Unsupervised image translation with distributional semantics awareness

Zhexi Peng
State Key Lab of CAD&CG, Zhejiang University
Hangzhou, China
zhexipeng@zju.edu.cn

He Wang
School of Computing, University of Leeds
Leeds, UK
H.E.Wang@leeds.ac.uk

Yanlin Weng
State Key Lab of CAD&CG, Zhejiang University, Hangzhou
Hangzhou, China
weng@cad.zju.edu.cn

Yin Yang
School of Computing, Clemson University
Clemson, USA
yin5@clemson.edu

Tianjia Shao
State Key Lab of CAD&CG, Zhejiang University, Hangzhou
Hangzhou, China
tjshao@zju.edu.cn

Abstract

Unsupervised image translation (UIT) studies the mapping between two image domains. Since the mapping is under-constrained, existing research has pursued various desirable properties such as distributional matching or two-way consistency. In this paper, we reexamine UIT from a new perspective: distributional semantics consistency, based on the observation that data variations contain semantics, *e.g*. shoes varying in colors. Further, the semantics can be multi-dimensional, *e.g*. shoes also varying in styles, functionalities, *etc*. Given two image domains, matching these semantic dimensions during UIT will produce mappings with explicable correspondences, which has not been investigated previously. We propose the first UIT method, Distributional Semantics Mapping, which explicitly matches the semantics between two domains. We show that distributional semantics has been rarely considered within and beyond UIT, and is a common problem in deep learning. We evaluate DSM on several benchmark datasets, demonstrating its generalizability in capturing distributional semantics. By extensive comparisons, we show that DSM not only produces explicable mappings but also improve the image quality in general.

1. Introduction

Unsupervised image translation (UIT) has been intensively studied in recent years. Its ability of building mappings between two image domains has inspired many applications. Since there can be theoretically an infinite number of mappings between two domains, UIT is in nature an under-constrained problem. Naturally, different approaches have been developed to ensure certain desirable properties, such as shared latent spaces [25], two-way consistency [39], pair-wise distance preservation [3], and image semantics [33]. While existing researches tend to focus on general distributional matching [1, 3], we aim to investigate a rarely examined perspective: the distributional semantics during UIT.

We define distributional semantics as the visually understandable variations in samples (not within a single sample). Shoes vary in colors, styles (*e.g*. low/high collars), functionalities (*e.g*. sneakers/high heels). Similarly, bags also vary in colors, styles (*e.g*. with/without handles), functionalities (*e.g*. purses/backpacks). During UIT, we argue that it is not enough to simply translate images. It will be desirable if these distributional semantics can be maintained,*e.g*. red high-collar high heels mapped to black purses with handles, while white low-collar sneakers mapped to blue backpacks. This is exactly the goal of this research.

Maintaining distributional semantics during UIT requires answers to two key questions. The first question is what semantics to maintain. Due to the unsupervision requirement, no labeling should be required, which means that the data variations (the distributional semantics) should be characterized without prior knowledge but interpretable to humans. Second, distributional semantics are seldom considered in deep learning in general where data are normally transformed many times in the model. To maintain such semantics, we need a mechanism to ensure that the data distribution is as least distorted as possible, especially in the dimensions of the semantics in which we are interested, during transformations when mapping two domains.

In this paper, we propose a new deep learning method called Distributional Semantics Mapping (DSM). Given two image datasets $x \in \mathbf{A}$ and $y \in \mathbf{B}$, to characterize visual semantics in an unsupervised manner, we find that the covariance structure of the data naturally reflects important visual semantics. We choose Principal Component Analysis (PCA) to characterize the covariance structure because Härkönen *et al.* [9] show that interpretable controls for image synthesis can be achieved by PCA applied in feature space. Next, our approach is agnostic about specific network architectures and consists of three key modules. The first module is a semantic-preserving transformation $\mathbf{e}$ where the variations of xs in a direction (*e.g.* the first principal component of $\mathbf{A}$) has to be consistent with the variations of $z_x = \mathbf{e}(x)$ on its corresponding direction (*e.g.* the first principal component of $\mathbf{e}(\mathbf{A})$) in the latent space. We use two such encoders $\mathbf{e_A}$ and $\mathbf{e_B}$ to project $\mathbf{A}$ and $\mathbf{B}$ into a shared latent space. The second module aligns the key dimensions of $\mathbf{e_A}(\mathbf{A})$ and $\mathbf{e_B}(\mathbf{B})$. The last module is a decoder/generative network $\mathbf{g}$ which decodes $y' = \mathbf{g}(\mathbf{e_B}(y))$ and translates x by $\hat{y} = \mathbf{g}(\mathbf{e_A}(x))$.

Formally, as far as we know, we propose the very first distributional semantics preserving method for UIT. We identify the importance of preserving distributional semantics, which has a wide range of implications within and beyond image translation. We propose a new approach that helps preserve the distribution semantics during image transformations.

2. Related work

Generative adversarial networks (GANs). GANs [8] have achieved a great success in a fast-growing number of computer vision tasks, such as image generation [2, 15], image colorization [14], image inpainting [12], and image super-resolution [21]. Conditional GANs [31] can be used to perform image-to-image translation [17, 36, 39]. More recently, GANs supported interactive system is proposed for real-time editing of portrait images [4, 22]. Our work also utilizes GANs conditioned on an input image, but it does not rely on any specific GAN model. To validate its generalizability, we employ two widely used GAN models (LSGAN [29] and NSGAN [8]).

Image-to-image translation. To perform image-to-image translation, early methods often require the networks to be trained with paired training data, such as pix2pix [14, 35]. Recently, a variety of approaches [17, 36] have been proposed to learn the image translation from unpaired data. For example, CycleGAN [39] leverages cycle consistency to constrain the mapping. Lu *et al.* [28, 27] show that optimal transport cost can improve the generative network. UNIT [25] assumes two image domains can share the same latent space. By decomposing the image into the style (domain-specific) code and content (domain-invariant) code, MUNIT [11] and DRIT [23] can synthesize diverse outputs for an input image. Mejjati *et al.* [30] and Kim *et al.* [16] improve the translation results with the attention mechanism. Choi *et al.* [5] propose StarGAN which can perform image translation for multiple domains using a single GAN. More recently, DRIT++ [24] extends DRIT to support multiple domains, while StarGAN v2 [6] extends StarGAN to generate diverse images across multiple domains. FUNIT [26] can works on previously unseen target classes only by a few example images. To enable unsupervised one-sided mapping, Benaim and Wolf [3] present DistanceGAN that maintains the distances between images, and Fu *et al.* [7] employ other geometric constraints (*e.g.*, rotation). Our method differs from existing methods in that it explicitly preserves and matches the distributional semantics in domains during UIT, which generates explicable mappings between images.

3. Methodology

Given two image datasets $x \in \mathbf{A}$ and $y \in \mathbf{B}$, we aim to compute a mapping M: $\mathbf{A} \rightarrow \mathbf{B}$, so that the distributional semantics of $\mathbf{A}$ are aligned with that of $\mathbf{B}$. We use the principal components (PCs) to describe the semantics and first project $\mathbf{A}$ by an encoder to $\mathbf{z_A} = \mathbf{e_A}(\mathbf{A})$ while keeping the distributional semantics of $\mathbf{A}$ and $\mathbf{z_A}$ aligned. We then do a similar projection for $\mathbf{B}$ by $\mathbf{z_B} = \mathbf{e_B}(\mathbf{B})$. Next we ensure the PCs of $\mathbf{z_A}$ to be aligned with those of $\mathbf{z_B}$. Finally, we use a generative network $\mathbf{g}$ to reconstruct $\mathbf{B}$ by $y' = \mathbf{g}(\mathbf{e_B}(y))$ and translate x by $\hat{y} = \mathbf{g}(\mathbf{e_A}(x))$. The model is shown in Figure 1. Below, we give the details of the components.

3.1. Semantics-preserving Transformation

The images need to go through a series of transformations during translation, which in deep learning are usually some encoding processes such as convolutions. However, as far as we know, the shape of the latent distribution is rarely considered in terms of its semantic consistency with the data distribution itself. Existing efforts such as imposing a prior distribution [19] or geometric constraints [3] are

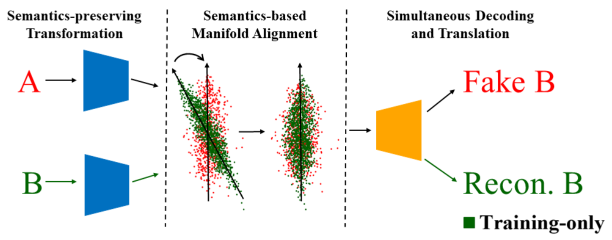
Semantics-preserving Transformation
Semantics-based Manifold Alignment
Simultaneous Decoding and Translation
A
B
Fake B
Recon. B
🟩 Training-only

Figure 1: DSM framework. Two image domains are aligned along their data-space PCs via a shared latent space before translation.

mainly to encourage good behaviors of the latent distribution but not to match it with the data distribution. As a result, current encoders may not to be able to preserve the distributional semantics, shown in the experiments. We, therefore, introduce a new general autoencoding scheme to preserve the distributional semantics:

$$z_x = \mathbf{e}(x), x' = \mathbf{d}(z_x)$$
$$\text{subject to } Ux = Vz_x \quad U, V \text{ have } K \text{ rows.} \tag{1}$$

where x is a data sample, $\mathbf{e}$ and $\mathbf{d}$ are some encoding and decoding schemes. U and V are the first K PCs of $\mathbf{A}$ and z_x. The autoencoder can be trained by *e.g.* minimizing $\sum ||x - x'||_2^2$. Equation 1 states the key difference between our autoencoder and existing autoencoders: it requires the projections of x on the data-space PCs U are equal to the projections of z_x onto the latent-space PCs V. Note that a standard dimensionality reduction using PCA is a special case of Equation 1, when $V = U$ and $\mathbf{e}(\mathbf{A}) = V\mathbf{A}$. But Equation 1 is more general because it does not dictate what V is, nor does it require the latent space to have fewer dimensions than the data space. Equation 1 only requires the covariance structure to persist during encoding along the first K PCs in both the data and latent space. One key question is why not just set $V = U$. This is because we need the flexibility of encoding data into an arbitrary V during UIT while keeping the general shape of the data distribution, as explained later.

Equation 1 is general but difficult to optimize because it needs to be computed on the whole dataset, requiring a high memory consumption, and V is unknown. Therefore, we propose a local scheme which keeps the global alignment by enforcing local alignments on samples:

$$l_{localAlign} = \frac{1}{N}\sum_{i=0}^{N-2}(l_{angle} + l_{norm})$$
$$l_{angle} = |\cos(x_i, x_{i+1}) - \cos(z_{x_i}, z_{x_{i+1}})|$$
$$l_{norm} = \frac{||x_i||_2}{||x_{i+1}||_2} - \frac{||z_{x_i}||_2}{||z_{x_{i+1}}||_2} \tag{2}$$

where $\cos()$ is the cosine distance, N is the batch size and $z_{x_i} = \mathbf{e}(x_i)$. $l_{localAlign}$ dictates that for any two data samples, their length ratio and angle should remain the same after they are projected into the latent space. l_{angle} tends to keep the overall shape of the distribution when the data is projected into a latent space, while l_{norm} allows scaling but prefers uniform scaling. $l_{localAlign}$ has an effect of preserving the covariance structure of data during transformations. Further, its locality (only considering two samples a time) essentially makes the covariance structure invariant under homogeneous transformations of the basis in the latent space, letting the autoencoder automatically decide the best V that can help reconstruction but also keep the covariance structure.

3.2. Semantics-based Manifold Alignment

Given $\mathbf{z_A} = \mathbf{e_A}(\mathbf{A})$ and $\mathbf{z_B} = \mathbf{e_B}(\mathbf{B})$, we have two latent distributions with their respective covariance structure characterized by their latent space PCs V_A and V_B. To make sure the UIT maintains the distributional semantics, we need to align $\mathbf{z_A}$ and $\mathbf{z_B}$, *e.g.* aligning the direction in which $\mathbf{z_A}$ shows the biggest variation with that of $\mathbf{z_B}$ by aligning their first PCs. Further, the visual semantics can manifest on multiple dimensions, corresponding to aligning the top K PCs in V_A and V_B. There are several alternative methods such as aligning V_A and V_B directly, or fix one and try to align the other to it. Since both V_A and V_B are unknown, directly aligning V_A and V_B corresponds to:

$$\text{minimize } l_{latentA} + l_{latentB} + l_{align}$$
$$l_{latentA} = \frac{1}{N_A}\sum_{i=0}^{N_A-1} ||z_{x_i} - V_A V_A^T z_{x_i}||_2^2$$
$$l_{latentB} = \frac{1}{N_B}\sum_{i=0}^{N_B-1} ||z_{y_j} - V_B V_B^T z_{y_j}||_2^2$$
$$l_{align} = \frac{1}{K}\sum_{k=0}^{K-1} ||V_A^k - V_B^k||_2^2 \tag{3}$$

where N_A and N_B are the number of images in $\mathbf{A}$ and $\mathbf{B}$. V_A^k and V_B^k are the kth PCs of V_A and V_B. After experiments, we find that allowing V_A and V_B to simultaneously change makes the optimization suffer from sub-optimal local minima. Therefore, we fix V_A but let V_B to align with

V_A, which also means that $\mathbf{e_A}$ can be pre-trained and we can compute V_A from $\mathbf{z_A}$ via PCA.

Next, aligning V_B to V_A still presents challenges because directly learning V_B requires to simultaneously transform all z_ys which is again equivalent to operating on the whole B. This is a similar difficulty to the one in Section 3.1. Again, we propose to operate only on a batch of N samples to ensure the global alignment of V_A and V_B:

$$l_{maniAlign} = \frac{1}{N}\sum_{i=0}^{N-1} ||z_{y_i} - V_A V_A^T z_{y_i}||_2^2 + ke^{-\alpha||z_{y_i}||_2^2} \tag{4}$$

where $y_i \in \mathbf{B}$ and V_A contains the first K PCs of $\mathbf{z_A}$ in interest. $l_{maniAlign}$ requires z_y to be reconstructable after projecting them into the basis of $\mathbf{z_A}$, which essentially encourages the covariance structure of $\mathbf{z_B}$ to be similar to that of $\mathbf{z_A}$ and the two bases to be aligned. $ke^{-\alpha||z_{y_i}||_2^2}$ is to prevent z_{y_i} from shrinking, which is a trivial solution of Equation 4. Since the covariance structure of $\mathbf{A}$ and $\mathbf{B}$ and kept in $\mathbf{z_A}$ and $\mathbf{z_B}$ via Equation 2, $l_{maniAlign}$ finishes the semantics-based alignment of two domains.

3.3. Simultaneous Decoding and Translation

After semantics-based manifold alignment, we transform $\mathbf{z_A}$ and $\mathbf{z_B}$ into the space of $\mathbf{B}$ to finish the UIT. Since $\mathbf{z_A}$ and $\mathbf{z_B}$ are aligned, we combine the reconstruction and translation tasks together by using one network $\mathbf{g}$ which serves both as a decoder and a translator. This is because the general shapes of the two latent distributions are similar after alignment. When the decoder is trained by the reconstruction loss on $\mathbf{z_B}$, it already to some extent learns to translate $\mathbf{z_A}$. We reconstruct $\mathbf{B}$ by $y' = \mathbf{g}(z_y)$. Meanwhile, we treat $\mathbf{g}$ as a generator in a Generative Adversarial Network by $\hat{y} = \mathbf{g}(z_x)$ and use a discriminator network $\mathbf{h}(\hat{y}, y) = [0, 1]$ to further improve the translation.

Overall, given a pre-trained $\mathbf{e_A}$ and $\mathbf{z_A}$, hence also V_A, we minimize the following objective function:

$$\begin{aligned} l =& \omega_1 \frac{1}{N_B}\sum_{i=1}^{N_B-1} ||y_i - y_i'||_2^2 + \omega_2 l_g + \omega_3 l_d \\ &+ \omega_4 l^B_{localAlign} + (1 - \sum_{i=1}^{4}\omega_i) l_{maniAlign} \end{aligned} \tag{5}$$

where l_g and l_d are the GAN loss that depends on the chosen GAN model. N_B is the total number of images in $\mathbf{B}$ and ωs are weights. In $l^B_{localAlign}$, we apply Equation 2 to both the y-to-z_y, and z_x-to-$\hat{y}$ mapping. All details are in the appendix for the sake of simplicity.

4. Implementation Details

We pre-train an autoencoder in dataset A with $l_{localAlign}$ to get $\mathbf{e_A}$ and calculate V_A from $\mathbf{z_A}$ using PCA. Next, for $\mathbf{e_B}$ and $\mathbf{g}$, we adopt the network architectures from UNIT. In all experiments, we set $\omega_1 = 0.033, \omega_2 = \omega_3 = 0.333$ and other weights depend on the experiment. Please refer to the appendix for details. $\mathbf{e_A}$ is trained for 100 epochs and the rest is trained for 300 epochs on all datasets, using Adam [18] with a batch size of 16, a learning rate of 0.0001, and exponential decay rates $(\beta_1; \beta_2) = (0.5; 0.999)$. All the experiments are conducted on 2 NVIDIA 1080 Ti GPUs, implemented in PyTorch. The training takes between 6 to 18 hours.

5. Experiments

5.1. Data

We employ several benchmark datasets to extensively validate our method, including SummerWinter [39], CatDog [23] and ShoeHandbag [37, 38]. Moreover, we build a very challenging dataset named MMISTHandbag by using hand-drawn digits from MNIST [20] and handbags randomly sampled from [38], which has two distinctive distributions and distributional semantics. All results shown are computed using LSGAN. Please refer to the appendix for details.

5.2. Evaluation Metrics

In addition to visual evaluation, we employ the Frechet Inception Distance (FID) [10] as a quantitative measure. However, there is no good metric to measure the faithfulness of the preservation of distributional semantics. We, therefore, propose a new evaluation metric called Ordering-Tolerance curve (OTC). Given images $x \in \mathbf{A}$ and their translations $\hat{y} \in \mathbf{B}$, we can define the OTC as:

$$\begin{aligned} c =& \frac{1}{N_A}\mathbf{card}(\{x|\frac{d(x,\hat{y})}{N_A} \le \beta, x \in \mathbf{A}\}),\ \beta \in [0,1] \\ & d(x,\hat{y}) = |rank(\hat{y}) - rank(x)| \end{aligned} \tag{6}$$

where $\mathbf{card}(\cdot)$ is the cardinality of a set, $rank(x)$ is the rank of x in $\mathbf{A}$ along a chosen PC among all data samples in $\mathbf{A}$ and N_A is the total number of data samples in $\mathbf{A}$. $rank(\hat{y})$ is the rank of $\hat{y}$ along a chosen PC in $\mathbf{B}$. $d(x, \hat{y})$ is equal to zero if the rank of x is kept during translation, and non-zero otherwise (the larger the worse). c is the *Percentage of Correct Ordering* of xs whose normalized rank errors are within β, the *Ordering Error Tolerance*.

5.3. Semantics-preserving Transformation

A key contribution of our work is a straightforward but extremely effective transformation scheme that preserves

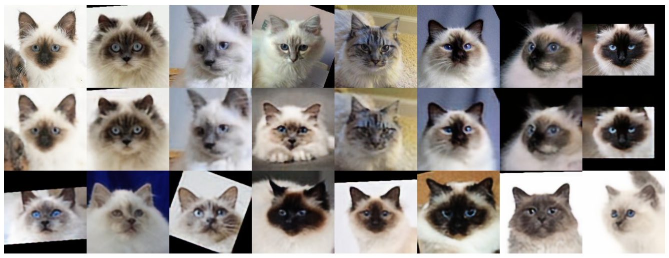
Figure 2: Top: original data. Mid: latent samples of DSM. Bottom: latent samples of a standard autoencoder. From left to right: the images ranked the 4th, 19th,24th,37th,44th,52th,94th and 97th on the first PCs of the data (top) and latent space (mid and bottom).

the distributional semantics. To show the necessity of such transformations in UIT, we train an autoencoder on cat images in CatDog then compute the 1st PCs of A and $\mathbf{z_A}$. The autoencoder is based on the encoder and decoder of UNIT. Please refer to the appendix for details. We then rank all the images along the two PCs in Figure 2.

In the original data, the variation on the 1st PC is mainly the color transition from light to dark (Figure 2: Top). However, without semantics-preserving transformations, the shape of the latent distribution is already changed, unable to preserve the visual semantics (Figure 2: Bottom). In contrast, DSM preserves the distributional semantics (Figure 2: Mid). We also show the OTCs in Figure 3 Left where DSM can contain the rank error within 3% while a standard autoencoder fails systematically. Although we only show the results from a specific autoencoder, this is a common problem of autoencoders shown by preliminary experiments.

5.4. Ablation Studies

Two main adjustable components of DSM are the GAN architecture and the number of PCs, K. We first evaluate DSM on the first K PCs on all datasets and present the FID scores in Table 1. Different from our expectation that a larger K will result in a harder optimization problem and hence lower quality, the results show that it depends on the dataset. After looking into the data, it is understandable because different datasets have different amounts of variances distributed on PCs, *e.g.* some have large variances on the 1st PC while others have more even distributions of variances on the first K PCs, which affects the behavior of Equation 4. Although control can be added, *e.g.* by adding weights on different PCs, we choose not to do so and let DSM adapt to the data. Also, we only test DSM up to $K = 3$. This is mainly because although DSM can work for $K \geq 3$ numerically, it is usually hard for humans to visually understand the semantics in PCs where $K \geq 3$. We also compare two GAN architectures LSGAN and NSGAN on ShoeHandbags

	$K = 0$	$K = 1$	$K = 2$	$K = 3$
CatDog	64.5159	**58.0004**	96.1909	84.4792
SummerWinter	100.9174	**90.4108**	107.5985	99.4951
ShoeHandbag	**123.6028**	129.7010	155.5799	128.9304
MNISTHandbag	172.6736	180.5911	**149.3162**	171.7602

Table 1: FID scores on four datsets with the first K PCs aligned. $K = 0$ means DSM is trained without alignment loss.

and show the OTCs in Figure 3 Mid. Both can keep the rank error within around 20%, showing that alignment constraints can improve translation ability visibly on different GAN architectures. More results can be found in the appendix.

5.5. Image Quality in Translation

Although normally the first K PCs bear visually understandable semantics, the specific K however depends on the dataset. The first PC is almost always visually interpretable across datasets and some datasets have meaningful variations on other PCs. We show the images from the four datasets ranked along their respective meaningful PCs in Figure 4. For CatDog, SummerWinter and ShoeHandbags, the first PC (PC0) shows color variations from light to dark; for MINST, it is mainly the shape varying from slim to round; for ShoeHandbags, the second PC (PC1) is the shoe collar height variation for shoes and handle length variation for handbags.

In Figure 5, we show our results in mappings of cat-to-dog, summer-to-winter, shoes-to-handbags, and digits-to-handbags. The PC0s of CatDog and SummerWinter are color variations (light to dark). The major difference is the color variations in CatDog are clearly separated into foreground (faces) and background while there is no such separation in SummerWinter. In both cases, DSM successfully translates the images with high quality while simultaneously matching the semantics. In cat-to-dog, DSM transforms the faces and maintains the separation of the foreground and background, while in summer-to-winter, the changes are more heterogeneous depending on the scenes where DSM lays snow on different landscapes. In shoes-to-handbags, unlike CatDog and SummerWinter, the color variation is restricted to the object itself, where DSM faithfully keeps the semantics. Finally, to push DSM further, we test a digit-to-handbags translation. These two datasets have distinctive distributions. The results show that long slim digits are translated to handbags in light colors while fat round digits are translated to handbags in dark colors, which are consistent with Figure 4 along their respective PC0s. We also show the OTCs in Figure 3 Right.

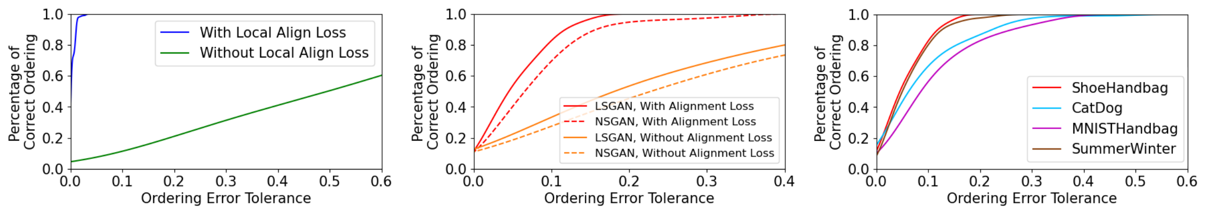

Figure 3: Ablation Studies. Left: OTC along PC0 with/without semantics-perserving transformation on CatDog. Mid: LSGAN vs NSGAN along PC0 on ShoeHandbags. Right: OTC along PC0 on four datasets.

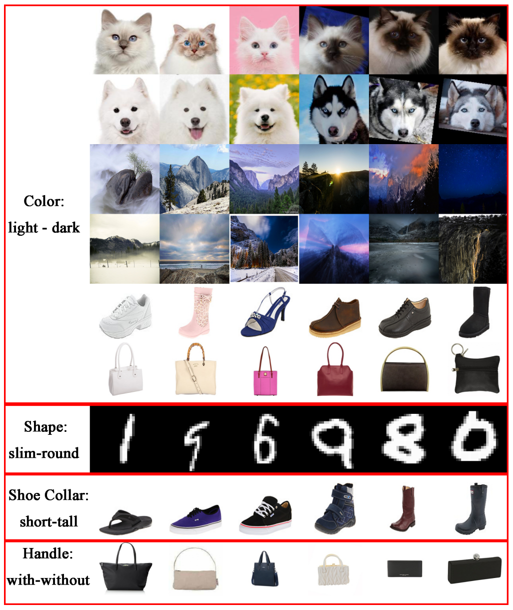

Figure 4: Top to bottom: CatDog(PC0), SummerWinter(PC0), ShoeHandbags(PC0), MNIST(PC0), and ShoeHandbags(PC1).

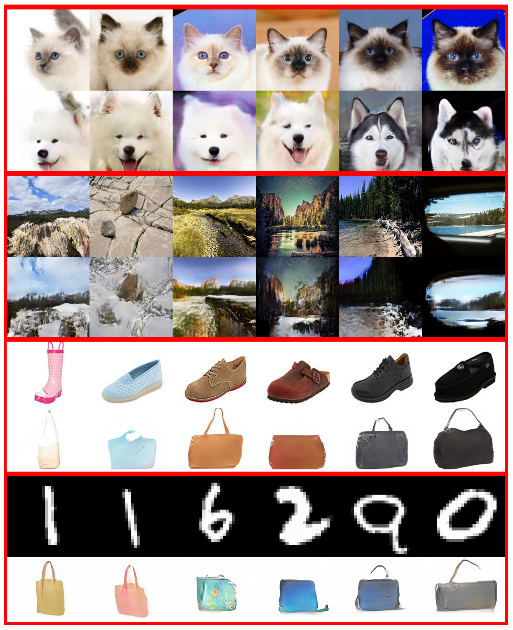

Figure 5: Top to bottom: CatDog, SummerWinter, ShoeHandbag and MNISTHandbag, ordered along PC0. In each group, the first row is the input images and the second is the translated images.

Method/Data	CatDog	ShoeHandbag
Ours	**58.0004**	**128.9304**
CycleGAN	95.1197	135.5504
DistanceGAN	63.6016	185.2788
DRIT++	61.9530	185.0121
UNIT	64.6241	140.8820

Table 2: FID scores on CatDog and ShoeHandbag.

5.6. Comparisons

We compare our model with UNIT, CycleGAN, DistanceGAN and DRIT++ on CatDog and ShoeHandbag, by using the official code shared by the authors of the baseline methods. We first show the FID scores of all methods in Table 2 and the OTCs in Figure 6. By aligning two data manifolds based on their semantics, DSM is able to improve the translation quality, which is observed on both datasets. The OTCs show clearly that DSM can keep the semantics on PC0 better than others by containing the rank error within around 30% and 18% (Figure 6 Left and Mid). The second best methods come with roughly only 50% and 40%. On PC1 of ShoeHandbag (Figure 6 Right), CycleGAN is close to DSM. However, we argue its behavior is inconsistent, especially considering Figure 6 Left and Mid where the variance is large and contains large amounts of semantics information.

Visual comparisons can be found in Figure 7. Overall, DSM generates images with higher visual qualities. Further, the distributional semantics cannot be preserved and matched by other methods. In Figure 7 Left, the major variation of the input images (left-to-right) is the color variation (light-to-dark). While DSM obviously keeps the same variations during UIT, it is hard to find similar variations

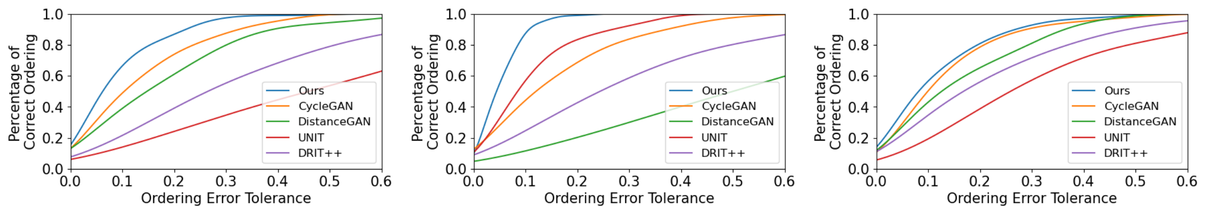

Figure 6: Left: CatDog. Mid: Shoe-to-Handbags on PC0 of Handbags. Right: Shoe-to-Handbags on PC1 of Handbags.

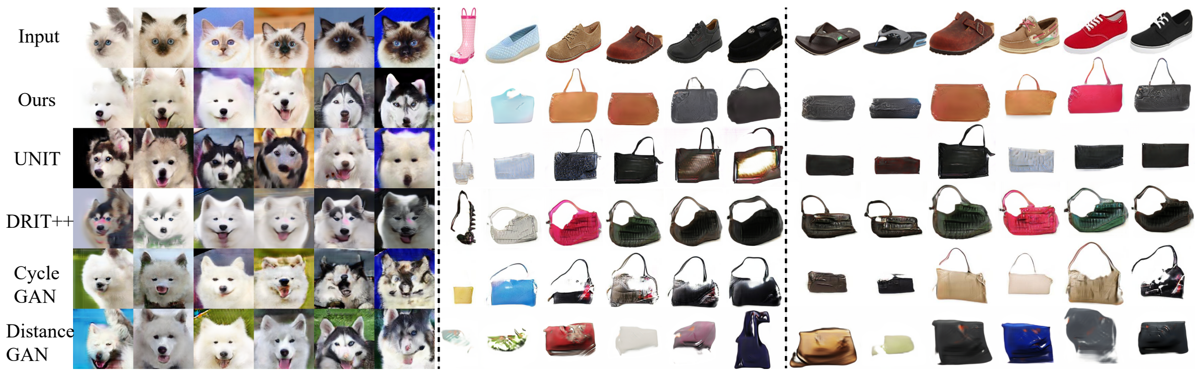

Figure 7: Left and Mid: ordered along PC0 of Dogs and Handbags. Right: ordered along PC1 of Handbags.

in other methods. Similar results can also be found in Figure 7 Mid. To further explore semantics in other PCs, we show Figure 7 Right. While the input varies from slippers to sneakers, our results generates handbags varying from without handles to with handles. In contrast, other methods struggle to generate consistent semantics variations. More results can be found in appendix.

6. Discussions and Conclusion

Although PCA is a straightforward choice to characterize the distributional semantics, our method can incorporate other methods such as Kernel PCA, Multidimensional Scaling, etc. which can characterize the covariance structure via 'flattening' the data manifold after which DSM can be performed.

We only evaluated DSM up to $K = 3$ because the semantics start to become not visually obvious after $K \geq 3$. However, we argue that DSM is effective and useful for two reasons. First, for almost all datasets, the first PC bears the majority of the variances, and the distributional semantics captured by the variance are always visually interpretable. Second, aligning the PCs of the two distributions improves the image quality during translation.

In summary, we proposed the first UIT method DSM which preserves and matches the distributional semantics of two image domains. It is straightforward and effective, proven on multiple datasets, and capable of improving translation quality compared with many state-of-the-art methods. DSM is also general in its capacity of incorporating any GANs and autoencoder models. In future, we will incorporate human-labelling into a semi-supervised setting of DSM where humans can arbitrarily decide the semantics by ranking images. This will enable DSM to encode arbitrary semantics and open it to many other applications.

References

[1] A. Almahairi, S. Rajeswar, A. Sordoni, P. Bachman, and A. C. Courville. Augmented cyclegan: Learning many-to-many mappings from unpaired data. In *ICML 2018*, volume 80, pages 195–204, 2018. 1

[2] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *ICML 2017*, volume 70, pages 214–223, 2017. 2

[3] S. Benaim and L. Wolf. One-sided unsupervised domain mapping. In *NIPS 2017*, pages 752–762, 2017. 1, 2

[4] S. Chen, F. Liu, Y. Lai, P. L. Rosin, C. Li, H. Fu, and L. Gao. Deepfaceediting: Deep face generation and editing with disentangled geometry and appearance control. *CoRR*, abs/2105.08935, 2021. 2

[5] Y. Choi, M. Choi, M. Kim, J. Ha, S. Kim, and J. Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR 2018*, pages 8789–8797, 2018. 2

[6] Y. Choi, Y. Uh, J. Yoo, and J. Ha. Stargan v2: Diverse image synthesis for multiple domains. In *CVPR 2020*, pages 8185–8194, 2020. 2

[7] H. Fu, M. Gong, C. Wang, K. Batmanghelich, K. Zhang, and D. Tao. Geometry-consistent generative adversarial networks for one-sided unsupervised domain mapping. In *CVPR 2019*, pages 2427–2436, 2019. 2

[8] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS 2014*, pages 2672–2680, 2014. 2

[9] E. Härkönen, A. Hertzmann, J. Lehtinen, and S. Paris. Ganspace: Discovering interpretable GAN controls. *CoRR*, abs/2004.02546, 2020. 2

[10] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, pages 6626–6637. Curran Associates, Inc., 2017. 4

[11] X. Huang, M. Liu, S. J. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. In *ECCV 2018*, volume 11207, pages 179–196, 2018. 2

[12] S. Iizuka, E. Simo-Serra, and H. Ishikawa. Globally and locally consistent image completion. *ACM Trans. Graph.*, 36(4):107:1–107:14, 2017. 2

[13] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. 9

[14] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR 2017*, pages 5967–5976, 2017. 2

[15] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *CVPR 2019*, pages 4401–4410, 2019. 2, 11

[16] J. Kim, M. Kim, H. Kang, and K. Lee. U-GAT-IT: unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation. In *ICLR 2020*, 2020. 2

[17] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim. Learning to discover cross-domain relations with generative adversarial networks. In *ICML 2017*, volume 70, pages 1857–1865, 2017. 2

[18] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR 2015*, 2015. 4

[19] D. P. Kingma and M. Welling. Auto-encoding variational bayes, 2014. 2

[20] Y. LeCun, C. Cortes, and C. J. C. Burges. The mnist database of handwritten digits, 1998. http://yann.lecun.com/exdb/mnist/. 4

[21] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR 2017*, pages 105–114, 2017. 2

[22] C.-H. Lee, Z. Liu, L. Wu, and P. Luo. Maskgan: Towards diverse and interactive facial image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[23] H. Lee, H. Tseng, J. Huang, M. Singh, and M. Yang. Diverse image-to-image translation via disentangled representations. In *ECCV 2018*, volume 11205, pages 36–52, 2018. 2, 4

[24] H. Lee, H. Tseng, Q. Mao, J. Huang, Y. Lu, M. Singh, and M. Yang. DRIT++: diverse image-to-image translation via disentangled representations. *Int. J. Comput. Vis.*, 128(10):2402–2417, 2020. 2

[25] M. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In *NIPS 2017*, pages 700–708, 2017. 1, 2, 9

[26] M.-Y. Liu, X. Huang, A. Mallya, T. Karras, T. Aila, J. Lehtinen, and J. Kautz. Few-shot unsupervised image-to-image translation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 2

[27] G. Lu, Z. Zhou, J. Shen, C. Chen, W. Zhang, and Y. Yu. Large-scale optimal transport via adversarial training with cycle-consistency. *CoRR*, abs/2003.06635, 2020. 2

[28] G. Lu, Z. Zhou, Y. Song, K. Ren, and Y. Yu. Guiding the one-to-one mapping in cyclegan via optimal transport. *CoRR*, abs/1811.06284, 2018. 2

[29] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley. Least squares generative adversarial networks. In *ICCV 2017*, pages 2813–2821, 2017. 2

[30] Y. A. Mejjati, C. Richardt, J. Tompkin, D. Cosker, and K. I. Kim. Unsupervised attention-guided image-to-image translation. In *NIPS 2018*, pages 3697–3707, 2018. 2

[31] M. Mirza and S. Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014. 2

[32] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. *CoRR*, abs/1802.05957, 2018. 9

[33] M. Tomei, M. Cornia, L. Baraldi, and R. Cucchiara. Art2real: Unfolding the reality of artworks via semantically-aware image-to-image translation. In *CVPR 2019*, pages 5849–5859, 2019. 1

[34] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *CoRR*, abs/1607.08022, 2016. 9

[35] T. Wang, M. Liu, J. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR 2018*, pages 8798–8807, 2018. 2

[36] Z. Yi, H. R. Zhang, P. Tan, and M. Gong. Dualgan: Unsupervised dual learning for image-to-image translation. In *ICCV 2017*, pages 2868–2876, 2017. 2

[37] A. Yu and K. Grauman. Fine-grained visual comparisons with local learning. In *CVPR 2014*, pages 192–199, 2014. 4

[38] J. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros. Generative visual manipulation on the natural image manifold. In *ECCV 2016*, volume 9909, pages 597–613, 2016. 4

[39] J. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV 2017*, pages 2242–2251, 2017. 1, 2, 4

Appendix

A. Implementation Details

Loss functions. In Equation 5 of the paper, we have a local alignment loss $l^{B}_{localAlign}$, which applies Equation 2 to both y-to-z_y, and z_x-to-$\hat{y}$. Applying Equation 2 to y-to-z_y is straightforward, which ensures semantics-preserving transformation between $\mathbf{B}$ and $\mathbf{z_B}$, just as the encoder of $\mathbf{A}$. Here we explain why applying Equation 2 to z_x-to-$\hat{y}$ is essential. As mentioned in Sec. 3.3 of the paper, we have aligned $\mathbf{z_A}$ and $\mathbf{z_B}$, and use a single network $\mathbf{g}$ serving as both the decoder for $\mathbf{z_B}$ and the generator for $\mathbf{z_A}$. Such mechanism provides an implicit constraint on the translated images $\hat{B}$ from $\mathbf{z_A}$ so that it has a similar distribution as the reconstructed images B' from $\mathbf{z_B}$. However, as there is an additional GAN loss which will modify the distribution of translated images, the alignment between $\hat{B}$ and B' could be affected, which will compromise the distributional semantics matching. To this end, we apply Equation 2 to z_x-to-$\hat{y}$ to explicitly preserve the semantics.

$$\omega_4 l^{B}_{localAlign} = \omega_4^{\mathbf{e}} l^{\mathbf{e}}_{localAlign} + \omega_4^{\mathbf{g}} l^{\mathbf{g}}_{localAlign} \tag{7}$$

where $l^{\mathbf{e}}_{localAlign}$ and $l^{\mathbf{g}}_{localAlign}$ are the loss terms of y-to-z_y and z_x-to-$\hat{y}$. In all experiments, we set $\omega_1 = 0.033, \omega_2 = \omega_3 = 0.333, \omega_4^{\mathbf{g}} = 0.167$ in Equation 5 of the paper. We set $\omega_4^{\mathbf{e}} = 0.066$ in ShoeHandbag align 3PCs experiments and $\omega_4^{\mathbf{e}} = 0.05$ in all else experiments. We set $k = 15, \alpha = 10^{-6}$ for the regularization term in Equation 4 of the paper.

Network Architecture. Our image translation network architecture is based on the one from UNIT [25]. For the encoder, to ensure the latent vector size is smaller than the input image size in the case of resolution $256 * 256$, we halve the kernel numbers in the second and third convolutional layers and add one more convolutional layer (see Table 3 for details). Besides, instead of using instance normalization (IN) [34], we utilize batch normalization (BN) [13] in the encoder and generator. For the discriminator network, we employ spectral normalization [32], and multi-scale discriminators at 3 scales. The network architecture is illustrated in Table 3. We use the following abbreviation for ease of presentation: N=Kernel number, K=Kernel size, S=Stride size. The 2 nearest-neighbor upsampling layer is denoted as UP and the residual basic block is denoted as RESBLK. For the pre-trained autoencoder for $\mathbf{A}$, the detailed network architecture is listed in Table 4.

B. More Experiment Results

B.1. Data Details

For all datasets, the images are resized to 256x256. In Catdog, 871 cat (birman) images, 1364 dog (husky, samoyed) images are randomly divided into 771 (cat)

Layer	Encoder	Generator	Discriminator
1	CONV(N64,K3,S1),RELU	CONV(N512,K3,S1),RELU	CONV(N64,K4,S2),LeakyReLU
2	CONV(N64,K3,S2),RELU	RESBLK(N512,K3,S1)	CONV(N128,K4,S2),LeakyReLU
3	CONV(N128,K3,S2),RELU	RESBLK(N512,K3,S1)	CONV(N256,K4,S2),LeakyReLU
4	CONV(N128,K3,S2),RELU	RESBLK(N512,K3,S1)	CONV(N512,K4,S2),LeakyReLU
5	RESBLK(N128,K3,S1)	RESBLK(N512,K3,S1)	CONV(N512,K1,S1),LeakyReLU
6	RESBLK(N128,K3,S1)	UP+CONV(N256,K3,S1),RELU	
7	RESBLK(N128,K3,S1)	UP+CONV(N128,K3,S1),RELU	
8	RESBLK(N128,K3,S1)	UP+CONV(N64,K3,S1),RELU	
9		CONV(N3,K7,S1),TanH	

Table 3: Network architecture for the distributional semantics mapping experiments

Layer	Encoder	Decoder
1	CONV(N64,K3,S1),RELU	RESBLK(N128,K3,S1)
2	CONV(N64,K3,S2),RELU	RESBLK(N128,K3,S1)
3	CONV(N128,K3,S2),RELU	RESBLK(N128,K3,S1)
4	CONV(N128,K3,K2),RELU	RESBLK(N128,K3,S1)
5	RESBLK(N128,K3,S1)	UP+CONV(N128,K3,S1),RELU
6	RESBLK(N128,K3,S1)	UP+CONV(N128,K3,S1),RELU
7	RESBLK(N128,K3,S1)	UP+CONV(N64,K3,S1),RELU
8	RESBLK(N128,K3,S1)	CONV(N3,K7,S1),TanH

Table 4: Network architecture for the autoencoder in ablation study

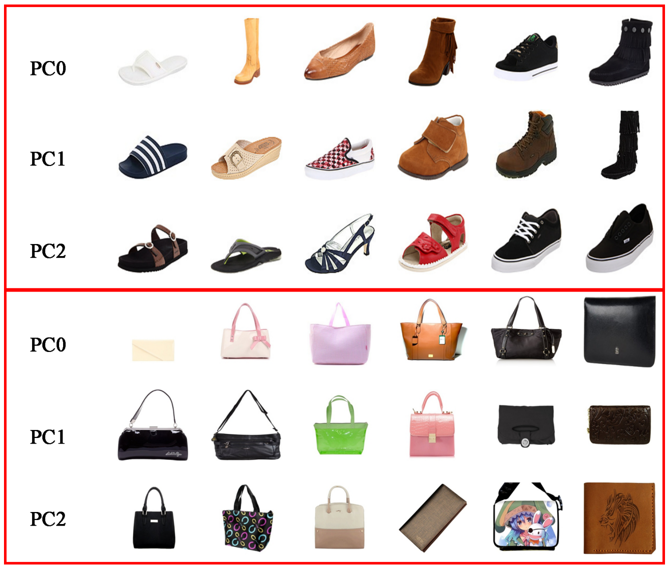

Figure 8: Top: Shoe dataset. Bottom: Handbag dataset. From left to right: the images ordered along the corresponding PC

and 1264 (dog) for training and the remaining for testing. For SummerWinter, it consists of 1540 Summer Photos and 1200 Winter photos, randomly divided into 1231 (summer), 962 (winter) for training and the rest for testing. For ShoeHandbag, we randomly sample images from edges2shoes and edges2handbags, with 3726 (shoe), 3822 (handbag) for training and 101 (shoe), 178 (handbag) for testing. For MNISTHandbag: 1600 MNIST images and 1600 handbag images are randomly selected from MNIST and edges2handbags, with 1500 of each for training and 100 for testing. We show the images from ShoeHandbag datasets along the first 3 PCs in Figure 8. Please note that the MNISTHandbag dataset is very challenging, because the distributions of MNIST and handbags are very different. The top five variance ratios along PCs in MNIST are [0.095, 0.071, 0.066, 0.052, 0.047], while in handbags such ratios are [0.274, 0.115, 0.065, 0.054, 0.026].

B.2. Our OTCs on CatDog, MNISTHandbag and SummerWinter

Figure 10 shows the OTCs of our method on the CatDog, MNISTHandbag and SummerWinter datasets, with the first 2 and 3 PCs aligned respectively. We can see that our method always keeps the semantics best on PC0, and the rank errors become larger on PC1 and PC2. We find that it is mainly because the variance ratio along PC0 is always much larger than those along other PCs. For example, in the Handbag dataset, the ratios of variances along the first 3 PCs are around 4:2:1. As a result, the network prefers to perform the alignment along PC0 as priority to minimize the total loss. We also notice that the performance of semantics preserving varies largely across different datasets. In the challenging MNISTHandbag dataset which has two dis-

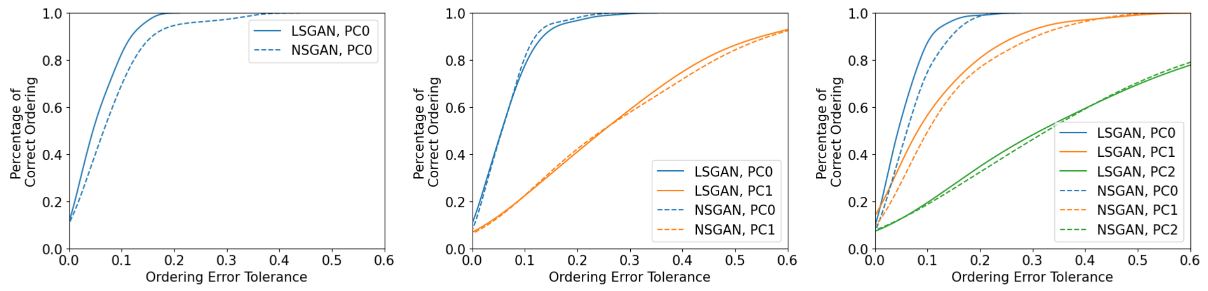

Figure 9: LSGAN/NSGAN

tinctive distributions, our method preserves semantics well along PC0, while in the SummerWinter dataset, our method is capable of preserving the semantics on the first 3 PCs. Although control can be put e.g. via weighting to enforce the alignment of multiple PCs, we choose not to do so and make DSM adapt to data, as the distribution of variances on different PCs is an intrinsic property of the data itself and should be respected during translation.

We evaluate the semantics preservation only on the first 3 PCs mainly because of two reasons. First, in most image datasets, compared to the variances on the first 3 PCs, the variances on the remaining PCs are very small. For example, in the Summer dataset, even the sum of variances on the 4th to 20th PCs is smaller that the variance of PC0. Enforcing alignment along these directions with very small variations does not contribute to the explicability of the mapping while adding more complexity to the optimization. Second, after investigating the popular datasets (e.g. Shoes, Handbags, Cars, Animals, Faces, Art works), we find that in all datasets people can easily perceive semantics on the first PC, but can only perceive semantics on the second PC in the Shoe, Handbags and MNIST. People cannot perceive any semantics on the 4th and subsequent PCs. Hence we focus on the first 3 PCs.

B.3. Ablation Study on LSGAN and NSGAN

	$K = 1$	$K = 2$	$K = 3$
LSGAN	129.7010	155.5799	**128.9304**
NSGAN	172.8317	186.9708	**154.8572**

Table 5: FID scores on ShoeHandbag with the first K PCs aligned

We demonstrate the results of using LSGAN and using NSGAN in Figure 9 and Table 5, which give the image translation results on the ShoeHandbag dataset by aligning the first 1PC, 2 PCs and 3 PCs respectively. From the OTCs of LSGAN and NSGAN, we can see that the two GAN models have very similar performance on semantics preserving in all cases, which validates that our method does not rely on a specific GAN model and can ensure semantics preserving with different GAN models. We also notice that the FIDs of NSGAN are higher than that of LSGAN. It is mainly because that the inherent image generation capability of NSGAN is weaker than LSGAN. Employing other GAN models such as StyleGAN [15] can improve the FID scores.

B.4. Alignment of PCs

Equation 4 in the paper dictates that the first K PCs of two domains need to be aligned. However, it does not specify the order of alignment. In other words, it does not specify if PC1 of the first domain needs to be aligned with PC1 of the second domain. We have two choices. The first one is to enforce the order, PC0-to-PC0, PC1-to-PC1, and so on. However, we find it sub-optimal in the sense that Equation 4 is affected by the amounts of variances distributed on PCs. For a dataset with its majority variances distributed on PC0, the alignment forces by Equation 4 for its PC1 and subsequent PCs are small. We argue that they should be small as the dataset has less explicable variances on PC1 and higher PCs. Hence, the major force by Equation 4 should naturally change according to the variance distribution. We therefore do not enforce the order of alignment of PCs for two datasets. As a result, although PC0-to-PC0 is always ensured in all experiments, sometimes PC1 of a dataset can be mapped to PC2 of the other. However, we argue that the mapping is still valid because it is also explicable and reflects the distributional semantics.

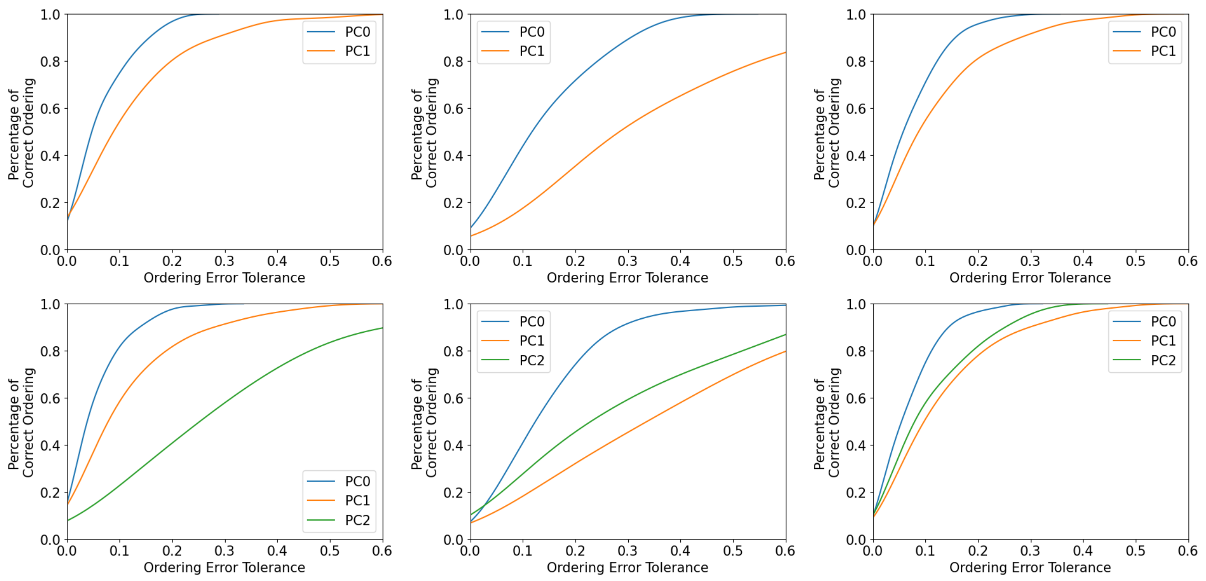

Figure 10: Top: Aligning the PC0 and PC1; Bottom: Aligning PC0-2; Left to Right: CatDog, MNISTHandbag and Summer-Winter