DISENTANGLING PROPERTIES OF CONTRASTIVE METHODS

Anonymous authors
Paper under double-blind review

ABSTRACT

Disentangled representation learning is an important topic in representation learning, since it not only allows the representation to be human interpretable, but it is also robust and benefits downstream task performance. Prior methods achieved initial successes on simplistic synthetic datasets but failed to scale to complex real-world datasets. Most of the previous methods adopt image generative models, such as GAN and VAE, to learn the disentangled representation. But we observe they are hard to learn disentangled representation on real-world images. Recently, self-supervised contrastive methods such as MoCo, SimCLR, and BYOL have achieved impressive performances on large-scale visual recognition tasks. In this paper, we explored the possibility of using contrastive methods to learn a disentangled representation, a discriminative approach that is drastically different from previous approaches. Surprisingly, we find that the contrastive method learns a disentangled representation with only minor modifications. The contrastively learned representation satisfies a "group disentanglement" property, which is a relaxed version of the original disentanglement property. This relaxation might be useful for scaling disentanglement learning to large and complex datasets. We further find contrastive methods achieve state-of-thet-art disentanglement performance on several widely used benchmarks, such as dSprites and Car3D. It also achieves significantly higher performance on the real-world dataset CelebA.

1 INTRODUCTION

Learning a disentangled representation is a long-desired goal in the deep learning community (Bengio et al., 2013; Peters et al., 2017; Goodfellow et al., 2016; Bengio et al., 2007; Schmidhuber, 1992; Lake et al., 2017; Tschannen et al., 2018). A disentangled representation matches how humans understand the world and provides many other benefits besides model interpretability (Bengio et al., 2013; Chen et al., 2016; Kulkarni et al., 2015). And it usually needs much fewer labels to learn challenging downstream tasks (van Steenkiste et al., 2019). It also generalizes much better even in face of examples generated by an unseen combination of the attribute values (Achille et al., 2018).

Given its importance and potential large impacts on downstream applications, disentangled representation learning has recently attracted great attention. Previous research has proposed a lot of methods, either built on variational auto-encoders (Kingma & Welling, 2013), such as β-VAE (Higgins et al., 2016) and FactorVAE (Kim & Mnih, 2018), or generative adversarial network (Goodfellow et al., 2014), such as InfoGAN (Chen et al., 2016) and InfoGAN-CR (Lin et al., 2020). Those methods have achieved preliminary successes on synthetic datasets such as dSprites (Matthey et al., 2017) and 3Dshapes (Burgess & Kim, 2018).

Albeit the initial successes, those synthetic datasets are limited in many aspects, for example, the background is usually clean and composed of a single color, the number of objects is small and the objects are mainly 2D without texture and occlusions. It is still an open question to scale those models to real-world complex datasets. It is even non-trivial to scale basic generative models to complex datasets to learn disentangled features, such as ImageNet (Deng et al., 2009).

In this paper, instead of studying disentanglement feature learning with a generative model, we investigate whether the discriminatively-trained contrastive models have the disentanglement property. Contrastive learning is a class of self-supervised learning methods that pull two augmentations of the same image close. The recent contrastive methods have achieved state-of-the-art performance on

the image pretraining tasks. Representative methods include MoCo (He et al., 2020), BYOL (Grill et al., 2020), SimCLR (Chen et al., 2020) and SwAV (Caron et al., 2020) etc. Contrastive learning has been proven to learn good visual representations from large-scale datasets. We continue to investigate the disentanglement property of their learned representations in this work.

To our surprise, we find that the widely used BYOL algorithm without any auxiliary loss exhibits strong feature disentanglement property. However, the disentanglement of contrastive learning is a weaker form of the disentanglement. It follows the pattern that a representation dimension is disentangled to correspond to a single factor but a single ground truth factor might appear in multiple latent feature dimensions. We name this type of disentanglement as "group disentanglement". Although group disentanglement is a weaker form of disentanglement, we hypothesize that directly learning a compact and disentangled representation might be hard, due to the lottery ticket hypothesis (Frankle & Carbin, 2018). Pursuing group disentanglement instead of full disentanglement might be necessary to achieve disentanglement on the complex real-world dataset. The reason is probably that real-world images usually contain more details and noise, and a "factor" might be always correlated to some other visual existence due to dataset bias. A good example is a ship that usually comes together with a large area of blue background in existing datasets.

Further, we find that contrastively trained representation achieves the state-of-the-art FactorVAE disentanglement score when evaluated on established benchmarks, such as Car3D (Reed et al., 2015), dSprites (Matthey et al., 2017) and SmallNORB (LeCun et al., 2004b). Besides the widely used synthetic benchmarks, we also evaluate the contrastive method on the CelebA (Liu et al., 2018) human faces real-world dataset. We find that it also achieves better or comparable performances than the other methods on five commonly used disentanglement metrics.

In summary, our contributions in this paper are:

1. We show that a contrastive method, in particular BYOL, learns "group-disentangled" representations, without any extra auxiliary losses.
2. The contrastive method achieves the state-of-the-art performance on several widely used disentanglement learning benchmarks.
3. We propose to quantitatively evaluate disentanglements on a real-world dataset, which avoids the biases of synthetic images. Our contrastive method also achieves the state-of-the-art performance on this benchmark.

2 Related Works

Disentangled Representation Learning Disentangled representation is important (Goodfellow et al., 2016; Bengio et al., 2007; Schmidhuber, 1992; Lake et al., 2017; Tschannen et al., 2018), since it learns a human interpretable intermediate representation (Bengio et al., 2013; Chen et al., 2016; Kulkarni et al., 2015); it enables the downstream task to be learned more easily (van Steenkiste et al., 2019); it potentially generalizes better to novel combinations of the factors (Achille et al., 2018); and it can help to answer counterfactual questions (Pearl, 2009; Spirtes et al., 2000; Peters et al., 2017). In this paper, we consider the fully unsupervised disentangled representation learning setting, i.e. we assume no annotations on which factors should be learned.

Recently, methods built on VAE (Kingma & Welling, 2013) have achieved successes on the disentangled representation learning task, such as β-VAE (Higgins et al., 2016), factorVAE (Kim & Mnih, 2018), β-TCVAE (Chen et al., 2018) and DIP-VAE (Kumar et al., 2017). Those methods usually enforce a factorized constraint on the posterior of the latent code in the VAE. Another line of work builds on GAN (Goodfellow et al., 2014) to learn disentangled representations. InfoGAN (Chen et al., 2016) is one of the representative works, which maximize the information and the generated image. More recently, people have shown that the GAN-based approach can achieve competitive performance as the above VAE variants (Lin et al., 2020; Jeon et al., 2018; Lee et al., 2020b).

Under the context and self-supervised learning, Zimmermann et al. (2021) suggests that the contrastive method can invert the data generation process. They also try to connect contrastive learning and independent component analysis. However, the study is limited under strong assumptions and no quantitative analysis has been provided. Besides, their theory critically relies on the existence of negative samples for contrastive learning. Compared with that, our experiments mainly deal with

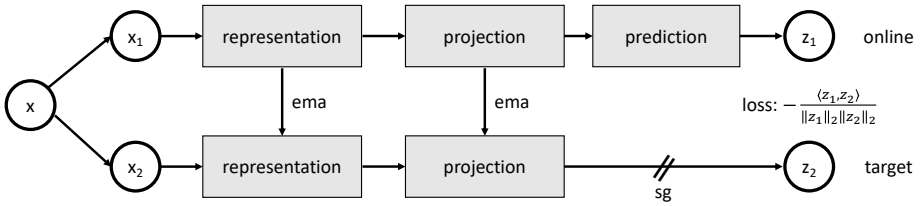

Figure 1: The BYOL network architecture. It has a online stream and a target stream. The online stream has a representation-projection-prediction pipeline while target stream lacks the prediction part. The target branch is updated by exponential moving average of the online branch. The model is trained by encouraging output of two streams from a positive pair of views to get close.

self-supervised methods without negative examples and we build a series of quantitative benchmarks on diverse synthetic and even real-world datasets.

Contrastive Learning Contrastive learning methods such as SimCLR (Chen et al., 2020), MoCo (He et al., 2020) and BYOL (Grill et al., 2020) have achieved great successes to learn good visual representation from no label. They create "views" by applying augmentations over images. They treat two views of the same image as "positive pairs", and views of all the other images as negatives. This setup is also known as examplar classification (Dosovitskiy et al., 2014), or instance discrimination (Wu et al., 2018). Representation learned by contrastive learning has shown great transfer capabilities to downstream tasks, such as object detection and semantic segmentation.

More recently, there are a lot of works trying to understand contrastive learning either theoretically (Wang & Isola, 2020; Arora et al., 2019; Tsai et al., 2020; Tosh et al., 2021; Tian et al., 2020b; Lee et al., 2020a), or empirically (Tian et al., 2020a; Zhao et al., 2020; Purushwalkam & Gupta, 2020). However, none of them has studied whether contrastive methods would encourage disentanglement or not. Our work bridges contrastive learning with the previous line of disentanglement research empirically and extends the quantitative study onto real-world datasets.

3 METHOD

In this paper, we explore whether contrastive methods learn a disentangled feature representation. If yes, under what condition it learns a disentangled representation. There are quite a few contrastive learning algorithms proposed (Grill et al., 2020; He et al., 2020; Chen et al., 2020; Caron et al., 2020). Although they differ on some specific aspects, they all aim to pull two augmentations of one image close. Without the uniformity property provided by negative samples (Wang & Isola, 2020), we find BYOL (Grill et al., 2020) still achieves unexpected good disentangled feature representation.

3.1 BYOL METHOD

BYOL is an unsupervised learning method that pulls two augmentation views of the same image close (Grill et al., 2020) to learn a high-level image representation. A significant difference of it against other contrastive learning is the absence of negative pairs during training. As shown in Figure 1, for each image x, we obtain two views of it: x_1 and x_2 by data augmentations. One of them goes through the online network stream, and the other goes through the target network stream. The target network's parameter is not trained by the gradient descent algorithm but set as the exponential moving average (ema) of the online network. Both the online and the target stream have a representation network (encoder) and a projection network. The online network has an extra prediction network after the projection network. The online stream's output z_1 and the target stream's output z_2 are pulled close to each other by requiring the two vectors to have similar directions in the latent space. More specifically, the loss function is

$$\mathcal{L} = -\frac{\langle z_1, z_2 \rangle}{\|z_1\|_2 \|z_2\|_2}.$$

In practice, the online representation network (without the projection or the prediction network) is usually the representation model for the downstream tasks. In this work, we follow the same convention, i.e. we study the disentanglement property of the output of the representation network.

3.2 Implementation Details

We follow most of the implementation details in the original BYOL paper, but we find that several details can be changed to achieve better feature disentanglement. For full implementation details please refer to the appendix A.1.

Architecture To make a fair comparison, we follow the encoder architecture used in FactorVAE (Kim & Mnih, 2018) for all synthetic datasets, except that the latent code dimension is set to be 1000, since the contrastive learning requires a large latent code dimension to work (Grill et al., 2020). On CelebA, encoders are unified to be ResNet-50 (He et al., 2016).

Normalization By default, BYOL uses batch normalization throughout the network. However, we find that batch norm in encoder can reduce the feature disentanglement in some cases (see Section 5.5 for ablations). Thus we replace all batch norm with group normalization (Wu & He, 2018). Without notion otherwise, the group number is set to 4 in our implementation by default.

Augmentation Augmentation is a crucial component for unsupervised learning methods as it provides a feasible way to create positive pairs in contrastive learning. In this work, we follow the default data augmentation used in BYOL, i.e., the composition of color jitter, graying, horizontal flip, gaussian blur, and random resize crop. We recognize that the parameter of random resize crop size is critical to learn disentangled features, which we will discuss in the ablation study.

4 Major Results

In this section, we summarize our major empirical findings. Section 5 will then continue to present experimental evidence to support these claims.

4.1 The Group Disentanglement Property

We find that the contrastive feature representation exhibits some form of disentanglement property: different ground truth factors map to different sets of latent dimensions, but multiple latent dimensions are corresponding to a single ground truth factor. We name this type of disentanglement as "group disentanglement". See Section 5.2 for experimental evidence to support this characteristic.

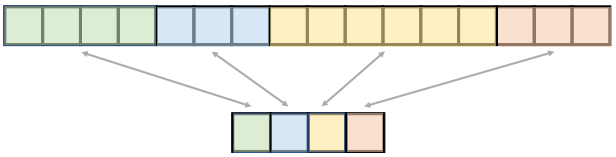

Figure 2: Illustration of group disentanglement. The lower vector denotes four ground truth factors and the upper vector represents the learned group disentangled feature vector. Each ground truth factor may correspond to multiple feature dimensions, but each feature dimension only correspond to one factor.

While there is not a single widely accepted definition of the original disentanglement concept, we describe it intuitively as (1) **Disentanglement**: one dimension in the representation vector only represents one ground truth factor (2) **Compactness**: one ground truth factor only corresponds to one dimension in the representation (3) **Completeness**: no ground truth factor is left out in the representation. Group disentanglement relaxes the compactness requirement, but only requires the disentanglement and the completeness. Figure 2 shows how group disentanglement looks like.

Group disentanglement is a weaker form of disentanglement, but we hypothesis that it might be a better proxy of disentanglement to scale to complex and realistic datasets. Directly learning the ideal compact disentanglement representation with neural network might be very hard because of the lottery ticket hypothesis (Frankle & Carbin, 2018). The lottery ticket hypothesis states that usually a well-trained neural network can be pruned to a much smaller network while keeping the same accuracy. However, it is much harder to directly train the same small network from scratch when randomly initialized. The hypothesis encourages a large randomly initialized network from start to contain a subset of the network that is the "winning ticket". We suspect that a compact disentangled representation is also hard to obtain for the same reason. Instead, group disentanglement might be a much more realistic goal to achieve.

4.2 COMPETITIVE PERFORMANCE ACROSS BENCHMARKS AND METRICS

To quantitatively evaluate the disentanglement properties of the contrastive methods, we evaluate with existing benchmarks (Cars3D (Reed et al., 2015), dSprites (Matthey et al., 2017), SmallNORB (LeCun et al., 2004a), Shapes3D (Burgess & Kim, 2018)) with various metrics. We find that although contrastive learning is not designed to learn disentangled feature representation, it still achieves better or comparable disentangled features when compared to existing specialized methods. We even build benchmarks on the real-world CelebA dataset for the first time, and the performance of contrastive learning is still robust to be top. See Section 5.3 for experimental support.

4.3 BATCH NORM DISCOURAGES FEATURE DISENTANGLEMENT

We find that batch norm in encoder consistently decreases the feature disentanglement level compared to no normalization layer (Section 5.5). By keeping the batch norm in the projector and the predictor, removing batch norm in the encoder will not cause model collapsing, which agrees with the observation in previous works (Richemond et al., 2020). On the contrary, replacing batch norm in encoder with group norm or layer norm will increase the feature disentanglement while achieving similar accuracy in downstream factor prediction. We notice that a similar phenomenon has been discovered before in supervised feature disentanglement. For example, Bau et al. (2017) discovered that a network trained with batch normalization layers has less interpretable (disentangled) neurons. 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.

5 EXPERIMENTS

In this section, we show the quantitative results to support the described observations above. We first introduce the experiment setup, including benchmarks, metrics, baseline models, and our implementation details. Some details are provided in the appendix. Then, we provide qualitative analysis about the learned representation pattern by contrastive learning. At last, we list a series of quantitative benchmark experiments to prove the good disentanglement property of BYOL on both synthetic and real-world datasets. Besides the main experiments in this section, more ablation experiments are provided in the appendix.

In this section, we aim to empirically understand the disentanglement properties of the contrastive method following the four questions. (1) What does the contrastive method learn in its latent representation? Is it disentangled? Can we visualize the latent space? (2) Quantitatively, how disentangled is the representation of contrastive methods? (3) How about real-world datasets? (4) What set of hyper-parameters best promotes the learned representation to be disentangled?

5.1 EXPERIMENTS SETUP

In this section, we describe the experimental setup, including the datasets we evaluate, the metrics for a quantitative study, and the previous disentangled feature learning algorithms for comparison.

Datasets Previous works evaluate representation disentanglement only on synthetic datasets, such as dSprites, Cars3D, SmallNORB, and Shape3D. Besides those datasets, we also include a real-world dataset CelebA. CelebA contains human celebrity faces images with 40 binary attributes annotations. The attributes include fine-grained properties of the human face such as whether wearing glasses or whether has wavy hair. Table 7 in Appendix explains the dataset details.

Metrics We follow `disentanglement_lib` (Locatello et al., 2019) to use five popular metrics to evaluate each method. We leave out the Modularity metric since Locatello et al. (2019) suggests that Modularity is inconsistent with other metrics. Specifically, we use BetaVAE, FactorVAE, Mutual Information Gap (MIG), SAP, and DCI metrics. Please refer to Appendix A.4 for details.

Reference Methods Most of the previous state-of-art disentangled feature representation learning methods are either built on VAE (Kingma & Welling, 2013) or GAN (Goodfellow et al., 2014). β-VAE (Higgins et al., 2016) introduces a hyper-parameter to adjust the KL constraint in VAE. FactorVAE (Kim & Mnih, 2018) and β-TCVAE use adversarial training to reduce the correlations on different dimensions of the latent code. DIP-VAE (Kumar et al., 2017) pull the posterior to a factorized prior. Besides the VAE-based methods, we also investigate Generative Adversarial Networks (GANs). InfoGAN (Chen et al., 2016) encourages the latent code to have high mutual

information with the generated image. IB-GAN (Jeon et al., 2018) and InforGAN-CR (Lin et al., 2020) add more constraint on InfoGAN to further promote disentanglement.

5.2 Understanding the Learnt Representation

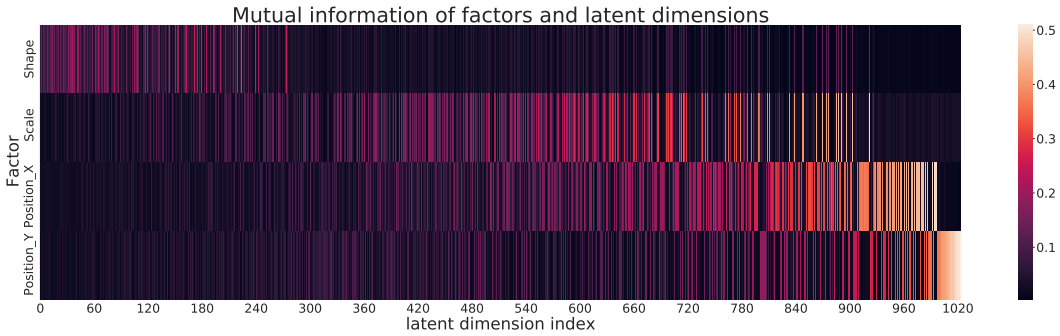

Figure 3: The mutual information heatmap between factors and latent code dimensions. It explains the pattern of "group disentanglement" that a factor shares high mutual information with multiple dimensions of representation instead of only one dimension described by the perfectly compact disentanglement. On the other hand, a certain representation dimension will not have high mutual information with more than one factor.

In this part, we make the qualitative study of the disentanglement of representation learned by BYOL. Since contrastive methods are not generative models, it is hard to directly do factor-controlled pixel-wise reconstruction for visualization. We thus turn to measure the mutual information between learned feature code and the ground truth factors to study that precisely. The model is analyzed on dSprites dataset. dSprites has in total five factors (shape, scale, orientation, position_x, and position_y). But as explained in Appendix A.3, the orientation is ill-defined with ambiguity. For example, it is impossible to distinguish if a square rotates 0 degrees or 180 degrees. Therefore, we ignore this factor temporarily in the following discussion.

After encoding an input image to a representation vector, we compute the mutual information between each factor and each representation dimension. The mutual information between all dimensions and all four left factors are included in Figure 3. It shows an evident pattern that a ground truth factor can correspond to multiple representation dimensions but a single dimension of representation only has the high response to one factor. To have a more intuitive understanding of to what extent a representation dimension may respond to more than one factor, we define a metrics by the normalized co-occurrence of mutual information. Given the mutual information between the representation vector and the ith factor, noted as M_i, M_i^j is the mutual information by the jth dimension. Then, the normalized co-occurrence of mutual information between the ith factor and the kth factor is

$$\widehat{C}_{i,k} = \frac{\langle M_i, M_k \rangle}{||M_i||_2 \cdot ||M_k||_2} = \frac{\sum_{l=0}^{L} M_i^l M_k^l}{||M_i||_2 \cdot ||M_k||_2}.$$

where L is the representation vector length. The definition conforms to the InfoMax (Linsker, 1988) principle of maximizing the mutual information of conceptually correlated pairs. We visualized the normalized co-occurrence of mutual information among the four factors by the learned representation in Figure 4. It agrees with the pattern revealed in Figure 3 that representation code on a certain dimension will not have high mutual information to more than one factor. Moreover, it shows the independence degree of factor pairs. For example, the shape and scale of dSprites objects are actually not fully disentangled

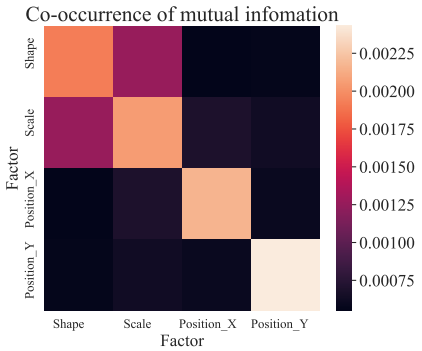

Figure 4: The visualization of normalized co-occurrence of mutual information.

because objects with the same scale value but in different shapes have different pixel area. The co-occurrence mutual information on them is thus slightly higher than other factor pairs.

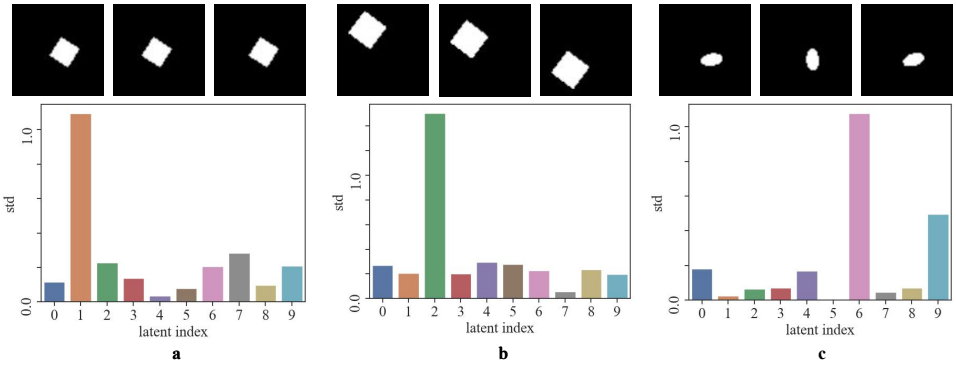

Figure 5: Representation variation when manipulating one factor only in the dimension-reduced version. In (a) and (b), *position_x* and *position_y* are manipulated respectively and only cause one dimension severely variate. While when manipulating the ill-defined factor *orientation*, two dimensions variate. The result shows a more compact disentanglement after reduction of dimension for those well-defined independent factors.

The analysis above recognizes the pattern of "group disentanglement". Then it comes a natural question whether we can get more compact and interpretable representation. Here, we reduce representation dimension by PCA to 10. Figure 5 shows the result of latent variation when changing only one factor at once. Given three images with only one factor's value different, we generate the 10-dim representation vectors from them. Then, we compute the variance across the three vectors. The Figure 5(a) and (b) show how reduced latent code 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 5(c) we show a failure mode of the ill-posed factor *orientation* that change of factor causes both 6th and 9th dimension of reduced representation to have large variations. From the results, we observe that manipulating one well-defined independent factor causes evident variance on only one dimension. Compared with the original group disentanglement pattern, the dimension-reduced version becomes more compact and interpretable.

To summarize, we have found strong empirical evidence about the "group disentanglement" property of the representation learned by contrastive learning. Moreover, we have shown that the representation can be made more compact through proper unsupervised reduction.

5.3 Comparison on Existing Benchmarks

	Model	dSprites	Cars3D	SmallNORB	Shapes3D
VAE	β-VAE	0.658	0.893	0.601	0.822
	β-TCVAE	0.766	0.922	0.603	0.890
	FactorVAE	0.753	0.917	0.625	0.817
	DIP-VAE-I	0.591	0.905	0.696	0.807
	DIP-VAE-II	0.586	0.850	0.584	**0.893**
GAN	InfoGAN	0.590	–	–	–
	IB-GAN	0.800	–	–	–
	InfoGAN-CR	0.880	–	–	–
CL	BYOL (Ours)	**0.916**	**0.954**	**0.810**	0.833

Table 1: Comparison of contratstive learning (CL) versus generative models. The metric used is the FactorVAE metric. All results are averaged over multiple random seeds. Contrastive learning methods show good disentanglement and BYOL even achieves SoTA on multiple benchmarks.

In order to quantitatively evaluate the disentanglement of contrastive learning, we compare contrastive learning with previous line of disentangled representation learning methods on commonly used benchmarks in this section. In the previous works, a lot of datasets have been proposed to evaluate the disentanglement learning method. Locatello et al. (2019) suggest that new method should be evaluated on a wide range of datasets to minimize potential dataset biases.

In this part, we follow Locatello et al. (2019) to evaluate on the following four datasets: dSprites, Cars3D, SmallNORB and Shapes3D. Table 1 shows the result with FactorVAE score as the evaluation metrics. As the large-scale benchmark of Locatello et al. (2019) provides the original logs on dSprites, Cars3D, and SmallNORB datasets, we simply report the best configuration VAE-based methods trained by them. The original logs on Shapes3D are not available, so we train and evaluate on Shapes3D by ourselves and the output results are well aligned with the reported result in Locatello et al. (2019). The results of InfoGAN, IB-GAN and InfoGAN-CR are reported in Lin et al. (2020). From the results in the table, we could find the representation learned from BYOL surprisingly shows SoTA performance on multiple benchmarks.

	Model	BetaVAE	FactorVAE	MIG	SAP	DCI
VAE	β-VAE	0.823	0.658	0.263	0.052	0.393
	β-TCVAE	0.867	0.766	0.238	0.069	0.363
	FactorVAE	0.849	0.753	0.184	0.068	0.288
	DIP-VAE-I	0.827	0.591	0.096	0.052	0.144
	DIP-VAE-II	0.815	0.586	0.074	0.036	0.123
GAN	InfoGAN	–	0.590	–	–	–
	IB-GAN	–	0.800	–	–	–
	InfoGAN-CR	0.855	0.880	0.198	0.060	0.140
CL	BYOL (Ours)	**0.932**	**0.916**	**0.294**	**0.080**	**0.631**

Table 2: Evaluation of disentanglement on dSprites by different metrics. All results are averaged on multiple random seeds. It shows strong and robust disentanglement property of BYOL.

However, as discovered in (Locatello et al., 2019), the existing disentanglement metrics are argued to misagree with each other in certain cases. To make sure that the contrastive learning does not hack FactorVAE metric here, we continue to show a benchmark on dSprites dataset under five evaluation metrics in Table 2. Here, for all VAE-based methods, we still report the best configuration's performance from Locatello et al. (2019). For InfoGAN-CR, we use the officially released implementation and pretrained weights for evaluation. All evaluation settings follow the configuration of Locatello et al. (2019). The results strongly suggest that contrastive learning can well promote disentangled features. Moreover, it shows the evidence that different metrics can not be precisely aligned as they use different perspectives to measure the disentanglement degree, which itself evens lacks a unified definition. Moreover, as the latent dimension of VAE-based methods is only 128, smaller than that of BYOL (1000-d), we increase the representation dimension of VAEs to 1000 for a more fair comparison. The results are in Table 9 in Appendix, showing that increasing latent dimension does not boost the VAEs' disentanglement. Therefore, the performance gap between BYOL and VAE-based methods does not come from the change of representation dimension.

5.4 Comparison on Real-World Datasets

Although the four datasets above can establish how well each method perform, those datasets are all synthetic and they are limited in various aspects. For example dSprites only has one object per image on a black background; SmallNORB contains synthetic images of 3D objects viewing from different angles, but only contains 50 instances within 5 categories. A high performance on those dataset does not necessarily transfer to real world images, where there might be a lot more variations and unseen objects during test time. To get a clearer sense of how each of the method would perform on the real world datasets, we further evaluate our method and several other methods on CelebA.

For evaluation of MIG and SAP metrics, to make a fair comparison, the representation vector of all methods is reduced to 40 dimensions. The results are shown in Table 3. BYOL continues to show high disentanglement compared with other methods.

	Model	BetaVAE	FactorVAE	MIG	SAP	DCI
VAE	VAE	0.215	0.061	0.008	0.009	0.112
	β-VAE	0.191	0.058	0.001	0.006	0.087
	β-TCVAE	0.199	0.098	0.006	0.012	0.035
	FactorVAE	0.253	**0.120**	0.004	0.006	0.071
	DIP-VAE-I	0.210	0.093	0.002	0.009	0.138
GAN	InfoGAN-CR	0.168	0.113	0.016	0.028	0.220
CL	BYOL (Ours)	**0.357**	0.115	**0.026**	**0.082**	**0.410**

Table 3: Disentanglement evaluation on the CelebA dataset. The result shows great robustness of BYOL's learned representations to show disentanglement on real-world datasets. Yet, the large gap between the score from that on synthetic datasets emphasises the difficulty of learning disentangled factors on real-world images.

normalization	w/o norm	BN	GN	LN	IN
FactorVAE score	0.896	0.850	**0.916**	0.914	0.191

Table 4: Results of using different normalization strategies on dSprites. For group normalization, we set group number to 4. BYOL collapses with instance normalization (IN).

5.5 Effects of Hyper-Params

Hyper-parameters significantly affect the performance of contrastive learning. In this section, we systematically study the influence on disentanglement property from some recognized crucial inductive bias for contrastive learning.

The batch normalization in BYOL has been considered the reason why BYOL does not collapse even without negative pairs for long. However, a previous work (Richemond et al., 2020) finds BYOL can avoid collapsing even without batch statistics. But the normalization strategy between layers is still recognized as a key variant of BYOL. Thus we experiment with five normalization layers configuration in the encoder network on the dSprites dataset. The results are shown in Table 4. We find the commonly used BN actually decreases the disentanglement performances. No normalization already does good regarding to feature disentanglement. Group norm (Wu & He, 2018) and layer norm (Ba et al., 2016) are two best normalization techniques. On dSprites, instance norm (Ulyanov et al., 2017) completely breaks the contrastive learning process. We provide more study on normalization in Table 11 and Table 12 in Appendix where more variables, such as batch size, are taken into consideration. From all the study, it still remain unclear why batch norm could harm the feature disentanglement. However, this phenomenon was also empirically observed in the supervised learning literature as well. For example, in the Bau et al. (2017) paper, the author shows that CNN with batch norm layer has significantly less interpretable concepts than those without BN.

We also perform ablation study on other hyper-parameters. For example, we find the representation trained on one dataset can show transferrable disentanglement to other datasets and BN causes model collapse with big learning rate (Table 10). More details are provided in Appendix B.

6 Conclusion

In this paper, we show some empirical study on the disentanglement property by contrastive learning. We find with minor modification, the existing BYOL framework can achieve state-of-the-art disentanglement performance on multiple benchmarks and under diverse quantitative metrics. Besides, the qualitative study reveals the different "disentanglement" nature of contrastive learning that the representation actually shows "group disentanglement". It is thus a relaxed form of disentanglement due to weaker bound of compactness. To the best of knowledge, we are the first to build quantitative disentanglement benchmark with contrastive learning involved and the first to build benchmark of disentanglement measurement on real-world datasets (CelebA). Recently, the study of contratsive learning, or generally self-supervised learning, is still motivated by empirical observations. We wish our work can reveal some clues on this line of study.

7 REPRODUCIBILITY

In this section, we provide the information required to reproduce our results reported in the main text. And we commit to make the code implementation and evaluated checkpoints public.

BYOL implementation For the implementation details of BYOL, please refer to Appendix A.1. The model architecture, training setups, and dataset preprocessing are all explained in detail.

VAE methods implementation For evaluation on synthetic datasets, i.e., dSprites, Cars3D, SmallNORB and Shapes3D, the disentanglement score is from the original logs of `disentanglement_lib` Locatello et al. (2019) [1]. 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 [2] and align the encoder architecture of all methods to be the same as described in Appendix A.1. Parameters are kept as the default well-tuned version in the provided implementation.

GAN methods implementation It is hard to insert 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. (2020). But the evaluation of other metrics in Lin et al. (2020) uses a not aligned settings with Locatello et al. (2019), so we check its officially release [3] to reevaluate the provided implementation and model weights under the unified evaluation setup. We perform the same evaluation process for results on CelebA dataset.

Evaluation Protocol Though our study is focused on disentanglement property, the evaluation of a representation model is two-step. In the first step, we perform factor prediction as a downstream task based on the learned representation model. Only when the accuracy of factor prediction is high enough to ensure the model weights "valid", we would go to the evaluation of disentanglement metric scores. For the implementation details of factor prediction, please refer to Appendix A.3. For disentanglement metrics evaluation, we use the official implementation of Locatello et al. (2019), i.e. `disentanglement_lib`. The settings of some important parameters in Appendix A.4.

[1] https://github.com/google-research/disentanglement_lib

[2] https://github.com/AntixK/PyTorch-VAE

[3] https://github.com/fjxmlzn/InfoGAN-CRhttps://github.com/fjxmlzn/InfoGAN-CR

REFERENCES

Alessandro Achille, Tom Eccles, Loic Matthey, Christopher P Burgess, Nick Watters, Alexander Lerchner, and Irina Higgins. Life-long disentangled representation learning with cross-domain latent homologies. *arXiv preprint arXiv:1808.06508*, 2018.

Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*, 2019.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6541–6549, 2017.

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

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

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

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020.

Ricky T. Q. Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems*, 2018.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pp. 2180–2188, 2016.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. *Advances in neural information processing systems*, 27:766–774, 2014.

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

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020.

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. *arXiv preprint*, 2016.

Insu Jeon, Wonkwang Lee, and Gunhee Kim. Ib-gan: Disentangled representation learning with information bottleneck gan. *arXiv preprint*, 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.

Tejas D Kulkarni, Will Whitney, Pushmeet Kohli, and Joshua B Tenenbaum. Deep convolutional inverse graphics network. *arXiv preprint arXiv:1503.03167*, 2015.

Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations. *arXiv preprint arXiv:1711.00848*, 2017.

Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and brain sciences*, 40, 2017.

Yann LeCun, Fu Jie Huang, and Léon 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, 2004a.

Yann LeCun, Fu Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pp. II–104. IEEE, 2004b.

Jason D Lee, Qi Lei, Nikunj Saunshi, and Jiacheng Zhuo. Predicting what you already know helps: Provable self-supervised learning. *arXiv preprint arXiv:2008.01064*, 2020a.

Wonkwang Lee, Donggyun Kim, Seunghoon Hong, and Honglak Lee. High-fidelity synthesis with disentangled representation. In *European Conference on Computer Vision*, pp. 157–174. Springer, 2020b.

Zinan Lin, Kiran Thekumparampil, Giulia Fanti, and Sewoong Oh. Infogan-cr and modelcentrality: Self-supervised model training and selection for disentangling gans. In *International Conference on Machine Learning*, pp. 6127–6139. PMLR, 2020.

Ralph Linsker. Self-organization in a perceptual network. *Computer*, 21(3):105–117, 1988.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, 15(2018):11, 2018.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *international conference on machine learning*, pp. 4114–4124. PMLR, 2019.

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

Judea Pearl. *Causality*. Cambridge university press, 2009.

Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017.

Senthil Purushwalkam and Abhinav Gupta. Demystifying contrastive self-supervised learning: Invariances, augmentations and dataset biases. *arXiv preprint arXiv:2007.13916*, 2020.

Siyuan Qiao, Huiyu Wang, Chenxi Liu, Wei Shen, and Alan Yuille. Weight standardization. *arXiv preprint arXiv:1903.10520*, 2019.

Scott E Reed, Yi Zhang, Yuting Zhang, and Honglak Lee. Deep visual analogy-making. *Advances in neural information processing systems*, 28:1252–1260, 2015.

Pierre H Richemond, Jean-Bastien Grill, Florent Altché, Corentin Tallec, Florian Strub, Andrew Brock, Samuel Smith, Soham De, Razvan Pascanu, Bilal Piot, et al. Byol works even without batch statistics. *arXiv preprint arXiv:2010.10241*, 2020.

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

Peter Spirtes, Clark N Glymour, Richard Scheines, and David Heckerman. *Causation, prediction, and search*. MIT press, 2000.

Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? *arXiv preprint arXiv:2005.10243*, 2020a.

Yuandong Tian, Lantao Yu, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning with dual deep networks. *arXiv preprint arXiv:2010.00578*, 2020b.

Christopher Tosh, Akshay Krishnamurthy, and Daniel Hsu. Contrastive learning, multi-view redundancy, and linear models. In *Algorithmic Learning Theory*, pp. 1179–1206. PMLR, 2021.

Yao-Hung Hubert Tsai, Yue Wu, Ruslan Salakhutdinov, and Louis-Philippe Morency. Demystifying self-supervised learning: An information-theoretical framework. *arXiv e-prints*, pp. arXiv–2006, 2020.

Michael Tschannen, Olivier Bachem, and Mario Lucic. Recent advances in autoencoder-based representation learning. *arXiv preprint arXiv:1812.05069*, 2018.

Dmitry Ulyanov, Andrea Vedaldi, and Victor 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*, pp. 6924–6932, 2017.

Sjoerd van Steenkiste, Francesco Locatello, Jürgen Schmidhuber, and Olivier Bachem. Are disentangled representations helpful for abstract visual reasoning? *arXiv preprint arXiv:1905.12506*, 2019.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pp. 9929–9939. PMLR, 2020.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–19, 2018.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3733–3742, 2018.

Nanxuan Zhao, Zhirong Wu, Rynson WH Lau, and Stephen Lin. What makes instance discrimination good for transfer learning? *arXiv preprint arXiv:2006.06606*, 2020.

Roland S Zimmermann, Yash Sharma, Steffen Schneider, Matthias Bethge, and Wieland Brendel. Contrastive learning inverts the data generating process. *arXiv preprint arXiv:2102.08850*, 2021.

A APPENDIX

A.1 IMPLEMENTATION OF BYOL MODEL

Architecture We describe the implementation details of BYOL in this part. To make a fair comparison with previous methods, for synthetic datasets, i.e. dSprites, Cars3D, SmallNORB, and Shapes3D, we follow the encoder architecture in Factor VAE (Kim & Mnih, 2018). The pipeline details are shown in Table 5. After each shown convolutional layer in the figure, there is a ReLU activation layer. By default, there is also a group normalization (group number = 4) layer. So, the encoder is a stack of (Conv-ReLU-GN) blocks. For CelebA dataset, the encoder is the commonly adopted ResNet-50 (He et al., 2016) backbone. By default, the final output channel number is 1000, i.e, $D = 1000$.

Besides the representation network (encoder), BYOL also has a projector network and a predictor network. Both of them consists 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.

Encoder of BYOL
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

Table 5: The encoder architecture for our implemented BYOL on synthetic datasets. By default, we set $D = 1000$ to be aligned with the commonly used ResNet-50 backbone network. 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.

Training settings We make minor modification on the training setting of BYOL. For training on all datasets, the images are resized to 64x64. For the data preprocessing, we copy 1-channel images of dSprites and SmallNORB to 3-channel. During 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. *transforms.RandomHorizontalFlip()*
3. *RandomApply(transforms.GaussianBlur((3,3), (1.0, 2.0)), p=0.2)*
4. *transforms.RandomResizeCrop(size=(64, 64), scale=(0.6,1.0))*
5. normalization

For dSprites and SmallNORB, image pixel value 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 evaluation. We select the weights after training for 140 epochs for evaluation on Cars3D and weights of the 200th epoch on SmallNORB considering the small scale of these two datasets.

As explained in some ablation experiments, the default training setting is not the one we could get the highest disentanglement score. For example, using Group Normalization + Weight Standardization achieves a higher disentanglement score than only using Group Normalization or not using normalization. But we aim to make minor modifications to the original BYOL implementation and follow previously commonly used setups.

To decrease the influence of randomness, we train each model configuration multiple times with different random seeds. We report the results by the average and three random seeds. To be precise, as our implementation is based on Pytorch, we would set initialize the libraries of *numpy*, *torch*, *torch.cuda* and *random* with the same random seeds. We use 0, 1, 2 as the three values of random seeds in trails.

A.2 Implementation of VAE models on CelebA

The benchmark (Locatello et al., 2019) provides VAE-based methods' disentanglement performance on synthetics datasets only, e.g., dSprites, Shapes3D, Cars3D, and SmallNORB. For the real-world CelebA dataset, we need to train the models by ourselves to gain the results in Table 3. We use an open-sourced implementation [4] for VanillaVAE, β-VAE, Factor-VAE, β-TCVAE and DIP-VAE-I. We follow the provided well-tuned hyper-parameters provided by them. Please refer to the repository for details.

Linear classifier for factor prediction
input: D-dim latent code
pipeline:
linear layer (1000 $\longrightarrow$ 256)
ReLU
linear layer (256 $\longrightarrow$ 128)
ReLU
linear layer (128 $\longrightarrow$ 100)
ReLU
K linear layers (in parallel, 100 $\longrightarrow n_k, 1 \leq k \leq K$)

Table 6: The implementation of linear classifier for factor prediction. With the D-dim latent code from the encoder, the classifier has multiple fully connected layers to shrink the feature vector to 100-dimensional. Then, given the number of factor types K for the target dataset, as each factor has $n_k(1 \leq k \leq K)$ values, we have K linear layers following the last shared layer in parallel. These layers predict the factor value on the corresponding K factors.

A.3 Implementation of Factor Prediction

As mentioned in the main paper, for each gained representation model, i.e., encoder, we will first confirm it "valid" by training a linear classifier for factor prediction. In the stage, the encoder weights will be frozen and only the attached classifier trained. For the task on all datasets, we use a unified classifier as shown in Table 6. The classifier has stacked fully connected layers (linear layers) and multiple in-parallel heads for factor prediction. On different target datasets, the number of factors and the possible values of factors varies. For the datasets we evaluate on, the value range and the accuracy threshold to confirm the quality of a representation acceptable are shown in Table 7. Usually, the accuracy threshold is set to be 0.8 or higher. But for the orientation factor on dSprites and Shapes3D, we relax the constraint to 0.7. The reason is that the factors are ill-defined to have ambiguity. For example, when a square or ellipse object in dSprites has an orientation of 180 degrees or 0 degrees, the image can be the same while the factors of orientation are regarded differently. For this reason, these factors bring noise into our study and validating the model weights. We, therefore, relax the accuracy requirement for them in practice.

[4] https://github.com/AntixK/PyTorch-VAE/

Dataset	Factor	Value Range	Accuracy Threshold
dSprites	Shape	3 (square, ellipse, heart)	0.95
	Scale	6	0.95
	Orientation	40	0.70
	Position X	32	0.80
	Position Y	32	0.80
Shapes3D	Floor hue	10	0.80
	Wall hue	10	0.80
	Object hue	10	0.80
	Scale	8	0.80
	Shape	4	0.80
	Orientation	15	0.70
Cars3D	Elevation	4	0.80
	Azimuth	24	0.80
	Object id	183	0.80
SmallNORB	Instacne category	10	0.90
	Elevation	9	0.80
	Azimuth	18	0.90
	Lighting condition	6	0.90
CelebA	40 human face factors	binary	0.80

Table 7: The factor of datasets we evaluate on. Some factors are originally continuous but discretized into all integers. Therefore, all factor prediction is classification task. Given a representation model trained on the training set of a dataset, the linear classifier should achieve accuracy higher than the theshold on all factors to be recognized "valid". Only a "valid" representation model would be put into the next step for disentanglement score evaluation. For the details of these factor definition, please refer to the original papers of these datasets.

A.4 EVALUATION METRICS

Five metrics are used to quantify the disentanglement of our model.

Beta Vae Metrics Introduced in Higgins et al. (2016), beta vae metrics assumes each dimension corresponds to one category in linear classifier. Representations are obtained after generated samples with only one factor k fixed. Calculating the summation of the divergence between different 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 beta vae metrics.

Factor Vae Metrics Kim & Mnih (2018) argues the beta vae metrics has the tendency to fail into a spurious disentanglement, and proposes a new metrics based on a linear classifier. Representations are obtained after 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 for the linear classifier. The accuracy of this linear model is the value of factor vae metrics.

Mutual Information Gap Chen et al. (2018) assumes 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 of highest and second-highest normalized mutual information between fixed k factor and dimensions in representations as shown below:

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

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

DCI disentanglement As Eastwood & Williams (2018) suggests, disentanglement is the entropy of the relative importance, evaluated with a Lasso or a Random Forest classifier, of each dimension for specific factors. Completeness is the entropy of the possibility that one factor is captured by a single code variables. Informativeness, overlapping with Disentanglement, is the prediction error of the model predicting factors.

SAP Kumar et al. (2017) 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 difference of the highest and second-highest score for each column.

For evaluation protocol, we follow the implementation provided by Locatello et al. (2019). Despite exception, 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 used discretizer function is the histogram discretizer and the number of bins in discretization is 20. For evaluation of MIG and SAP on dSprites, BYOL representation vectors are reduced to 10 dimension by PCA to be aligned with other methods. For evaluation of MIG and SAP on CelebA, to have a fair comparison, the representation vectors of all methods are reduced to 40 dimensions.

B MORE ABLATION EXPERIMENTS

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

In the original implementation of contrastive learning, the min-ratio of Random-Resize-Crop is usually very small, e.g, 0.08 for BYOL or 0.2 for MoCo/SimCLR. However, we find this hyper-parameter is critical. We thus perform a study with different minimum ratios of random-resize-crop. The results are shown in Table 8. From the table, we find the disentanglement score first increases when increasing the min-ratio of random-resize-crop but then drops. The default min-ratio is set to be 0.6 in our implementation.

Learning rate is also another important hyper-parameter, we use Adam optimizer with learning rate to be $3e-4$ as default. But we are still interested in the influence of learning rate to the disentanglement property of learned representation. At the same time, batch normalization in the encoder

min-scale	BetaVAE	FactorVAE
0.1	0.934	0.866
0.2	0.954	0.870
0.3	0.944	0.848
0.4	**0.960**	0.868
0.5	0.948	0.883
0.6	0.938	0.880
0.7	0.935	**0.890**
0.8	0.916	0.814
0.9	0.877	0.778

Table 8: The influence of minimum scale of random resize-crop augmentation on the disentanglement of BYOL learned features. Here, the batch size is set to be 64.

method	BetaVAE	FactorVAE	MIG	SAP
VAE	0.796	0.500	0.043	0.0031
β-VAE	0.203	0.245	**0.074**	0.0015
β-TCVAE	0.621	0.547	0.011	0.0029
Factor-VAE	0.632	0.540	0.002	**0.0083**
DIP-VAE-I	**0.820**	**0.673**	0.007	0.0024

Table 9: Disentanglement performance of VAE-based methods on dSprites, with an increased dimension of latent code to 1000-dimensional.

network is another critical component. In this part, we aim to study the transfer ability of learned representation, so we train BYOL under with different learning rate and batch normalization on three datasets respectively. But the evaluation is always performed on the test set of dSprites. The results are included in Table 10. It shows, no norm not just shows a higher disentanglement score than BN but is also more robust with the variance of learning rate. When the learning rate is high, e.g., 3e-3 or 3e-2, the model with BN has collapsed already while the model without norm can still output a reasonably high disentanglement score. On the other hand, on the transferability from one dataset to another, the disentanglement score also shows high agreement.

	dSprites		3dShapes		CelebA	
learning rate	w/ BN	w/o BN	w/ BN	w/o BN	w/ BN	w/o BN
3e-5	0.756	0.830	0.605	0.770	0.596	0.783
6e-4	0.832	0.920	0.643	0.724	0.673	0.811
1e-3	0.809	0.918	0.652	0.723	0.637	0.810
3e-3	0.423	0.914	0.231	0.710	0.651	0.805
3e-2	0.191	0.859	0.170	0.706	0.259	0.803

Table 10: We study the impact of learning rates over trained model's disentanglement. As all models are all tested on dSprites benchmark, BYOL shows its good generalization ability when trained on one dataset but transferring to other datasets.

As the supplement of study on the normalization strategy, we provide a more detailed study in Table 11 where five normalization strategies are evaluated with different batch sizes, which was found crucial in choosing normalization strategy in visual tasks. The result of the table still suggests

that Batch Normalization weakens BYOL's disentanglement property and no normalization, group normalization and layer normalization show good disentanglement with different batch sizes for training. When using instance normalization in the representation network, BYOL goes to collapse on dSprites.

batch size	w/o norm	BN	GN	LN	IN
64	0.885	0.800	**0.904**	0.892	0.191
128	0.896	0.833	**0.916**	0.910	0.191
256	0.920	0.856	0.932	**0.938**	0.191
512	0.896	0.850	**0.916**	0.914	0.191
1024	0.861	0.831	**0.880**	0.859	0.191
2048	**0.845**	0.798	0.825	0.828	0.191
4096	**0.824**	0.758	0.822	0.819	0.191

Table 11: Results of using different normalization strategy with different batch size during training on dSprites. We evaluate the FactorVAE score to indicate the disentanglement property of model weights from dSprites. For group normalization, we set group number to 4. BYOL collapses with instance normalization (IN) only and the evaluated disentanglement score from it also collapses to a constant here.

Richemond et al. (2020) shows that it is possible to achieve similar results when replacing BN with group norm and weight standardization (Qiao et al., 2019). We continue to show the influence of involving Group Normalization (GN) and weight standardization (WS) in Table 12. Here, we keep setting the group number to 4 and batch size to 512. And we use the two weight decay values in the original BYOL and Richemond et al. (2020) respectively (3e-8 and 1.5e-6). The results show that compared with solely using GN, adding additional Weight Standardization can not just help learn features of higher quality but also be more disentangled.

normalization	BetaVAE	FactorVAE	MIG	SAP
w/o norm	0.929	0.896	0.164	0.056
GN	0.932	0.916	0.294	**0.080**
GN + WS	0.966	0.916	0.315	0.074
GN + WS (wd = 3e-8)	**0.967**	**0.935**	0.313	0.076
GN + WS (wd = 1.5e-6)	0.962	0.191	**0.322**	0.075

Table 12: The results to study the influence of Group Normalization (GN) and Weight Standardization (WS) on the representation disentanglement. The results prove the effectiveness of both GN and WS to help promote representation disentanglement.