Appendices

A. Total Correlation Approximation

The density ratio of the total correlation term is:

$$r(\boldsymbol{h}) = \frac{\gamma}{\gamma_1 \cdot \gamma_2}. \tag{10}$$

Let γ_3 denote $\gamma_1 \cdot \gamma_2$. Assume there are equal numbers of pairs of $\boldsymbol{h}_s$ and $\boldsymbol{h}_n$, when $y = 1$, $\boldsymbol{h}_s$ and $\boldsymbol{h}_n$ inter-dependent, and when $y = 0$, $\boldsymbol{h}_s$ and $\boldsymbol{h}_n$ are independent. We have

$$\begin{aligned} r(\boldsymbol{h}) &= \frac{\gamma}{\gamma_1 \cdot \gamma_2} && (11)\\ &= \frac{\gamma}{\gamma_3} && (12)\\ &= \frac{\tau(\boldsymbol{h} \mid y=1)}{\tau(\boldsymbol{h} \mid y=0)} && (13)\\ &= \frac{\tau(y=1 \mid \boldsymbol{h})\tau(\boldsymbol{h})/\tau(y=1)}{\tau(y=0 \mid \boldsymbol{h})\tau(\boldsymbol{h})/\tau(y=0)} && (14)\\ &= \frac{\tau(y=1 \mid \boldsymbol{h})}{\tau(y=0 \mid \boldsymbol{h})} && (15)\\ &= \frac{\tau(y=1 \mid \boldsymbol{h})}{1-\tau(y=1 \mid \boldsymbol{h})}. && (16) \end{aligned}$$

If we use a classifier/discriminator $Dis_\varphi(\boldsymbol{h})$ to approximate the term $\tau(y = 1 \mid \boldsymbol{h})$, we can write the density ratio above as:

$$r(\boldsymbol{h}) \approx \frac{Dis_\varphi(\boldsymbol{h})}{1 - Dis_\varphi(\boldsymbol{h})}. \tag{17}$$

Thus, the total correlation can be approximated by:

$$TC \approx \mathbb{E}_\gamma \left(\log \frac{Dis_\varphi(\boldsymbol{h})}{1 - Dis_\varphi(\boldsymbol{h})} \right), \tag{18}$$

B. Implementation

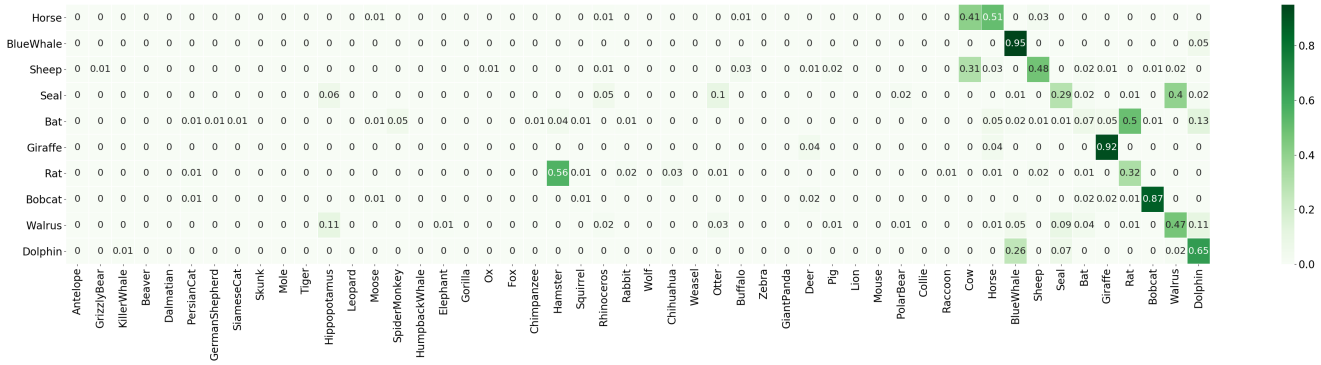
Our framework is implemented by the popular deep learning framework PyTorch 1.4. The disentangling encoder E_ψ and decoder D_ω contain one and two fully connected (FC) layers, respectively. Each layer is followed a LeakyReLU activation function layer and a dropout layer. The numbers of hidden units of E_ψ and D_ω are $[l + m]$ and $[2048, 2048]$, where l and m are the numbers of dimensions of $\boldsymbol{h}_s$ and $\boldsymbol{h}_n$. The relation module R_κ is built with two FC layers, where the first layer is followed by a ReLU activation function layer and the second layer is followed by a Sigmoid activation function layer. The discriminator is implemented with a single FC layer followed by a Sigmoid activation function layer. We set the dimension number l and m as the same value between 32 and 512. The hyper-parameters for relation weight λ_1, TC weight λ_2, and discriminator loss λ_3 are set between 0.1 and 5. We use Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and batch size with 64. For the generative model cVAE, we use a five-layer MLP for encoder Q_ϕ and its structure can be written as: FC-LeakyReLU-FC-Dropout-LeakyReLU-FC for the first part; a single FC layer for the mean vector output; FC-Dropout-Softplus for the variance vector output. Another three-layer MLP for decoder P_θ can be written as: FC-ReLU-Dropout-FC-LeakyReLU. We warm up the KL term and the TC term gradually with increasing epochs. All the experiments are performed on a Lenovo workstation with two NVIDIA GeForce GTX 2080 Ti GPUs.

C. Class-wise Analysis

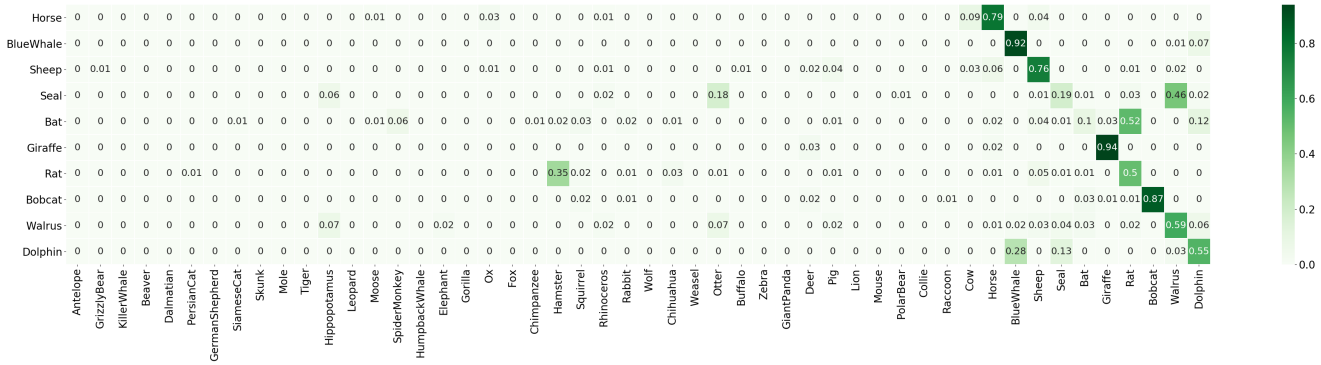
To validate the merit of specific classes in our disentanglement approach, we compare the class-wise performance of unseen classes in AWA between the base generative method, *i.e.,* a standard cVAE, and the proposed SDGZSL. As shown in Figure 8, the top sub-figure is the confusion matrix of the cVAE while the bottom one is of the proposed SDGZSL. The rows represent the groundtruth labels of the target test samples while the columns represent the predicted labels of the test samples. As unseen classes are hard to achieve high performance and can be usually misclassified into seen classes, we take the test samples from unseen classes to compare between the two settings. There are 10 unseen classes in AWA dataset, and almost all these classes in SDGZSL gain higher accuracy than cVAE. The test samples of unseen classes are usually misclassified into visually similar classes. Notably, in cVAE we can see that 41% of test samples from category "horse" are misclassified into "sheep", 31% from "sheep" to "cow", 35% from "rat" to "hamster". In contrast, our approach shows the ability to alleviate the problem by reducing the misclassification rate to 9%, 3%, and 35% for categories "horse", "sheep" and "rat", respectively. However, some extremely hard categories, *e.g.,* "seal" and "bat" can be easily misclassified into "walrus" and "rat", also fail in our proposed method. This will be investigated in our future work.

D. Comparison with Traditional Methods

To further demonstrate the superiority of the disentangled semantic-consistent representations, we conduct experiments on traditional embeddings methods. Specifically, we use the converged encoder E_ψ to process the original visual features. The learned semantic-consistent representations $\boldsymbol{h}_s$ substitute the original visual features to learn a compatibility function. We choose the standard embeddings method ALE [1] as the base method. Table 2 shows the performance comparison between our approach and the representative traditional embedding-based approaches. It



(a) cVAE



(b) SDGZSL

Figure 8: The comparison between cVAE and the proposed method on confusion matrices of the unseen images being predicted over all classes. The vertical axis denotes the ground truth and the horizontal axis represents the predictions.

Table 2: Performance comparison in accuracy (%) with traditional methods on four datasets. We report the accuracies of unseen, seen classes and their harmonic mean, which are denoted as U, S and H. The best results of the harmonic mean are highlighted in bold.

	aPaY			AWA			CUB			FLO		
Methods	*U*	*S*	*H*	*U*	*S*	*H*	*U*	*S*	*H*	*U*	*S*	*H*
DAP [19]	4.8	78.3	9.0	0.9	84.7	0.0	1.7	67.9	3.3	-	-	-
LATEM [34]	0.1	73.0	0.2	13.3	77.3	20.0	15.2	57.3	24.0	6.6	47.6	11.5
ALE [1]	4.6	73.7	8.7	14.0	81.8	23.9	23.7	62.8	34.4	13.3	61.6	21.9
DeVise [9]	3.5	78.4	6.7	17.1	74.7	27.8	23.8	53.0	32.8	13.2	82.6	22.8
SJE [2]	1.3	71.4	2.6	8.0	73.9	14.4	23.5	59.2	33.6	-	-	-
ESZSL [27]	2.4	70.1	4.6	5.9	77.8	11.0	14.7	56.5	23.3	-	-	-
SAE [17]	0.4	80.9	0.9	1.1	82.2	2.2	7.8	54.0	13.6	-	-	-
SDGZSL-ALE	11.1	71.8	**19.3**	21.4	88.1	**34.4**	25.6	63.4	**36.5**	24.8	80.0	**37.9**

can be seen from the table that traditional embedding-based methods perform poorly on GZSL setting, especially on unseen classes. These traditional embedding-based methods are originally proposed for conventional zero-shot learning setting that only aims to classify test unseen samples over unseen classes. We argue that the simple embeddings functions cannot draw a clear distinction between the seen class domain and the unseen class domain so that under GZSL setting the unseen class samples tend to be misclassified into seen classes. However, our semantic-consistent representations can alleviate this problem. From the performance table, training with our semantic-consistent representations instead of the original visual features, ALE can boost the performance by a large margin. The improvement verifies the disentangled semantic-consistent representations can help to transfer visual-semantic relationship from seen classes to unseen classes.