3.2)	6.1 (3.8)	0.8 (0.1)	0.9 (0.2)	11.2 (2.3)	3.8 (0.2)
	β-VAE	19.1 (1.9)	5.8 (1.8)	0.1 (0.1)	0.6 (0.2)	8.7 (1.9)	3.3 (0.1)
	β-TCVAE	19.9 (2.3)	9.8 (2.4)	0.6 (0.2)	1.2 (0.3)	3.5 (1.1)	4.7 (0.1)
	FactorVAE	25.3 (3.0)	**12.0 (2.1)**	0.4 (0.1)	0.6 (0.2)	7.1 (0.7)	0.6 (0.6)
	DIP-VAE-I	21.0 (1.9)	9.3 (1.1)	0.2 (0.1)	0.9 (0.3)	13.8 (2.2)	3.6 (0.2)
	InfoGAN-CR	16.8	11.3	1.6	2.8	22.0	–
	BYOL	**35.7 (2.1)**	11.5 (1.1)	**2.6 (0.7)**	**8.2 (0.9)**	**41.0 (1.3)**	**4.8 (0.4)**

C More Quantitative Results

In the main text, we evaluate the disentanglement under our proposed MED metric on multiple datasets. In this section, to provide a more complete understanding of the disentanglement property of contrastive learning without negatives, we report the disentanglement scores with other metrics, such as FactorVAE score, BetaVAE score, MIG, SAP, and DCI Disentanglement, here.

For the results of VAE-based methods, as the large-scale benchmark of Locatello et al. (34) provides the original logs on dSprites, Cars3D, and SmallNORB datasets, we simply report the performance of the best configuration. The original logs on Shapes3D are not available, so we train and evaluate on Shapes3D by ourselves for the MED scores. For scores under other metrics, we report the median disentanglement scores. Some results are from Locatello et al. (35) but the std error is not available. The median performance of SlowVAE is from its original paper (25). For the results of CelebA, the result of InfoGAN-CR is from its officially released checkpoint without availability to the std error. For other methods, we report the mean value of our trained weights over three random seeds as default. Because the evaluation of DCI is extremely time-consuming, around 14 hours for a 1000-d model, we only take BYOL as an example here for negative-free contrastive learning methods. All results are combined and shown in Table 4.

Same as the analysis we provide in the main text, the results show significant disagreement among the existing metrics. To be precise, for those metrics (BetaVAE score, FactorVAE score, SAP, DCI Disentanglement) using a learnable model such as a regressor or classifier, the high-dimensional BYOL model achieves a significant advantage. However, for the metrics relying on only one or two dimensions to reveal the connection between a latent dimension and a factor (MIG and MED), BYOL's performance is not that impressive anymore.

Finally, the result on CelebA shows the great robustness of BYOL's learned representations to show disentanglement on real-world datasets. Yet, the large gap between the score of those on synthetic datasets emphasizes the difficulty of learning disentangled factors on real-world images. It is hard to empirically study whether it is the high dimension that gives BYOL advantages on some metrics because the nature of BYOL makes it hard to be trained with a small latent dimension to make a comparison.

D Ablation Study

Limited by the main content page length, we put some additional ablation studies here to help better understand the influence of important inductive bias of BYOL when studying representation disentanglement.

Table 5: Results of using different normalization strategies on dSprites.

normalization	w/o norm	BN	GN	LN	IN
MED	23.8 (0.6)	29.4 (0.5)	31.3 (0.4)	31.3 (0.8)	0.0 (0.0)

D.1 Normalization

We experiment with five normalization layers configuration in the encoder network on the dSprites dataset. The results are shown in Table 5. For group normalization, we set the group number to 4. On dSprites, we find the commonly used BN decreases the disentanglement performances. By keeping the batch norm in the projector and the predictor, removing the batch norm in the encoder will not cause the model to collapse, which agrees with the observation in previous works (40). On the contrary, replacing batch norm in encoder with group norm or layer norm will increase the representation disentanglement while achieving similar accuracy in downstream factor prediction. We notice that a similar phenomenon has been discovered before in supervised representation disentanglement. For example, Bau et al. (3) discovered that a network trained with batch normalization layers has less interpretable (disentangled) neurons. On the other hand, instance norm (47) completely breaks the contrastive learning process. We still do not fully understand this behavior, but we hypothesize that it may be caused by the shared batch statistics that make it hard for a feature to be aligned to the ground truth factor.

E Limitations

Our work still has some limitations, especially considering the design of contrastive learning methods still depends on heavy empirical practice. For the fair comparison in the benchmark, we use a shared encoder architecture for all methods but they may still have other inductive bias potentially influencing the results such as the hyperparameters in VAE-based methods and contrastive learning methods. We select the normalization in BYOL as an example in the ablation study above showing that such inductive bias can make influence over the evaluation results. But we can do an ablation study on all possible inductive bias. We basically inherit the available best settings from DisLib ([34]) if possible and the settings from the public official implementations of other methods. All hyperparameters and details we customize have been indicated in the Section of Reproducibility.

References

[1] A. Achille, T. Eccles, L. Matthey, C. P. Burgess, N. Watters, A. Lerchner, and I. Higgins. Life-long disentangled representation learning with cross-domain latent homologies. *arXiv preprint arXiv:1808.06508*, 2018.

[2] S. Arora, H. Khandeparkar, M. Khodak, O. Plevrakis, and N. Saunshi. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*, 2019.

[3] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6541–6549, 2017.

[4] Y. Bengio, Y. LeCun, et al. Scaling learning algorithms towards ai. *Large-scale kernel machines*, 34(5):1–41, 2007.

[5] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

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

[7] R. T. Q. Chen, X. Li, R. Grosse, and D. Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems*, 2018.

[8] X. Chen and K. He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021.

[9] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 2180–2188, 2016.

[10] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox. Discriminative unsupervised feature learning with convolutional neural networks. *Advances in neural information processing systems*, 27:766–774, 2014.

[11] C. Eastwood and C. K. Williams. A framework for the quantitative evaluation of disentangled representations. In *ICML*, 2018.

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[13] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[14] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

[15] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

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

[17] A. Hyvarinen and H. Morioka. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. *Advances in Neural Information Processing Systems*, 29:3765–3773, 2016.

[18] A. Hyvärinen and E. Oja. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430, 2000.

[19] A. Hyvarinen, H. Sasaki, and R. Turner. Nonlinear ica using auxiliary variables and generalized contrastive learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 859–868. PMLR, 2019.

[20] T. Jakab, A. Gupta, H. Bilen, and A. Vedaldi. Unsupervised learning of object landmarks through conditional image generation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 4020–4031, 2018.

[21] I. Khemakhem, D. Kingma, R. Monti, and A. Hyvarinen. Variational autoencoders and nonlinear ica: A unifying framework. In *International Conference on Artificial Intelligence and Statistics*, pages 2207–2217. PMLR, 2020.

[22] I. Khemakhem, R. P. Monti, D. P. Kingma, and A. Hyvärinen. Ice-beem: Identifiable conditional energy-based deep models based on nonlinear ica. *arXiv preprint arXiv:2002.11537*, 2020.

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

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

[25] D. Klindt, L. Schott, Y. Sharma, I. Ustyuzhaninov, W. Brendel, M. Bethge, and D. Paiton. Towards nonlinear disentanglement in natural data with temporal sparse coding. *arXiv preprint arXiv:2007.10930*, 2020.

[26] T. D. Kulkarni, W. Whitney, P. Kohli, and J. B. Tenenbaum. Deep convolutional inverse graphics network. *arXiv preprint arXiv:1503.03167*, 2015.

[27] A. Kumar, P. Sattigeri, and A. Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations. *arXiv preprint arXiv:1711.00848*, 2017.

[28] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman. Building machines that learn and think like people. *Behavioral and brain sciences*, 40, 2017.

[29] Y. LeCun, F. J. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2:II–104 Vol.2, 2004.

[30] J. D. Lee, Q. Lei, N. Saunshi, and J. Zhuo. Predicting what you already know helps: Provable self-supervised learning. *arXiv preprint arXiv:2008.01064*, 2020.

[31] Z. Lin, K. Thekumparampil, G. Fanti, and S. Oh. Infogan-cr and modelcentrality: Self-supervised model training and selection for disentangling gans. In *International Conference on Machine Learning*, pages 6127–6139. PMLR, 2020.

[32] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015.

[33] Z. Liu, P. Luo, X. Wang, and X. Tang. Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, 15(2018):11, 2018.

[34] F. Locatello, S. Bauer, M. Lucic, G. Raetsch, S. Gelly, B. Schölkopf, and O. Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *ICML*, 2019.

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

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

[37] J. Peters, D. Janzing, and B. Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017.

[38] S. Purushwalkam and A. Gupta. Demystifying contrastive self-supervised learning: Invariances, augmentations and dataset biases. *arXiv preprint arXiv:2007.13916*, 2020.

[39] S. E. Reed, Y. Zhang, Y. Zhang, and H. Lee. Deep visual analogy-making. *Advances in neural information processing systems*, 28:1252–1260, 2015.

[40] P. H. Richemond, J.-B. Grill, F. Altché, C. Tallec, F. Strub, A. Brock, S. Smith, S. De, R. Pascanu, B. Piot, et al. Byol works even without batch statistics. *arXiv preprint arXiv:2010.10241*, 2020.

[41] K. Ridgeway and M. C. Mozer. Learning deep disentangled embeddings with the f-statistic loss. *arXiv preprint arXiv:1802.05312*, 2018.

[42] J. Schmidhuber. Learning factorial codes by predictability minimization. *Neural computation*, 4(6):863–879, 1992.

[43] Y. Tian, C. Sun, B. Poole, D. Krishnan, C. Schmid, and P. Isola. What makes for good views for contrastive learning? *arXiv preprint arXiv:2005.10243*, 2020.

[44] C. Tosh, A. Krishnamurthy, and D. Hsu. Contrastive learning, multi-view redundancy, and linear models. In *Algorithmic Learning Theory*, pages 1179–1206. PMLR, 2021.

[45] Y.-H. H. Tsai, Y. Wu, R. Salakhutdinov, and L.-P. Morency. Demystifying self-supervised learning: An information-theoretical framework. *arXiv e-prints*, pages arXiv–2006, 2020.

[46] M. Tschannen, O. Bachem, and M. Lucic. Recent advances in autoencoder-based representation learning. *arXiv preprint arXiv:1812.05069*, 2018.

[47] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6924–6932, 2017.

[48] S. van Steenkiste, F. Locatello, J. Schmidhuber, and O. Bachem. Are disentangled representations helpful for abstract visual reasoning? *arXiv:1905.12506*, 2019.

[49] T. Wang and P. Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR, 2020.

[50] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018.

[51] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021.

[52] N. Zhao, Z. Wu, R. W. Lau, and S. Lin. What makes instance discrimination good for transfer learning? *arXiv preprint arXiv:2006.06606*, 2020.

[53] R. S. Zimmermann, Y. Sharma, S. Schneider, M. Bethge, and W. Brendel. Contrastive learning inverts the data generating process. *arXiv:2102.08850*, 2021.