Appendix for Deep Goal-Oriented Clustering

Anonymous authors
Paper under double-blind review

1 Pseudocode of DGC

We present the pseudocode of DGC in terms of training (Algorithm 1) and testing (Algorithm 2).

```
Algorithm 1: Deep Goal-Oriented Clustering—Training
Input: the observation $x$, the side-information $y$, the encoder network $f_{\text{enc}}$, the decoder network
    $f_{\text{dec}}$, the task networks $\{f^i_{\text{task}}\}^K_{i=1}$ where $K$ is the number of clusters wanted
Result: The predicted cluster assignment $\hat{c}$
1. Encode input $x$, map to the parameters of $q(\mathbf{z}|\mathbf{x})$: $\boldsymbol{\mu_z}, \boldsymbol{\Sigma_z} = f_{\text{enc}}(x)$
2. Sample latent representations from $q(\mathbf{z}|\mathbf{x})$: $z \sim \mathcal{N}(\boldsymbol{\mu_z}, \boldsymbol{\Sigma_z})$
3. Decode $z$ and reconstruct: $\boldsymbol{\theta}_x = f_{\text{dec}}(z)$
4. Sample reconstruction: $\hat{x} \sim p(\boldsymbol{x}|\boldsymbol{\theta})$
5. Map $z$ to parameters of $p(\mathbf{y}|\mathbf{z}, c)$:
if y is discrete then
    $\boldsymbol{\pi_y} = f^i_{\text{task}}(z), \ \forall i$
else
    $\boldsymbol{\mu_y}, \boldsymbol{\Sigma_y} = f^i_{\text{task}}(z), \ \forall i$
end
6. $\mathcal{L} = \mathcal{L}_{\text{ELBO}} + \mathcal{L}_{\text{task}}(y, \hat{y})$ and backpropogate
7. Calculate
    $$q^{\text{train}}(c = k|\mathbf{z}, \mathbf{y}) = \frac{e^{\log p(\mathbf{y}|\mathbf{z},k) - \mathbb{H}_{\mathbf{max}}(p_{\mathbf{y}|\mathbf{z},k})} \cdot p(k|\mathbf{z})}{\sum_j e^{\log p(\mathbf{y}|\mathbf{z},j) - \mathbb{H}_{\mathbf{max}}(p_{\mathbf{y}|\mathbf{z},j})} \cdot p(j|\mathbf{z})}. \tag{1}$$
8. Obtain cluster prediction $\hat{c} = \text{argmax}_k \{q^{\text{train}}(c = k|\mathbf{z}, \mathbf{y})\}^K_{k=1}$
```

```
Algorithm 2: Deep Goal-Oriented Clustering—Testing
Input: the observation $x$, the encoder network $f_{\text{enc}}$, the decoder network $f_{\text{dec}}$, the task networks
    $\{f^i_{\text{task}}\}^K_{i=1}$ where $K$ is the number of clusters wanted
Result: The predicted cluster assignment $\hat{c}$, the predicted side-information $\hat{y}$
1. Encode input $x$, map to the parameters of $q(\mathbf{z}|\mathbf{x})$: $\boldsymbol{\mu_z}, \boldsymbol{\Sigma_z} = f_{\text{enc}}(x)$
2. Sample latent representations from $q(\mathbf{z}|\mathbf{x})$: $z \sim \mathcal{N}(\boldsymbol{\mu_z}, \boldsymbol{\Sigma_z})$
3. Decode $z$ and reconstruct: $\boldsymbol{\theta}_x = f_{\text{dec}}(z)$
4. Sample reconstruction: $\hat{x} \sim p(\boldsymbol{x}|\boldsymbol{\theta})$
5. Map $z$ to parameters of $p(\mathbf{y}|\mathbf{z}, c)$:
if y is discrete then
    $\boldsymbol{\pi_y} = f^i_{\text{task}}(z), \ \forall i$
else
    $\boldsymbol{\mu_y}, \boldsymbol{\Sigma_y} = f^i_{\text{task}}(z), \ \forall i$
end
6. Sample $\hat{y}$ from $p(\mathbf{y}|\mathbf{z}, c)$: $\hat{y} \sim p(\mathbf{y}|\mathbf{z}, c)$
7. Calculate the entropies of the task network distributions $\{\mathbb{H}_{\mathbf{max}}(p_{\mathbf{y}|\mathbf{z},k})\}^K_{k=1}$
8. Calculate
    $$q^{\text{test}}(c = k|\mathbf{z}, \mathbf{y}) = \frac{e^{-\mathbb{H}_{\mathbf{max}}(p_{\mathbf{y}|\mathbf{z},k})} \cdot p(k|\mathbf{z})}{\sum_j e^{-\mathbb{H}_{\mathbf{max}}(p_{\mathbf{y}|\mathbf{z},j})} \cdot p(j|\mathbf{z})} \tag{2}$$
9. Obtain cluster prediction $\hat{c} = \text{argmax}_k \{q^{\text{test}}(c = k|\mathbf{z}, \mathbf{y})\}^K_{k=1}$
```

2 Mathematical Details

2.1 Intuitive Explanation for the Choice of $q(\mathbf{z}|\mathbf{x})$

Recall that we choose $q(\mathbf{z}|\mathbf{x})$ to be $\mathcal{N}\left(\mathbf{z}|\tilde{\boldsymbol{\mu}}_{\mathbf{z}}, \tilde{\boldsymbol{\sigma}}_{\mathbf{z}}^2\mathbf{I}\right)$ where $\left[\tilde{\boldsymbol{\mu}}_{\mathbf{z}}, \tilde{\boldsymbol{\sigma}}_{\mathbf{z}}^2\right] = h(\mathbf{x};\theta)$, with h being parameterized as a feed-forward neural network with weights θ. Although it may seem unnatural to use a unimodal distribution to approximate a multimodal distribution, when the learned $q(c|\mathbf{z},\mathbf{y})$ becomes discriminative, dissecting the $\mathcal{L}_{\text{ELBO}}$ proposed in the main manuscript in the following way indicates that such an approximation will not incur a sizeable information loss (see the Appendix for a detailed derivation):

$$\begin{aligned}\mathcal{L}_{\text{ELBO}} = \mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log p(\mathbf{y}|\mathbf{z},c) + &\mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \log p(\mathbf{x}|\mathbf{z}) \\ &- \mathbb{KL}\left(q(c|\mathbf{z},\mathbf{y})||p(c)\right) - \sum_k \lambda_k \mathbb{KL}\left(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|c=k)\right)\ ,\end{aligned} \tag{3}$$

where λ_k denotes $q(c=k|\mathbf{z},\mathbf{y})$. Analyzing the last term in Eq. equation 3, we notice that if the learned variational posterior $q(c|\mathbf{z},\mathbf{y})$ is very discriminative and puts most of its weight on one specific index c, all but one $\mathbb{KL}$ terms in the weighted sum will be close to zero. Therefore, choosing $q(\mathbf{z}|\mathbf{x})$ to be unimodal to minimize that specific $\mathbb{KL}$ term would be appropriate, as $p(\mathbf{z}|c)$ is assumed to be a unimodal normal distribution for all c.

2.2 Theoretical Derivations

This section provides detailed derivations for the theoretical claims made in the main manuscript.

Theorem 1. *The variational lower bound for* DGC *is*

$$\log p(\boldsymbol{x},\boldsymbol{y}) \geq \underbrace{\mathbb{E}_{q(\boldsymbol{z},c|\boldsymbol{x},\boldsymbol{y})} \log p(\boldsymbol{y}|\boldsymbol{z},c)}_{\textit{Probabilistic Ensemble}} + \underbrace{\mathbb{E}_{q(\boldsymbol{z},c|\boldsymbol{x},\boldsymbol{y})} \log \frac{p(\boldsymbol{x},\boldsymbol{z},c)}{q(\boldsymbol{z},c|\boldsymbol{x},\boldsymbol{y})}}_{\textit{ELBO for VAE with GMM prior}} = \mathcal{L}_{\textit{ELBO}}\ . \tag{4}$$

Proof. We derive the $\mathcal{L}_{\text{ELBO}}$ as follows

$$\begin{aligned}\log p(\mathbf{x},\mathbf{y}) &= \log \int_{\mathbf{z}} \sum_c p(\mathbf{x},\mathbf{y},\mathbf{z},c)d\mathbf{z} \\ &= \log \int_{\mathbf{z}} \sum_c \frac{p(\mathbf{x},\mathbf{y},\mathbf{z},c)}{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} q(\mathbf{z},c|\mathbf{x},\mathbf{y})d\mathbf{z} \\ &\geq \mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log \frac{p(\mathbf{x},\mathbf{y},\mathbf{z},c)}{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \\ &= \underbrace{\underbrace{\mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log p(\mathbf{y}|\mathbf{z},c)}_{\text{Probabilistic Ensemble}} + \underbrace{\mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log \frac{p(\mathbf{x},\mathbf{z},c)}{q(\mathbf{z},c|\mathbf{x},\mathbf{y})}}_{\text{ELBO for VAE with GMM prior}}}_{\mathcal{L}_{\text{ELBO}}}\ .\end{aligned} \tag{5}$$

□

Proposition 1. *To explain the fact that choosing $q(\boldsymbol{z}|\boldsymbol{x})$ to be unimodal will not incur a sizable information loss when the learned $q(c|\boldsymbol{x})$ is discriminative, we dissect the $\mathcal{L}_{ELBO}$ as follows (Eq.3 in the main paper)*

$$\begin{aligned}\mathcal{L}_{ELBO} = \mathbb{E}_{q(\boldsymbol{z},c|\boldsymbol{x},\boldsymbol{y})} \log p(\boldsymbol{y}|\boldsymbol{z},c) + &\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})} \log p(\boldsymbol{x}|\boldsymbol{z}) \\ &- \mathbb{KL}\left(q(c|\boldsymbol{y},\boldsymbol{z})||p(c)\right) - \sum_k \lambda_k \mathbb{KL}\left(q(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}|c=k)\right)\ .\end{aligned} \tag{6}$$

Proof.

$$\begin{aligned}
\mathcal{L}_{\text{ELBO}} &= \mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log p(\mathbf{y}|\mathbf{z},c) + \mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log \frac{p(\mathbf{x},\mathbf{z},c)}{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \\
&= \mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log p(\mathbf{y}|\mathbf{z},c) + \mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log \frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z}|c)p(c)}{q(\mathbf{z}|\mathbf{x})q(c|\mathbf{z},\mathbf{y})} \\
&= \mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log p(\mathbf{y}|\mathbf{z},c) + \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \log p(\mathbf{x}|\mathbf{z}) - \mathbb{KL}\left(q(c|\mathbf{y},\mathbf{z})||p(c)\right) + \mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log \frac{p(\mathbf{z}|c)}{q(\mathbf{z}|\mathbf{x})}
\end{aligned} \tag{7}$$

where

$$\mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log \frac{p(\mathbf{z}|c)}{q(\mathbf{z}|\mathbf{x})} = \mathbb{E}_{q(c|\mathbf{y},\mathbf{z})}\mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \log \frac{p(\mathbf{z}|c)}{q(\mathbf{z}|\mathbf{x})} = \sum_k \lambda_k \mathbb{KL}\left(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|c=k)\right)$$

where $\lambda_k = q(c=k|\mathbf{y},\mathbf{z})$. □

Proposition 2. *Choosing $q(c|\boldsymbol{z},\boldsymbol{y}))$ requires us to decompose $\mathcal{L}_{ELBO}$ as follows*

$$\mathcal{L}_{ELBO} = \mathbb{E}_{q(\boldsymbol{z},c|\boldsymbol{x},\boldsymbol{y})} \log p(\boldsymbol{y}|\boldsymbol{z},c) + \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})} \log \frac{p(\boldsymbol{x},\boldsymbol{z})}{q(\boldsymbol{z}|\boldsymbol{x})} - \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}\mathbb{KL}\left(q(c|\boldsymbol{z},\boldsymbol{y})||p(c|\boldsymbol{z})\right). \tag{8}$$

Proof.

$$\begin{aligned}
\boldsymbol{\mathcal{L}}_{\text{ELBO}} &= \mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log p(\mathbf{y}|\mathbf{z},c) + \mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log \frac{p(\mathbf{x},\mathbf{z},c)}{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \\
&= \mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log p(\mathbf{y}|\mathbf{z},c) + \mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log \frac{p(c|\mathbf{x},\mathbf{z})p(\mathbf{x},\mathbf{z})}{q(\mathbf{z}|\mathbf{x})q(c|\mathbf{z},\mathbf{y})} \\
&= \mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log p(\mathbf{y}|\mathbf{z},c) + \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}\mathbb{E}_{q(c|\mathbf{z},\mathbf{y})} \left[\log \frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z}|\mathbf{x})} - \log \frac{q(c|\mathbf{z},\mathbf{y})}{p(c|\mathbf{z})}\right] \\
&= \mathbb{E}_{q(\mathbf{z},c|\mathbf{x},\mathbf{y})} \log p(\mathbf{y}|\mathbf{z},c) + \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \log \frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z}|\mathbf{x})} - \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}\mathbb{KL}\left(q(c|\mathbf{z},\mathbf{y})||p(c|\mathbf{z})\right)
\end{aligned} \tag{9}$$

□

Proposition 3. *The solution to the following convex program*

$$\begin{aligned}
\min_{q(c|\boldsymbol{z},\boldsymbol{y})} \quad & f_0(q) = \mathbb{KL}\left(q(c|\boldsymbol{z},\boldsymbol{y})||p(c|\boldsymbol{z})\right) - \mathbb{E}_{q(c|\boldsymbol{z},\boldsymbol{y})} \log p(\boldsymbol{y}|\boldsymbol{z},c)\,, \\
\textit{s.t.} \quad & \sum_k q(c=k|\boldsymbol{z},\boldsymbol{y}) = 1, \quad q(c=k|\boldsymbol{z},\boldsymbol{y}) \geq 0, \ \ \forall k
\end{aligned} \tag{10}$$

is

$$q(c=k|\boldsymbol{z},\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{z},c=k)\cdot p(c=k|\boldsymbol{z})}{\sum_k p(\boldsymbol{y}|\boldsymbol{z},c=k)\cdot p(c=k|\boldsymbol{z})}\,. \tag{11}$$

Proof. First, we note that the constraint, $q(c=k|\mathbf{z},\mathbf{y}) \geq 0$ for all k, is not needed (and effectively redundant), as the $\mathbb{KL}$ term in the objective function is not defined otherwise. Now consider a convex program that takes the form of

$$\begin{aligned}
\min_{\mathbf{t}\in\mathbb{R}^k_+} \quad & f_0(\mathbf{t}) \\
\text{s.t.} \quad & \mathbf{1}^T\mathbf{t} = 1\,.
\end{aligned} \tag{12}$$

where f_0 is a convex function and "$\succeq$" denotes "element-wise greater than or equal to." Forming the Lagrangian, we have

$$\mathbf{L}\left(\mathbf{t},\gamma\right) = f_0(\mathbf{t}) + \gamma\left(\mathbf{1}^T\mathbf{t} - 1\right)$$

The *Karush–Kuhn–Tucker conditions* state that the optimal solution dual, $(\mathbf{t}^*,\gamma^*)$, satisfies the following

- $-\mathbf{t}^* \preceq 0$
- $\mathbf{1}^T\mathbf{t}^* - 1 = 0$
- $\nabla_{\mathbf{t}}\mathbf{L}\left(\mathbf{t}^*, \gamma^*\right) = 0$

Since

$$\nabla_{\mathbf{t}}\mathbf{L}\left(\mathbf{t}, \gamma\right) = \nabla_{\mathbf{t}} f_0(\mathbf{t}) + \gamma \cdot \mathbf{1}$$

the third condition implies that

$$\nabla_{\mathbf{t}}\mathbf{L}\left(\mathbf{t}^*, \gamma^*\right) = \nabla_{\mathbf{t}} f_0(\mathbf{t}^*) + \gamma^* \cdot \mathbf{1} = 0\,. \tag{13}$$

Let $\mathbf{t} = q(c|\mathbf{z}, \mathbf{y})$ (i.e. $t_k = q(c = k|\mathbf{y}, \mathbf{z})$), and $f_0(\mathbf{t})$ as being specified in Eq. 10, we have

$$\begin{aligned}\nabla_{t_k} f_0(\mathbf{t}) &= \frac{\partial}{\partial t_k}\left(\sum_k t_k \log \frac{t_k}{p(c = k|\mathbf{z})} - \sum_k t_k \log p(\mathbf{y}|\mathbf{z}, c = k)\right) \\ &= \log \frac{t_k}{p(c = k|\mathbf{z})} + 1 - \log p(\mathbf{y}|\mathbf{z}, c = k)\,.\end{aligned} \tag{14}$$

Based on the condition in Eq. 13, we thus have

$$\nabla_{t_k}\mathbf{L}\left(\mathbf{t}^*, \gamma^*\right) = \log \frac{t_k^*}{p(c = k|\mathbf{z})} + 1 - \log p(\mathbf{y}|\mathbf{z}, c = k) + \gamma^* = 0$$

which leads to

$$t_k^* = e^{\log p(\mathbf{y}|\mathbf{z}, c=k) - 1 - \gamma^*} \cdot p(c = k|\mathbf{z})\,.$$

Since γ^* is chosen in a way such that $\sum_k t_k^* = 1$ (by the second condition), we obtain the solution

$$t_k^* = \frac{t_k^*}{\sum_k t_k^*} = \frac{p(\mathbf{y}|\mathbf{z}, c = k) \cdot p(c = k|\mathbf{z})}{\sum_k p(\mathbf{y}|\mathbf{z}, c = k) \cdot p(c = k|\mathbf{z})}\,. \tag{15}$$

□

3 Experimental Details

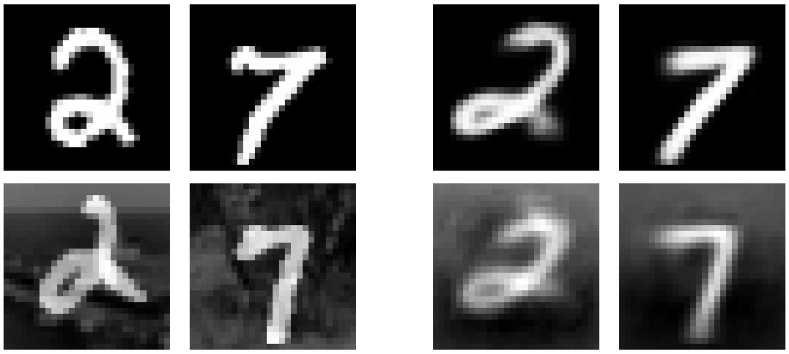
This section provides a detailed description of the experimental setups, such as the train/test splits, the chosen network architectures, and the choices of learning rate and optimizer, for the experiments conducted. We describe the architecture of DGC in terms of its encoder, decoder, and task network. We adopt the following abbreviations for some basic network layers

- $\texttt{FL}(d_i, d_o, f)$ denotes a fully-connected layer with d_i input units, d_o output units, and activation function f.
- $\texttt{Conv}(c_i, c_o, k_1, f, \texttt{BatchNorm2d}, O(k_2, s))$ denotes a convolution layer with c_i input channels, c_o output channels, kernel size k_1, activation function f, and pooling operation $O(k_2, s)$ with another kernel size k_2 and stride s.

3.1 Noisy MNIST

We extract images that correspond to the digits 2 and 7 from MNIST. The MNIST dataset is predivided into training/testing sets, so we naturally use the images that correspond to the digits 2 and 7 from the training set as our training data (12,223 images), and that from the testing set as our testing data (2,060 images). For each digit, we randomly select half of the images for that digit and superpose noisy backgrounds onto those images, where the backgrounds are cropped from randomly selected CIFAR-10 images (more specifically, we first randomly select a class, and then randomly select a CIFAR image that corresponds to that class). See Fig. 1 for the ground truth and generated noisy MNIST samples.

We use the Adam optimizer for optimization. We train with a batch size of 128 images, an initial learning rate of 0.002, and a learning rate decay of 10% after every 10 epochs, for 100 epochs. We use the following network architecture:



(a) Ground Truth Images

(b) Generated Images from DGC

Figure 1: Ground truth and generated noisy MNIST images.

Encoder
FL(784,500,ReLU)
FL(500,500,ReLU)
FL(500,2000,ReLU)
FL(2000,10,ReLU)

Decoder
FL(10, 2000,ReLU)
FL(200,500,ReLU)
FL(500,500,ReLU)
FL(500,784,ReLU)

Task Network
FL(10, 4,Sigmoid)

3.2 Pacman

This section provides more details for our Pacman experiments.

3.2.1 Experimental Setup

We create 20,000 points, with 10,000 for the outer annulus and 10,000 for the inner annulus. Both annuli center at the origin, with the outer annulus having a radius of 1 and the inner annulus having a radius of 0.8. We create the training set by sampling 7,500 points from each annulus, and leave the rest of the data for testing. We create the linear responses by dividing the [0,1] range into 10,000 sub-intervals and assign the split points to the points in the inner annulus in a way that it is increasing (from 1 to 0) in the clockwise direction. We create the exponential responses by evaluating the exponential function at the aforementioned split points (generated for the linear responses), and then assign them to the points on the outer annulus in a way that it is decreasing (from 0 to 1) in the clockwise direction. Fig. 3 shows the ground truth (first row) and the generated (second row) 2D Pacman annuli, the responses, and the 3D view of the entire dataset.

We use the Adam optimizer for optimization. We train with a batch size of 1,000 points, an initial learning rate of 0.001, and a learning rate decay of 10% after every 10 epochs, for 80 epochs. We use the following network architecture:

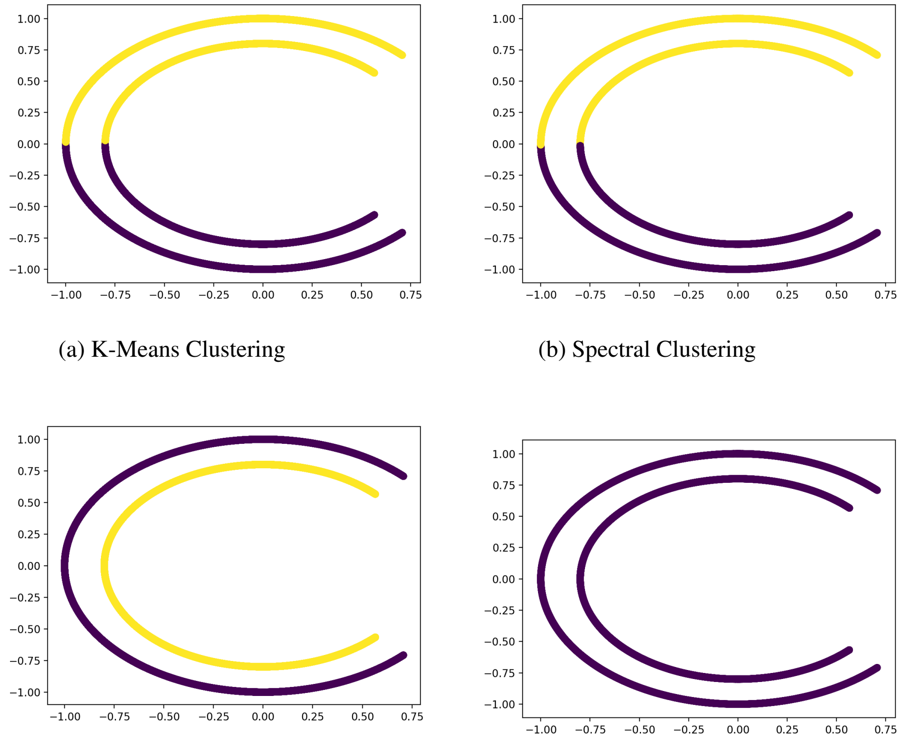
Encoder
FL(2,64,Sigmoid)
FL(64,128,Sigmoid)
FL(128,256,Sigmoid)
FL(256,60,Sigmoid)

Decoder
FL(60, 256,Sigmoid)
FL(256,128,Sigmoid)
FL(128,64,Sigmoid)
FL(64,2,Sigmoid)

Task Network
FL(64, 128,Sigmoid)
FL(128,4,Sigmoid)

3.2.2 Attempts using Unsupervised Methods

As mentioned in the main manuscript, we have tried the following unsupervised methods on the Pacman dataset to see how they perform: K-means clustering, hierarchical clustering with single linkage, spectral clustering, and VaDE. Fig. 2 shows the clustering results. Besides the hierarchical clustering with single linkage, none of the other methods can separate the two annuli in a satisfactory way. It is worth noting that hierarchical clustering with other distance metrics would not work, and choosing single linkage requires advanced, accurate knowledge on the dataset, which one would



(a) K-Means Clustering

(b) Spectral Clustering

(c) Hierarchical Clustering with Single Linkage

(d) VaDE Clustering

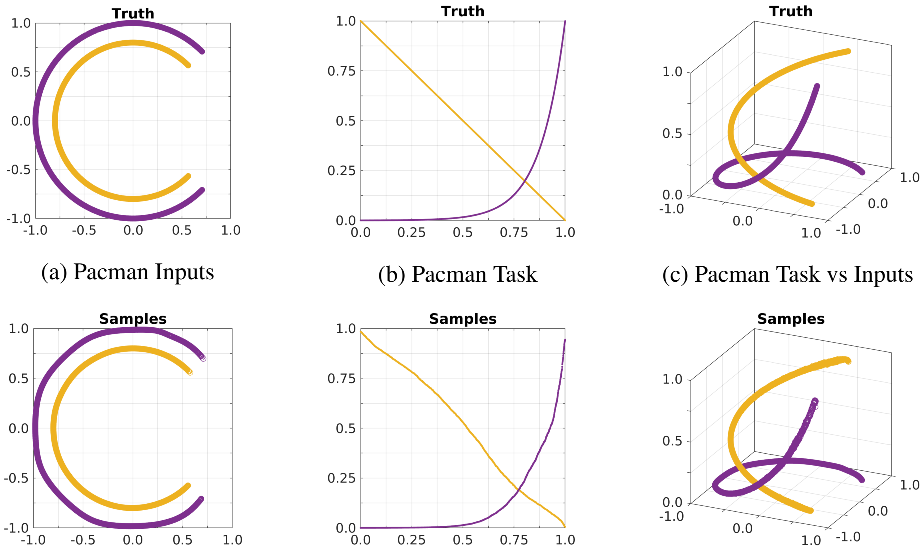
Figure 2: Clustering results obtained from four different unsupervised clustering methods, namely (a) K-means clustering; (b) spectral clustering; (c) hierarchical clustering; and (d) VaDE.

not usually have for more complicated data (e.g, images or sound). This phenomenon echos a deep-rooted obstacle for clustering methods in general: the concept of clusters is inherently subjective, and different clustering methods can potentially produce very different clustering results. Furthermore, for most unsupervised clustering methods, it is nontrivial to incorporate a prior given information about the clusters, even when such information exists.

3.3 SVHN

We apply DGC to the Street View House Number (SVHN) dataset. We follow the standard procedure to pre-process the images (Zagoruyko & Komodakis, 2016; Devries & Taylor, 2017), only normalizing them so that the pixel values are within the [0,1] range.

We use the Adam optimizer for optimization. We train with a batch size of 128 images and a learning rate of 0.0001 (stays constant throughout epochs) for 150 epochs. We use the following network architecture:



(a) Pacman Inputs

(b) Pacman Task

(c) Pacman Task vs Inputs

(d) Pacman Inputs

(e) Pacman Task

(f) Pacman Task vs Inputs

Figure 3: The first row shows the ground truth 2D Pacman, the responses **y** alone, and the combined 3D Pacman. The second row depicts the corresponding generated counterparts from DGC.

Encoder
Conv(3, 48, 5, ReLU, BatchNorm2d, MaxPool(2, 2))
Conv(48, 64, 5, ReLU, BatchNorm2d, MaxPool(2, 1))
Conv(64, 128, 5, ReLU, BatchNorm2d, MaxPool(2, 2))
Conv(128, 160, 5, ReLU, BatchNorm2d, MaxPool(2, 1))
Conv(160, 192, 5, ReLU, BatchNorm2d, MaxPool(2, 2))
Conv(192, 192, 5, ReLU, BatchNorm2d, MaxPool(2, 1))
Conv(192, 192, 5, ReLU, BatchNorm2d, MaxPool(2, 2))
Conv(192, 192, 5, ReLU, BatchNorm2d, MaxPool(2, 1))
FL(9408,3072, ReLU)
FL(3072,256, ReLU)

Decoder
FL(256, 3072,ReLU)
Conv(3072, 256, 4, ReLU, BatchNorm2d, MaxPool(2, 2))
Conv(256, 128, 4, ReLU, BatchNorm2d, MaxPool(2, 1))
Conv(128, 64, 4, ReLU, BatchNorm2d, MaxPool(2, 2))
Conv(64, 3, 4, ReLU, BatchNorm2d, MaxPool(2, 1))

Task Network
FL(256, 512, Sigmoid)
FL(512,1024, Sigmoid)
FL(1024,512, Sigmoid)
FL(512,256, Sigmoid)
FL(256,100, Sigmoid)

3.4 Carolina Breast Cancer Study (CBCS)

3.4.1 Data Processing

Due to the fact that the histopathological images collected in CBCS are large (of size $3 \times 3000 \times 3000$), we use a pretrained VGG16 network to extract feature representations for each image, and use the extracted, fixed features as the input to DGC. The features are of dimension 512, and are the output of the 8th layer of the pretrained VGG16 network.

As mentioned in the main manuscript, each patient has 2-4 associated histopathologial images. Due to the scarce nature of medical data, we treat each image as an individual "patient" during training. At test time, we obtain patient-level prediction by aggregating image-level predictions (i.e. taking majority vote), and disregard patients with ambiguous patient-level predictions (e.g. a patient has 4 associated images. 2 of the images are predicted to be in cluster 0 and the other 2 are predicted to be in cluster 1). The number of disregarded patients accounts for 3.4% of the entire population.

Finally, again due to the scarce nature of this dataset, we use 10-fold cross validation to obtain predictions on the entire dataset. More specifically, we split the dataset into 10 subsets, train on 9 of those subsets and predict on the remaining subset. We then repeat this process 10 times to obtain the final predictions on the entire dataset.

3.4.2 Experimental Setup

We use the Adam optimizer for optimization. We train with a batch size of 32, and an initial learning rate of 0.001 and decay rate of 0.9 (for every 10 epochs), for 150 epochs. The network architecture used is as follows

Encoder
FL(512,1024,ReLU)
FL(1024,2048,ReLU)
FL(2048,5,ReLU)

Decoder
FL(5, 2048,ReLU)
FL(2048,1024,ReLU)
FL(1024,512,ReLU)

Task Network
FL(5,3,Sigmoid)

3.4.3 Tumor Characteristics for Clusters Obtained from VaDE

We present the tumor characteristics for each cluster obtained from VaDE in Tab. 1. As one can see, cluster 0, who has the highest recurrence rate, should have the most negative ER subtype, the most high grade, and the most Basal-like tumor subtype. As for grade, it is not the cluster with the most high grade patients. For ER status and tumor subtype, it does have the highest negative ER subtype and the most Basal-like tumor subtype, but the differences are much less significant compared to the clusters obtained from DGC.

Table 1: Tumor characteristics for each cluster from VaDE. Features are color-coded as low, intermediate, or high risk.

		Cluster 0 N(%)	Cluster 1 N(%)	Cluster 2 N(%)
ER Status	Positive	24 (51.1)	9 (56.3)	88 (63.3)
	Negative	23 (48.9)	7 (43.8)	51 (36.7)
Grade	Low	6 (12.8)	0 (0)	22 (15.8)
	Medium	8 (17.0)	3 (18.8)	47 (33.8)
	High	33 (70.2)	13 (81.2)	70 (50.4)
Tumor Subtype	Luminal A	8 (22.9)	3 (23.1)	44 (44.0)
	Luminal B	5 (14.3)	3 (23.1)	16 (16.0)
	ER-/HER2+	1 (2.9)	0 (0.0)	9 (9.0)
	Basal-like	21 (60.0)	7 (53.8)	31 (31.0)

References

Terrance Devries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *ArXiv*, abs/1708.04552, 2017.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *ArXiv*, abs/1605.07146, 2016.