Unsupervised Object Representation Learning using Translation and Rotation Group Equivariant VAE

Anonymous Author(s)
Affiliation
Address
email

Abstract

In many imaging modalities, objects of interest can occur in a variety of locations and poses (i.e. are subject to translations and rotations in 2d or 3d), but the location and pose of an object does not change its semantics (i.e. the object's essence). That is, the specific location and rotation of an airplane in satellite imagery, or the 3d rotation of a chair in a natural image, or the rotation of a particle in a cryo-electron micrograph, do not change the intrinsic nature of those objects. Here, we consider the problem of learning semantic representations of objects that are invariant to pose and location in a fully unsupervised manner. We address shortcomings in previous approaches to this problem by introducing TARGET-VAE, a translation and rotation group-equivariant variational autoencoder framework. TARGET-VAE combines three core innovations: 1) a rotation and translation group-equivariant encoder architecture, 2) a structurally disentangled distribution over latent rotation, translation, and a rotation-translation-invariant semantic object representation, which are jointly inferred by the approximate inference network, and 3) a spatially equivariant generator network. In comprehensive experiments, we show that TARGET-VAE learns disentangled representations without supervision that significantly improves upon and avoids the pathologies of previous methods. Semantic representations learned by TARGET-VAE on images highly corrupted by rotation and translation, approach those learned on consistently posed objects, dramatically improving clustering and pose inference on multiple datasets.

1 Introduction

In many imaging modalities, objects of interest are arbitrarily located and oriented within the image frame. Examples include airplanes in satellite images, galaxies in astronomy images [1], and even particles in single-particle cryo-electron microscopy (cryo-EM) micrographs [2]. However, neither the location nor the rotation of an object within an image frame changes the nature (i.e. the semantics) of the object itself. An airplane is an airplane regardless of where it is in the image, and different rotations of a particle in a cryo-EM micrograph are still projections of the same protein. Hence, there is great interest in learning semantic representations of objects that are invariant to their locations and poses.

In general, unsupervised representation learning methods do not recover representations that disentangle the semantics of an object from its location or its pose. Popular unsupervised deep learning methods for images, such as variational autoencoders (VAE) [3], usually use encoder-decoder frameworks in which an encoder network is learned to map an input image to an unstructured latent variable or distribution over latent variables which is then decoded back to the input image by the decoder network. However, the unstructured nature of the latent variables means that they do not separate into any specific and interpretable sources of variation. To achieve disentanglement, unstructured methods

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

have been proposed that encourage independence between latent variables [4, 5]. However, these methods make no prior or structural assumptions about what these latent variables should encode, even when some sources of variation e.g. an object's location or pose are prevalent and well-known. Recently, several methods have proposed structured models that explicitly model rotation or translation within their generative networks [6, 7, 8] by formulating the image generative model as a function mapping coordinates in space to pixel values. Although promising, only the generative portion of these methods is equivariant to rotation and translation, and the inference networks have lackluster performance due to poor inductive bias for these structured latents.

To address this, we propose TARGET-VAE, a Translation and Rotation Group Equivariant Variational Auto-Encoder. TARGET-VAE is able to learn semantic object representations that are invariant to pose and location from images corrupted by these transformations by structurally decomposing the image generative factors into semantic, rotation, and translation components. We perform approximate inference with a group equivariant convolutional neural network [9] and specially formulated approximate posterior distribution that allows us to disentangle the latent variables into a rotationally equivariant latent rotation, translationally equivariant latent translation, and rotation and translation invariant semantic latent variables. By combining this with a spatial generator network, our framework is completely invariant to rotation and translation, unsupervised, and fully differentiable; the model is trained end-to-end using only observed images. In comprehensive experiments, we show that TARGET-VAE accurately infers the rotation and translation of objects without supervision and learns high quality object representations even when training images are heavily confounded by rotation and translation. We then show that this framework can be used to map continuous variation in 2D images of proteins collected with cryo-EM.

2 Related Work

In the recent years, there has been a great progress in developing models for unsupervised semantic analysis of images using VAEs [6, 10, 11, 12], flow-based methods [13], Generative Adversarial Networks (GAN) [14, 15, 16], and capsules [17]. In these methods, usually the goal is to build a low-dimensional representation of the images, which can be sampled to reconstruct similar images with some variations, or to use the representations to identify clusters in the data. The unsupervised learning methods used for clustering objects, need to address the variations caused by the common transformations in the objects. One simple approach to building invariant representations, is data augmentation. In this approach, training dataset plus samples generated by applying multiple transformations on them, are used to learn the transformation-invariant representations in a lower-dimensional latent space [18, 19]. This training approach is not efficient and more importantly, it does not guarantee invariance among the representations of the transformed versions of the same sample. Another group of methods that use data augmentation for learning transformation-invariant features, are based on contrastive learning [20, 21, 22]. These models learn to output similar encoding for the different transformed versions of the same data sample. Even though this training scheme has made great progress in the recent years, it is not an efficient training scheme for identifying invariant representations, since same as data augmentation, it needs to consider a large number of transformed samples to achieve invariance.

In some methods, the model is trained to learn an invariant representations by constraining the latent space [4, 5, 15]. In these methods, the latent space is not explicitly structured to separate the representation variable from the transformation ones. The latent space that can capture each targeted transformation in a structured space, can theoretically disentangle them from the content of the image. In [17], 3D point cloud representation of objects are decomposed into capsules, in a self-supervised learning scheme, and the model learns transformation-invariant descriptors along with the transformation-equivariant poses for the capsules, which leads to a more accurate reconstruction and clustering. A structured latent space is proposed in Spatial-VAE [6], where the latent space is divided to variables encoding the representation and transformations. However, only the generative model of Spatial-VAE is equivariant to rotation and translation, and the inference network does not perform any meaningful inference in these transformations.

To be able to disentangle the representation, from the transformations, the inference model should be equivariant to the transformations. Convolutional layers are one of the most popular feature extraction methods in image processing, due to their translation-equivariant features. However, they are not rotation-equivariant. There has been many studies in designing transformation-equivariant

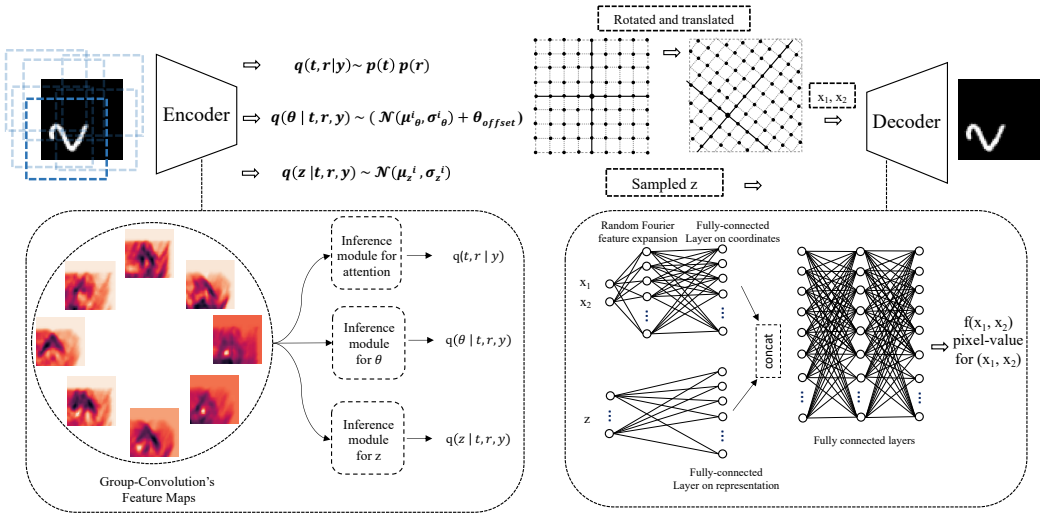

Figure 1: Overall framework, where the encoder uses group-equivariant layers to output mixture distributions over representation, rotation, and translation. The transformation-equivariant generator reconstructs the image based on the representation value z, and the transformed coordinates of the pixels.

and specially rotation-equivariant neural networks [9, 23, 24]. The Group-equivariant Convolutional Neural Networks (G-CNNs) [9], are one of these proposed methods where the 2D rotation space is discretized, and filters are applied over this rotation dimension, in addition to the spatial dimensions of the image. In this method, without adding anymore parameters and by adding small extra computational cost, the regular convolutional layers can reflect the rotation of the input images. Using G-CNNs in the inference model can add rotation-equivariance to the extracted features, which have not been accounted for in the previous unsupervised object detection methods.

Regarding the equivariance in the image generation process, a number of studies have proposed to train the generative model as a function that maps 2D or 3D spatial coordinates of pixels to their values [7, 8, 25, 26]. Due this mapping, any transformation in the pixel coordinates can transform the generated image. However, in these studies, no direct inference on the transformations is performed. To the best of our knowledge, we are the first method to establish rotation and translation equivariance in both the inference and the generative models of our proposed VAE framework, along with disentangling the mentioned transformations as structured variables in the latent space from the continuous translation-rotation-invariant representation variable of the objects.

3 Method

In this section, we present in details our proposed VAE framework that enables an unsupervised disentanglement of the rotation and translation of the objects from their representations. Figure 1 shows the overall framework.

3.1 Image generation process

Images can be considered as a combination of values for discreetly identifiable pixels. Transformation of an image, can also be considered as transforming the spatial coordinates of the pixels in the image. To have a generation process adherent to the transformations identified in the latent space, we define our generator as a function, which is mapping the spatial coordinates of the pixels to their values. The generator outputs a probability distribution as $p(\hat{y}_i|x_i, z)$, where z, x_i, and $\hat{y}_i$ are the latent representation variable, the spatial coordinates of a point, and the value for that point, respectively. Similar to [3], for an image with n pixels, we can define the probability of the image generated in

this manner as sum over probability of its pixel values:

$$log\ p(\hat{y}|z) = \sum_{i=1}^{n} log\ p(\hat{y}_i|z, x_i) \tag{1}$$

To define spatial coordinates over the pixels of the image, we use a Cartesian coordinates with the origin at the center of the image. The translation and rotation of the coordinates can be done by shifting the origin, and rotating coordinates around it, respectively. Depending on the value of the representation variable z, the generator acts as a function over the pixel coordinates to reconstruct the input image. In this setup the generation process is equivariant with regard to the transformations on the coordinate system. Assuming $R(\theta)$ as the rotation matrix for angle θ, and translation value of t, the probability of the generated image from Formula 1, can be rewritten as

$$log\ p(\hat{y}|z, \theta, t) = \sum_{i=1}^{n} log\ p(\hat{y}_i|z, R(\theta)x_i + t) \tag{2}$$

In this study, we focus on disentangling the rotation and translation from the representation of data, but the image generation probability from Formula 2, can be extended the other transformations too.

3.2 Approximate inference on latent variables

We implement a VAE framework to perform approximate inference on representation, rotation, and translation latent variables. Since rotation optimization is non-convex, we have discretized 2D rotation space to r values, each related to a specific $k\frac{2\pi}{r}$ angle, where $k \in \{0, 1, ..., r-1\}$. We represent the approximate posterior as $q(z, \theta, t, r|y)$, where z is the representation value, θ is the rotation angle, and t and r refer to the translation and discretized rotation values, and y is the input image. This posterior distribution is intractable, and making the simplifying assumption that q(z|t,r,y) and q(θ|t,r,y) are independent, we can factorize the posterior distribution as

$$q(z, \theta, t, r|y) = q(z|t, r, y)q(\theta|t, r, y)q(t, r|y) \tag{3}$$

, where $q(t, r|y)$ identifies a 3D distribution over t and r, and $q(\theta|t, r, y)$ and $q(z|t, r, y)$, are mixture distributions conditioned on r, t, and the input y.

We use $\mathcal{N}(0, I)$ for the prior distribution over z. The prior over r can be uniform or Gaussian, which is calculated based on the θ_{offset} of each rotation dimension. For example, when $r = 4$, we have rotation values of $\theta_{offset} \in \{0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}\}$, and assuming a uniform prior over r, means calculating probability of θ_{offset} for each rotation dimension. The prior over translation values t is set to be Gaussian with mean 0, which denotes the center of the 2D coordinate system. The standard deviation of this prior depends on the number of pixels in our fixed-size coordinate system and whether we want the model to be flexible with large translation values or not. Lower standard deviation for the prior over translations, will cause the model to pose higher loss towards larger predicted translation values. Using these priors, we calculate the Kullback-Leibler (KL) divergence for the latent variables:

$$KL(q(z, \theta, t, r|y)||p(z, \theta, t, r|y)) = \sum_{z,\theta,t,r} q(z, \theta, t, r|y) log \frac{q(z, \theta, t, r|y)}{p(z, \theta, t, r)} \tag{4}$$

To simplify this equation, we use the factorization from Equation 3, and based on calculations detailed in Appendix A1, we get:

$$KL(q(z, \theta, t, r|y)||p(z, \theta, t, r|y)) = KL_{t,r} + \sum_{t,r} q\,(t, r)\,(KL_\theta + KL_z) \tag{5}$$

, where $KL_{t,r} = \sum_{t,r} q(t, r|y) log \frac{q(t,r|y)}{p(t,r)}$, $KL_\theta = KL\,(q\,(\theta|t, r, y)\,||p\,(\theta|t, r, y))$, and $KL_z = KL\,(q\,(z|t, r, y)\,||p\,(z|t, r, y))$.

Multiplication of Kl_θ and KL_z with $q(t, r)$, weights the KL-divergence on θ and z based on the joint posterior distributions over t and r. We can calculate $p(t, r)$ by multiplying $p(t)$ and $p(r)$, since t and r are independent. The prior $p(z|t, r, y)$ can be considered equal to $p(z|y)$, which is $\mathcal{N}(0, I)$, and $p(\theta|t, r, y)$ can be simplified as $p(\theta|r)$ since θ is independent from t. We define this prior as $\mathcal{N}(\theta_{offset}, \frac{\pi}{r})$, which sets a Gaussian prior over rotation angles θ in each r dimension, with mean

equal to the rotation value of the filter in that r dimension and the standard deviation that is dependent on the number of r. We are tightening this prior by lowering their standard deviation as r grows, so that the distributions over the different r dimension would not overlap.

The full variational lower-bound for our model in this VAE framework is calculated based on the reconstruction loss and the Kl-divergence as:

$$\mathop{\mathbb{E}}_{(z,\theta,t)\sim\, q(z,\theta,t|y)}[log\ p(\hat{y}|z,\theta,t)] \quad - \quad KL(q(z,\theta,t,r|y)||p(z,\theta,t,r|y)) \tag{6}$$

, where we calculate the KL-divergence value in closed form using Equation 5 during the training.

3.3 Neural network architecture

Convolutional layers are translation equivariant meaning that the spatial shift in the image causes the same shift in their output feature maps, but convolutional layers are not rotation-equivariant. To incorporate rotation-equivariance in the inference model, we choose to use group convolutional layers from [9]. A symmetry group G is defined as a set X with an operation ., which is associative on the set, and has inverse and identity elements. Furthermore, combination of two symmetry transformations on a set, also creates a symmetry group. We can consider the regular convolutional layers as a function $f : \mathbb{Z}^2 \rightarrow \mathbb{R}^k$, that maps every pixel to a vector of k values where k is the number of kernels in the convolutional layer. The convolutional layer does this by sliding the kernels on the input image. Replacing this shifting operation with transformations from group G, creates a function over group G [27].

We define p_r group where $r \in \mathbb{N}$, and the operations consist of all translations, and r rotations, about any center of rotation in a 2D space. In the group convolutional layer, each kernel is rotated by $k\frac{2\pi}{r}$ angles, where $k \in \{0, 1, ..., r-1\}$ and then we calculate its dot product with the input image or feature map. Rotating kernels happens in addition to the translation of them across the input. As a result, the output feature map of the p_r group convolutions has one extra dimension which corresponds to the rotations of the kernel.

As shown in Figure 1, we define three parallel 1x1 3D convolutional layers on the translation and rotation equivariant feature maps of the group convolutional layers. These layers output three log posterior distributions of $log\ q(t, r|y)$, $log\ q(z|t, r, y)$, and $log\ q(\theta|t, r, y)$. To build the association between the θ values of each r dimension and the rotation of the kernel in that r dimension, we add the mean values from the $q(\theta|t, r, y)$ with the respective θ_{offset} values from their r dimension.

During the training, we sample from the joint posterior distribution of $log\ q(t, r|y)$ using Gumbel-Softmax [28], and use the soft one-hot output to calculate the weighted sum for the parameters that describe the approximate posterior distributions over content (z), and rotation (θ). We define a coordinate system where x is in [-1,1] range, and is centered on the middle of the image. The output of the Gumbel-Softmax directly describes a sample from the approximate posterior over the translation, which we get by calculating the weighted sum over the translation grid points. Then we translate and rotate this coordinate system using the sampled t and θ values. The generator receives the sampled z value along with the coordinates a of point at every pass, and outputs the value from that point in the image. We use random Fourier feature expansion [29] on the coordinates of the pixel before feeding it to the fully-connected layers. As shown in Figure 1, there are two sets of parallel fully-connected layers each with 512 hidden units, where one receives the Fourier feature expanded values of the coordinates, and the other one takes the z as input. The output of these layers are concatenated before being input to a final set of fully-connected layers to output the value for the pixel.

4 Results

4.1 Experiment setup

We run our experiments with p_4, p_8, and p_{16} group convolutional layers in the inference model. The input images are normalized to the range of [0,1]. We use 128 rectangular filters where size of the kernels in the first layer is set to a value larger than the size of the targeted objects. We do this because we use only one group convolutional layer in our models and we want to make sure that our receptive field is at least as large as the objects of interest. We use leaky-ReLU activation functions and the

batch size for all the experiments is set to 100. We use ADAM optimizer [30], with learning rate of 2e-4, and the learning rate is reduced by factor of 0.5 after no improvements in the loss for 10 epochs. We run the training for a maximum of 500 epochs with early-stopping in case of no improvements in the loss for 20 epochs. We run all our experiments in a single NVIDIA A100 gpu, with 80 GB memory.

The fully-connected layers in the generator, each has 512 hidden units, and the output can be one probability value in the case of binary valued pixels, or mean and standard deviation if the pixels are real values. For the RGB images, the dimension of output for each pixel can be easily extended to three.

4.2 TARGET-VAE predicts the translation and rotation values of objects with high correlation

We evaluate how accurately, our proposed model can identify the translation and rotation values of the objects in the images. For this purpose, we created two datasets using hand-written digits in MNIST. In the first one, the digits are randomly rotated by sampling from $\mathcal{N}(0, \frac{\pi^2}{16})$ and randomly translated by sampling from $\mathcal{N}(0, 5^2)$ pixels. We call this dataset MNIST(N), because of the normal distribution over the rotation angles. In the second dataset, which we call MNIST(U), the digits are translated similar to the MNIST(N), but the rotations are randomly sampled from the uniform distribution of $U(0, 2\pi)$. In both datasets, the train and test sets have 60,000, and 10,000 images, where each image is 50x50 pixels. Appendix A2 shows some examples of the data and the rotation distribution in these datasets.

We train TARGET-VAE with p_4, p_8, and p_{16} on these generated datasets. For the prior over θ, we use $\mathcal{N}(0, \frac{\pi}{4})$ for the MNIST(N) dataset, and $U(0, 2\pi)$ for the MNIST(U). For $p(\theta|r)$ we calculate $p(\theta_{offset(r)}) \sim p(\theta)$, where $\theta_{offset(r)}$ is the rotation angle of the kernels in each dimension of r. In the training for both of the datasets, we set the prior over t to $\mathcal{N}(0, 5^2)$.

After the training, to evaluate our model's performance on the prediction of the translation and rotation of the objects, we use samples from the test set as the input to the inference model to obtain $q(t, r|y)$ and $q(\theta|t, r, y)$. To identify the most-likely (t, r), we apply the arg-max function on $q(t, r|y)$, and we use them to get θ from $q(\theta|t, r, y)$. For the translation values, we calculate Pearson correlation, but because of the circular nature of the rotations values, we calculate the circular correlation [31] for them. Table 1, shows the correlation values on the test set for MNIST(N) and MNIST(U). The values in these tables are based on the model trained with dimension of the representation space (z_dim) set to 2. To the best of our knowledge, the only other method that performs direct inference on the rotation values to identify the invariant representations for the objects, is spatial-VAE [6]. To compare our results, we run spatial-VAE with the same number of parameters and z_dim=2. As shown in Table 1, in both MNIST(N) and MNIST(U) datasets, TARGET-VAE significantly improves the rotation correlation value. Our experiments show that increasing dimension of the representation space, does not have a significant effect on the translation and rotation correlation values, but increasing the level of the discretization over the rotation dimension usually improves the rotation correlation. This is confirmed by TARGET-VAE with p_{16} achieving the highest rotation correlation value. In the translation column, the two values are showing the translation correlation for x_1 and x_2 coordinates. We believe the reason for the slightly higher translation correlation of spatial-VAE than our proposed models might be in the fact that TARGET-VAE can predict the translations only in a discrete pixel level manner, and this might affect its ability in identifying the translations that have sub-pixel values.

Table 1: Translation and rotation correlation on MNIST(N) and MNIST(U) with z_dim=2

	MNIST(N)		MNIST(U)	
Group Convolution	Translation	Rotation	Translation	Rotation
Spatial-VAE [6]	**0.994, 0.993**	0.564	**0.982, 0.983**	0.005
TARGET-VAE p_4	0.97, 0.972	0.814	0.975, 0.976	0.80
TARGET-VAE p_8	0.981, 0.981	0.89	0.972, 0.971	0.859
TARGET-VAE p_{16}	0.969, 0.972	**0.898**	0.974, 0.971	**0.93**

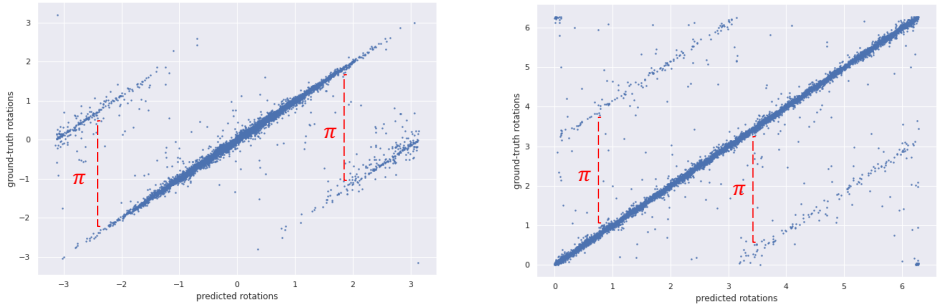

Figure 2: Left: Predicted θ vs. ground-truth data in MNIST(N), Right: Predicted θ vs. ground-truth data in MNIST(N), with TARGET-VAE p_8. There is a π difference between the predicted angles and the ground-truth values of some of the digits, which is related to the rotation symmetry of them.

Some of the digits in MNIST datasets, have natural ambiguity with regard to their rotations. Investigating the correlation between the predicted rotation values and the ground-truth ones, shows a difference about π between the some of the predicted angles and their ground-truth ones. Figure 2 shows this difference. By drawing digit-wise rotation correlation figures in appendix A.3, we show that this π difference mostly belongs to the three digits of 0, 1, and 8. This shows how our model can acknowledge the rotation symmetry in these digits.

4.3 TARGET-VAE improves clustering accuracy by learning translation and rotation invariant features

By disentangling rotation and translation from the representation values, TARGET-VAE can identify the different clusters of the objects. Similar to 4.2, we use arg-max$(q(t, r))$ to identify the most-probable representation from $q(z|t, r, y)$. We train TARGET-VAE with p_4, p_8, and p_{16} and evaluate their clustering performance using agglomerative clustering [32] on MNIST(N) and MNIST(U). Table 2 shows the clustering accuracies on these two datasets, where increasing the dimension of the representation space, helps the model with the separation of different digits and achieving higher clustering accuracy. Based on the accuracies in this table and rotation correlations in Table 1, higher discretization of the rotation space, allows the TARGET-VAE to have a better estimate of the rotation of the objects, and this is turn results in more rotation-invariant representation encoding and higher clustering accuracy. To compare the clustering performance of TARGET-VAE with other similar methods, we train a standard VAE [3], beta-VAE [4], and spatial-VAE [6] with about same number of parameters as TARGET-VAE, on MNIST(N) and MNIST(U) datasets. We set the z_dim=2 for both spatial-VAE and TARGET-VAE. Since the standard VAE and the beta-VAE do not have dedicated latent variables for identifying translation and rotation, we set the dimension of latent space to 4 for them, so that all of the models would have equal latent space dimensions. Also, for the beta-VAE, we set β value to 4 based on the proposed settings in [4]. As shown in the Table 2, all variations of the TARGET-VAE outperform the other methods. We believe that the direct inference on the rotation and translation, and dedicating structured latent variables to encode these transformations in TARGET-VAE, plus the transformation-equivariant generator, are the main reason for the dramatic increase in the clustering accuracy with our proposed method. In MNIST(N), since there are limited range for the rotation values, increasing the size of the group convolution, does not necessarily increase the rotation correlation or the clustering accuracy, where in MNIST(U) due to the larger variations in the rotation values, increasing the size of the group convolution leads to a better estimate of rotations and improved clustering accuracy. Figure 3 shows the confusion matrix and t-sne plot for TARGET-VAE with p_8 group-convolution, on MNIST(N).

In appendix A.4, we perform experiments using images with multiple digits to show that TARGET-VAE can detect multiple objects in the evaluation stage. Furthermore, in appendix A.5 and A.6, we train and evaluate TARGET-VAE on the dSprites [33], and galaxy zoo [1]. We also perform an ablation study on the important components of our proposed model in Appendix A.7.

Table 2: Clustering accuracy (%) on MNIST(N) and MNIST(U)

Model	MNIST(N)	MNIST(U)
VAE (z_dim=4) [3]	15.3	12.8
Beta-VAE (z_dim=4) [4]	15.1	18.0
Spatial-VAE (z_dim=2) [6]	37.1	28.2
TARGET-VAE p_4 (z_dim=2)	56.4	56.6
TARGET-VAE p_8 (z_dim=2)	60.1	57.1
TARGET-VAE p_{16} (z_dim=2)	60.1	63.4
TARGET-VAE p_4 (z_dim=32)	65.1	64.3
TARGET-VAE p_8 (z_dim=32)	**77.7**	69.1
TARGET-VAE p_{16} (z_dim=32)	75.2	**71.2**

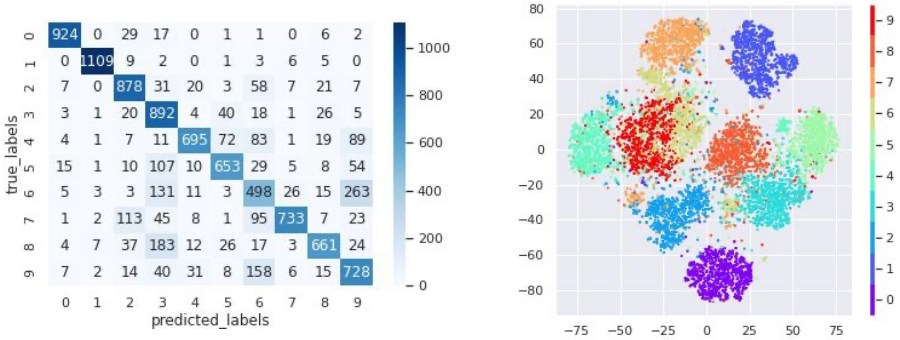

Figure 3: Confusion matrix of trained TARGET-VAE on MNIST(N) with z_dim=32 (left), and the t-sne figure of the 32 dimension latent representation space (right). Based on the confusion matrix, most of the digits are clustered correctly, and some misclassification is related to the confusion between rotated digits 6 and 9.

4.4 Identifying class-averages and protein conformations in the electron-microscopy images with TARGET-VAE

In the single particle electron microscopy images, biological particles can appear with different rotations and in different locations of the images. To be able to identify different types of particles or different conformations of the protein, the representation vector should be invariant to these transformations. Here, we demonstrate the capacity of TARGET-VAE in capturing different classes and variable conformations of particles in the electron microscopy images.

Identifying different classes of particles in EMPIAR-10025. The dataset EMPIAR-10025 [34] contains T20S Proteasome micrographs with 2.8 Å resolution. We use Topaz [35] package to create 161292 images of 400x400 pixels, where each image contains a particle. After downsampling by factor of 4 and normalizing the images, we train TARGET-VAE with p_8 and z_dim=2 on them. Due to the high-complexity of the structures of the proteins, we use 6 fully-connected layers with 512 units in the generator. In the rotation inference our prior over r is uniform. We randomly select 10% of images as validation set to monitor the training process. As shown in Figure 4, our model identifies three clusters in this dataset. The image on the left side of the Figure 4 contains instances of input images which are extremely noisy.

Identifying different views of particles in EMPIAR-10029. The EMPIAR-10029 dataset contains simulated EM dataset of GroEL particles. This dataset has 10000 images of shape 200x200, which we downsample to 100x100 and normalize. Similar to EMPIAR-10025, we train TARGET-VAE with the same settings as the EMPIAR-10025. The latent space does not identify separate clusters in this dataset, but performing a random sampling in the latent space helps us with identifying different projections of the biological particle (Fig. 5). We have also tested our model in identifying the heterogeneity in some other cryo-EM datasets, which we discuss in appendix A.8.

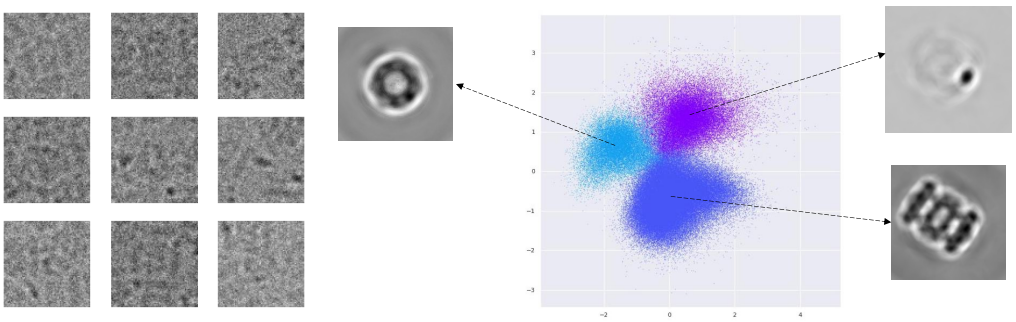

Figure 4: Left: Images of T20S Proteasome particles from EMPIAR-10025 dataset. Right: The 2D latent representation space for EMPIAR-10025 dataset learnt by TARGET-VAE, which identifies three clusters.

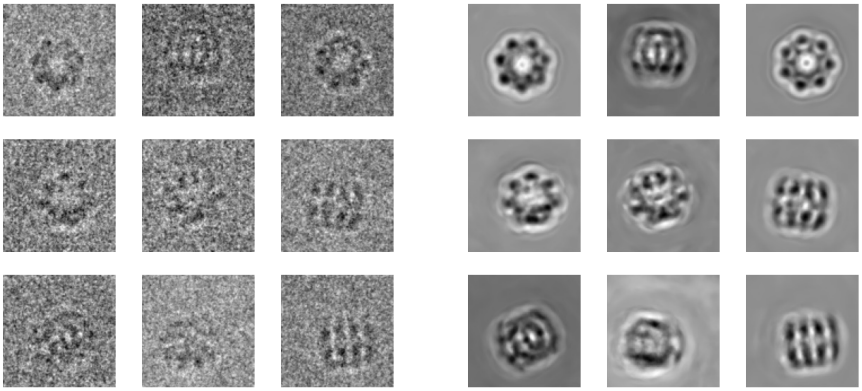

Figure 5: Left: Images of simulated EM dataset of GroEL particles from EMPIAR-10029 dataset. Right: The reconstructed particles with rotation angle set to zero.

5 Conclusion

In this paper, we presented TARGET-VAE, a translation and rotation group equivariant variational autoencoder framework for learning translation and rotation invariant object representation in images without supervision. By structuring the encoder to be translation and rotation equivariant, the model can identify the latent rotation and translation variables and disentangle them from the latent variables encoding the content of the image. The spatially equivariant generator samples from the posterior distribution to reconstruct the image. We showed that this approach not only accurately estimates the rotation and translation of objects, but also learns content representations that reflect known semantics (i.e., clusters match known semantic labels) across multiple datasets. Although we only consider 2d images containing single objects with rotation and translation, this framework can be extended to other object transformations by adopting encoder networks with the proper equivariance and adapting the latent distribution. In the future, we expect this framework can also be extended to multi-object detection, object tracking over time, and to 3d environments. TARGET-VAE lays the groundwork for a new generation of fully unsupervised object detection and semantic analysis methods for cryo-EM and other imaging modalities.

References

[1] Chris J Lintott, Kevin Schawinski, Anže Slosar, Kate Land, Steven Bamford, Daniel Thomas, M Jordan Raddick, Robert C Nichol, Alex Szalay, Dan Andreescu, et al. Galaxy zoo: mor-

phologies derived from visual inspection of galaxies from the sloan digital sky survey. *Monthly Notices of the Royal Astronomical Society*, 389(3):1179–1189, 2008.

[2] Fred J Sigworth. Principles of cryo-em single-particle image processing. *Microscopy*, 65(1):57–67, 2016.

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

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

[5] Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β-VAE. *arXiv preprint arXiv:1804.03599*, 2018.

[6] Tristan Bepler, Ellen Zhong, Kotaro Kelley, Edward Brignole, and Bonnie Berger. Explicitly disentangling image content from translation and rotation with spatial-VAE. *Advances in Neural Information Processing Systems*, 32, 2019.

[7] Ellen D Zhong, Tristan Bepler, Bonnie Berger, and Joseph H Davis. Cryodrgn: reconstruction of heterogeneous cryo-em structures using neural networks. *Nature methods*, 18(2):176–185, 2021.

[8] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer, 2020.

[9] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999. PMLR, 2016.

[10] SM Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Geoffrey E Hinton, et al. Attend, infer, repeat: Fast scene understanding with generative models. *Advances in Neural Information Processing Systems*, 29, 2016.

[11] Ishaan Gulrajani, Kundan Kumar, Faruk Ahmed, Adrien Ali Taiga, Francesco Visin, David Vazquez, and Aaron Courville. Pixelvae: A latent variable model for natural images. *arXiv preprint arXiv:1611.05013*, 2016.

[12] Eric Crawford and Joelle Pineau. Spatially invariant unsupervised object detection with convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3412–3420, 2019.

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

[14] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[15] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *Advances in neural information processing systems*, 29, 2016.

[16] Ivan Skorokhodov, Savva Ignatyev, and Mohamed Elhoseiny. Adversarial generation of continuous images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10753–10764, 2021.

[17] Weiwei Sun, Andrea Tagliasacchi, Boyang Deng, Sara Sabour, Soroosh Yazdani, Geoffrey E Hinton, and Kwang Moo Yi. Canonical capsules: Self-supervised capsules in canonical pose. *Advances in Neural Information Processing Systems*, 34:24993–25005, 2021.

[18] Luke Taylor and Geoff Nitschke. Improving deep learning with generic data augmentation. In *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1542–1547. IEEE, 2018.

[19] Chao Chen, Guanbin Li, Ruijia Xu, Tianshui Chen, Meng Wang, and Liang Lin. Clusternet: Deep hierarchical cluster network with rigorously rotation-invariant representation for point cloud analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4994–5002, 2019.

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

[21] 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*, pages 9729–9738, 2020.

[22] Enze Xie, Jian Ding, Wenhai Wang, Xiaohang Zhan, Hang Xu, Peize Sun, Zhenguo Li, and Ping Luo. Detco: Unsupervised contrastive learning for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8392–8401, 2021.

[23] Jan Eric Lenssen, Matthias Fey, and Pascal Libuschewski. Group equivariant capsule networks. *Advances in Neural Information Processing Systems*, 31, 2018.

[24] Taco S Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical cnns. *arXiv preprint arXiv:1801.10130*, 2018.

[25] Kenneth O Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic programming and evolvable machines*, 8(2):131–162, 2007.

[26] Paul Andrei Bricman and Radu Tudor Ionescu. Coconet: A deep neural network for mapping pixel coordinates to color values. In *International Conference on Neural Information Processing*, pages 64–76. Springer, 2018.

[27] Erik J Bekkers. B-spline cnns on lie groups. In *International Conference on Learning Representations*, 2019.

[28] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[29] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

[30] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[31] S Rao Jammalamadaka and Ambar Sengupta. *Topics in circular statistics*, volume 5. world scientific, 2001.

[32] Fionn Murtagh and Pierre Legendre. Ward's hierarchical agglomerative clustering method: which algorithms implement ward's criterion? *Journal of classification*, 31(3):274–295, 2014.

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

[34] Melody G Campbell, David Veesler, Anchi Cheng, Clinton S Potter, and Bridget Carragher. 2.8 å resolution reconstruction of the thermoplasma acidophilum 20s proteasome using cryo-electron microscopy. *Elife*, 4:e06380, 2015.

[35] Tristan Bepler, Andrew Morin, Micah Rapp, Julia Brasch, Lawrence Shapiro, Alex J Noble, and Bonnie Berger. Positive-unlabeled convolutional neural networks for particle picking in cryo-electron micrographs. *Nature methods*, 16(11):1153–1160, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] Our proposed framework can perform inference on the rotation and translation values of the object and use this inference to learn a rotation and translation invariant representation value for them.
   (b) Did you describe the limitations of your work? [Yes] We mentioned the limitations in the conclusion in the form the future directions.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] To the best of our knowledge, our work does not have any negative societal impacts.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] We have read the ethics review guidelines and we believe that our paper conforms to that.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] In the parts that we made any assumptions, we clearly explained them.
   (b) Did you include complete proofs of all theoretical results? [Yes] We included proof for one of our equations in section 3.2 which needs proof.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code is available at ——————————
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] All the details are either explained in the paper or can be found in the code.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] The variation in the results is not large enough to justify using error bars.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We included all the processing information.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] All the data that we are using is the public domain.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] The data that we are using does not contain any personally identifiable information or offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Appendix

A.1 Calculating Kullback-Leibler divergence

Based on the standard definition for the KL-divergence, we have:

$$KL(q(z,\theta,t,r|y)||p(z,\theta,t,r|y)) = \sum_{z,\theta,t,r} q(z,\theta,t,r|y) log \frac{q(z,\theta,t,r|y)}{p(z,\theta,t,r|y)} \tag{7}$$

$$= \sum_{z,\theta,t,r} q(t,r|y)q(\theta|t,r,y)q(z|t,r,y) log \frac{q(t,r|y)q(\theta|t,r,y)q(z|t,r,y)}{p(t,r|y)p(\theta|t,r,y)p(z|t,r,y)}$$

$$= \sum_{t,r} q(t,r|y) log \frac{q(t,r|y)}{p(t,r|y)} +$$

$$\sum_{z,\theta,t,r} q(t,r|y)q(\theta|t,r,y)q(z|t,r,y) log \frac{q(\theta|t,r,y)q(z|t,r,y)}{p(\theta|t,r,y)p(z|t,r,y)}$$

To simplify this equation, we define the first part of the result from Equation 7 as:

$$KL_{t,r} = \sum_{t,r} q(t,r|y) log \frac{q(t,r|y)}{p(t,r|y)} \tag{8}$$

The second part of the result from Equation 7 can be further expanded as:

$$\sum_{z,\theta,t,r} q(t,r|y)q(\theta|t,r,y)q(z|t,r,y) log \frac{q(\theta|t,r,y)q(z|t,r,y)}{p(\theta|t,r,y)p(z|t,r,y)} \tag{9}$$

$$= \sum_{t,r} q\left(t,r\right) \left(\sum_{\theta} q\left(\theta|t,r,y\right) log \frac{q\left(\theta|t,r,y\right)}{p\left(\theta|t,r,y\right)} + \sum_{z} q(z|t,r,y) log \frac{q(z|t,r,y}{p(z|t,r,y} \right)$$

$$= \sum_{t,r} q\left(t,r\right) \left(KL\left(q\left(\theta|t,r,y\right)||p\left(\theta|t,r,y\right)\right) + KL\left(q\left(z|t,r,y\right)||p\left(z|t,r,y\right)\right)\right)$$

Assuming these definitions for KL_θ and KL_z:

$$KL_\theta = KL\left(q\left(\theta|t,r,y\right)||p\left(\theta|t,r,y\right)\right) \tag{10}$$

$$KL_z = KL\left(q\left(z|t,r,y\right)||p\left(z|t,r,y\right)\right) \tag{11}$$

, we can rewrite the the Equation 7 using Equations 8, 9, 10, and 11 as:

$$KL(q(z,\theta,t,r|y)||p(z,\theta,t,r|y)) = KL_{t,r} + \sum_{t,r} q\left(t,r\right)\left(KL_\theta + KL_z\right) \tag{12}$$

A.2 MNIST(N) and MNIST(U) datasets

We generated two datasets, MNIST(N) and MNIST(U), by rotating and translating digits in MNIST. The rotations of digits in MNIST(N) are randomly sampled from $\mathcal{N}(0, \frac{\pi^2}{16})$, and in MNIST(U) are randomly sampled from $U(0, 2\pi)$ (Figure 6).

A.3 Digit-wise rotation correlation, and RMSE of the predicted rotations

Following our discussion about the difference between the predicted angle and the ground-truth values, we measure the digit-wise rotation correlation for each dataset (Figure 7). We find that the errors in the predicted rotations are concentrated around π for digits 0, 1, and 8. We suspect that this is because these digits have rough 180 degrees symmetry.

To study the accuracy of the predicted rotation angles by TARGET-VAE, we calculate the mean standard deviation of the predicted rotations, introduced in [17]. This metric basically measures the mean square error between the rotation of the object in the input image and the predicted rotation for that object. We rotate each image in the test set of MNIST(U), 160 times using angles uniformly picked from [0, 2π], and then pass them through the encoder to get the predicted rotation angle for them. Since assigning predicted rotation of zero to an object is arbitrary in our framework (model might assign predicted rotation angle of zero to the object that is actually rotated 90 degrees), we

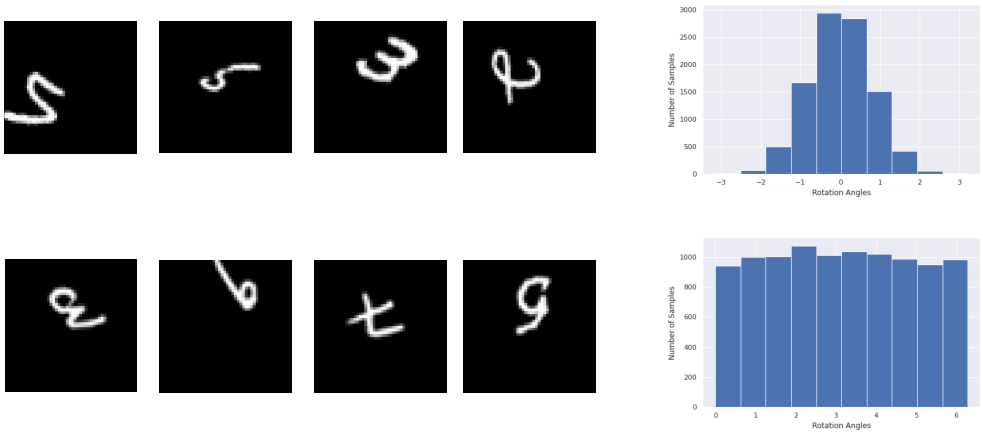

Figure 6: Left: Instances of MNIST(N) (top), and MNIST(U) (bottom) datasets. Right: Distribution of rotation angles in the test set of MNIST(N) (top), and MNIST(U) (bottom) datasets.

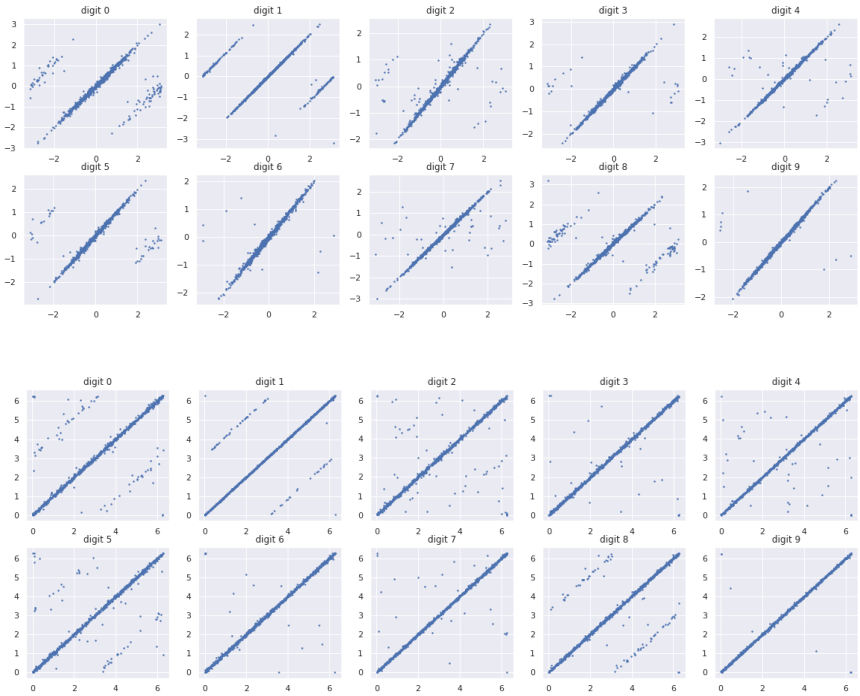

Figure 7: The predicted rotation values for each digit and their ground-truth rotation angles for digits in MNIST(N) (top), and MNIST(U) (bottom) using TARGET-VAE with model $p8$ and z_dim=2.

subtract the predicted rotation for each one of 160 rotated objects from the predicted rotation value when the input object is not rotated. Table 3, shows the root mean squared error (RMSE) calculated in average for each digit's predicted rotation. As expected, the RMSE of rotations is highest when there is no inference done on rotations as in Spatial-VAE [6], and using finer discretization in rotation inference results in more accurate prediction of the rotation values. Some digits such as 0, 1, and 8, due to their symmetry, have less accurate rotation predictions compared to the other ones.

Table 3: The root mean squared error of the predicted rotations over MNIST(U) with z-dim=2

Digits	0	1	2	3	4	5	6	7	8.	9	AVG
Spatial-VAE [6]	96.50	98.73	97.83	97.88	98.59	98.06	97.82	98.10	97.62	98.55	97.97
P4	31.57	33.86	32.36	11.66	27.08	20.94	6.26	8.15	28.95	6.68	20.75
P8	32.81	22.04	16.69	6.36	20.71	16.24	13.00	15.84	16.97	7.80	16.85
P16	17.39	18.47	14.74	5.79	7.65	12.09	3.40	4.63	17.75	2.74	10.47

A.4 TARGET-VAE identifies multiple objects without supervision

We create a new dataset using multiple rotated and translated digits from MNIST(U). We select images at random from MNIST(U) and insert them in random locations of a 150x150 pixel frame. We call this new dataset MNIST(multi). We use the model trained on MNIST(U) to identify the location, rotation, and the content latent for images in MNIST(multi). We find that the model correctly identifies and reconstructs the objects (Figure 8). Even though the model, can identify non-overlapping objects in this approach, it struggles to identify the overlapping objects. We believe these preliminary results show the potential of our model to be trained on images with multiple objects and to identify the objects independent of their location and rotation.

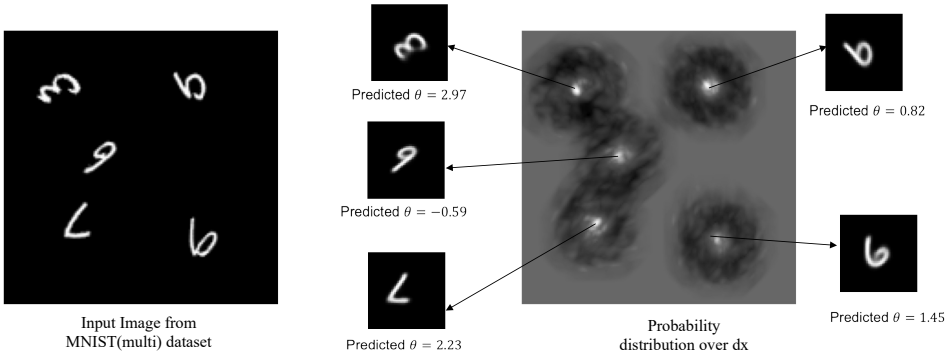

Figure 8: Left: Input image with multiple rotated and translated digits from MNIST(multi); Right: Probability distribution over t, the reconstructed objects and their predicted rotation angles. We marginalized $q(t, r|y)$ over r, for visualization purposes. The high probability values in this attention map show the predicted locations of the objects. We use the peaks in $q(t, r|y)$ to sample from $q(\theta|t, r, y)$ and $q(z|t, r, y)$, the predicted rotation and content values for each object. The sampled θ, t, and z values are used to reconstruct each individual object.

A.5 TARGET-VAE predicts the rotation of dSprites shapes and clusters them with high accuracy

The dSprites dataset was introduced to benchmark unsupervised disentangled representation learning methods. It contains 64x64 images of three shapes: squares, ellipses, and hearts. Each shape is rotated by one of 40 values in [0,2π], translated across both x and y dimensions, and scaled using one of six linearly spaced values in [0.5,1]. We train TARGET-VAE with p_8 group convolution and z_dim=2 on dSprites. We use a uniform prior over r and $\mathcal{N}(0, \pi)$ prior over θ. TARGET-VAE learns content representations that capture the type and scale of objects, but not their location and rotation (9). Furthermore, TARGET-VAE accurately predicts the rotation of the shapes, and, interestingly, the symmetry groups of the shapes become immediately apparent when examining the correlation between the predicted and ground truth rotations.

A.6 TARGET-VAE learns transformation-invariant representations in the galaxy dataset

Galaxy zoo contains images of galaxies gathered by Sloan Digital Sky Survey. This dataset contains more than 61000 RGB images, which we cropped and downsampled to 64x64 and split into train and test sets. The galaxies appear with different rotations and translations in the images. We train

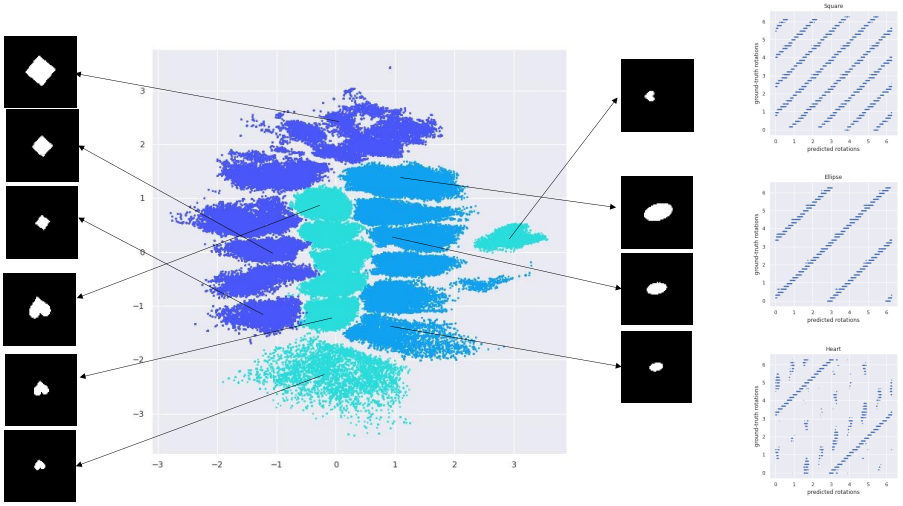

Figure 9: Left: Two-dimension latent space of dSprites using TARGET-VAE with p_8. Right: Correlation between rotations given by TARGET-VAE and ground truth rotations for squares, ellipses, and hearts.

TARGET-VAE with p_8 group convolution on the train set. TARGET-VAE learns to accurately predict the rotation and translation of the galaxies and reconstructs centered and aligned galaxy images when decoding with t and θ set to zero (Figure 10).

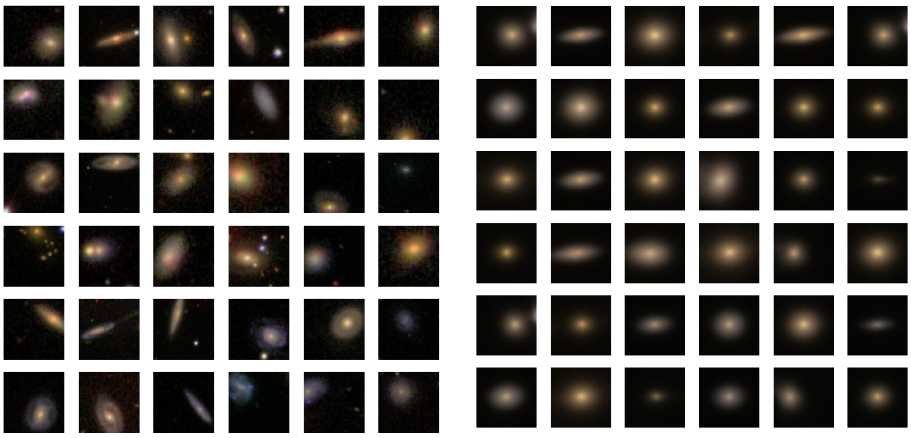

Figure 10: Left: Images of galaxy dataset used for testing. Right: Reconstructed images by TARGET-VAE with p_8 and z_dim=2, where rotation and translation are set to 0, by using the transformation-invariant representations.

A.7 Ablation studies

We conducted extensive ablation study to validate the effectiveness of the main components of our proposed method.

Variant 1 - Inference only on translations We evaluated the importance of the rotation equivariant features, by modifying the inference model to use regular convolutional layers instead of the group convolutional ones. As a result, the posterior distributions will be q(t|y), q(z|t,y), and q(θ|t,y), where they only depend on the input y and the translation value t. We observed that, as expected,

Table 4: Performace of variants in the ablation study on MNIST(U)

Model	Group Conv	Translation Corr	Rotation Corr	Clustering Accuracy
Variant 1	-	0.966, 0.967	0.005	33.8%
Variant 2	p4	0.967, 0.967	0.005	37.1%
Variant 3	p4	0.968, 0.972	0.008	36.6%
Variant 4	p4	0.975, 0.976	0.80	56.6%
Variant 5	p8	0.972, 0.971	0.859	57.1%
Variant 6	p16	0.974, 0.971	0.93	63.4%

eliminating inference on the discretized rotation dimension has a negative effect on identifying transformation-invariant representations (Table 4).

Variant 2 - Inference only on translations + using group convolutional layers In this variant, we use the group convolutional layers in the inference model, but we do not perform any inference on the rotation dimension. We apply a fully-connected layer on the output of the group convolutional layers to map the rotation values at every location, to a single value. As a result the posterior distribution are the same as variant 1. We experiment with this variant to identify how effective is the use of the group convolution layers. It turns out that if we do not perform any inference on the rotation dimension, just using group convolutional layers for feature extraction only slightly improves the clustering accuracy.

Variant 3 - Inference on both translations and rotations without adding θ_{offset}: In this variant, we use group convolutional layers and we perform inference on both translations and rotations. The difference between this variant and our proposed framework is that we are not adding θ_{offset} of kernels in each rotation dimension to the q(θ|t,r,y). Adding θ_{offset} to the posterior on θ, allows us to break down the rotation space among the r rotation dimension, and without it, the model proves not capable of identifying the rotations of the digits in the MNIST(U) dataset.

Variant 4 to 6 - Increasing the level of discretization of the rotation space In these variants, we perform inference on rotation and translations and we add θ_{offset} of each rotation dimension to its corresponding q(θ|t,r,y) distribution. In these variants, we show that by increasing the level of discretization of the rotation space, model can have a better estimate of the actual rotation values, and this in turn helps the model with improving the clustering accuracy.

Table 4, shows the translation, rotation correlation along with the clustering accuracy of the mention variants above, on the MNIST(U) dataset. Performing inference on rotations plus adding θ_{offset} to the posterior on θ, significantly increases the rotation correlation and the clustering accuracy.

A.8 Learning transformation-invariant representations of proteins with TARGET-VAE

In cryo-EM images, rotation and translation are the major transformations that cause variations in particles in the image. Here, we show the result of our experiments with TARGET-VAE to learn transformation-invariant representations, for some more cryo-EM datasets.

Identifying hinge motion of 5HDB We train our model on a dataset of 20000 simulated projections of integrin α-IIb in complex with integrin β-3 (5HDB). We aim to identify the rotation invariant representations of the protein to be able to identify the variations in its structure. We train TARGET-VAE with p_8 and z_dim=1 and uniform prior over θ. Figure 11 shows some examples of the input images and their reconstructed versions with translation and rotation set to 0. After training, we sample from the representation latent space and reconstruct the images with no rotation and translation. We observe that the reconstructed images identifies different conformations of the particle.

Learning arm motion of particle in CODH/ACS We have about 14000 40x40 pixels images of the CODH/ACS protein complex. We train TARGET-VAE with p_8 and z_dim=2 for 100 epochs. The prior over θ is set to π and we set the generator to have 6 fully-connected layers with 512 hidden units in each. After training, we sample from the transformation-invariant representation space and reconstruct the particle to identify different conformations of the protein. Figure 12 shows some of these different conformations.

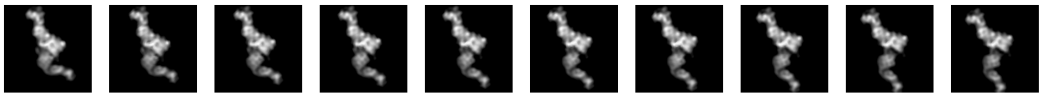

Figure 11: Reconstructed proteins from 5HDB dataset by sampling from 1D representation space, shows the movement in the bottom half of the particle.

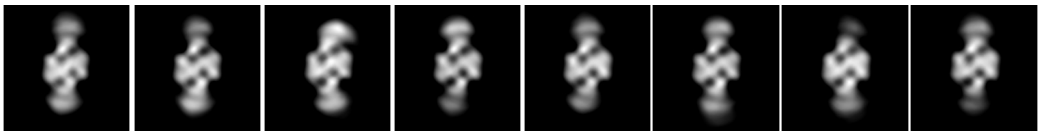

Figure 12: Reconstructed proteins from CODH/ACS dataset, where the motion in the upper and lower arms of the particle can be captured.