Computational Visual Media
DOI 10.1007/s41095-xxx-xxxx-x Vol. x, No. x, month year, xx–xx

Research Article

Unsupervised image translation with distributional semantics awareness

Zhexi Peng[1], **He Wang**[2], **Yanlin Weng**[1](✉), Yin Yang[3] and Tianjia Shao[1]

Abstract Unsupervised image translation (UIT) studies the mapping between two image domains. Since such mappings are under-constrained, existing research has pursued various desirable properties such as distributional matching or two-way consistency. In this paper, we re-examine 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 style, functionality, *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 *distributional semantics mapping* (DSM), the first UIT method which explicitly matches semantics between two domains. We show that distributional semantics has been rarely considered within and beyond UIT, even though it is a common problem in deep learning. We evaluate DSM on several benchmark datasets, demonstrating its general ability to capture distributional semantics. Extensive comparisons show that DSM not only produces explicable mappings, but also improves image quality in general.

Keywords generative adversarial networks; manifold alignment; unsupervised learning; image-to-image translation; distributional semantics.

1 State Key Lab of CAD&CG, Zhejiang University, Hangzhou, 310058, China. E-mail: Z. Peng, zhexipeng@zju.edu.cn; Y. Weng, weng@cad.zju.edu.cn(✉); T. Shao, tjshao@zju.edu.cn.

2 School of Computing, University of Leeds, Leeds, UK. E-mail: H.E.Wang@leeds.ac.uk.

3 School of Computing, Clemson University, Clemson, USA. E-mail: yin5@clemson.edu.

Manuscript received: 2014-12-31; accepted: 2015-01-30.

1 Introduction

Unsupervised image translation (UIT) has been intensively studied in recent years. Many applications have been inspired by its ability to create mappings between two image domains. Since there can be theoretically an infinite number of mappings between two domains, UIT is by nature an under-constrained problem. Naturally, different approaches have been developed to ensure certain desirable properties, such as shared latent spaces [26], two-way consistency [40], pair-wise distance preservation [3], and image semantics [34]. 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 between samples (not within a single sample). Shoes vary in color, style (*e.g.* low/high collars), and functionality (*e.g.* sneakers/high heels). Similarly, bags also vary in color, style (*e.g.* with/without handles), and functionality (*e.g.* purses/backpacks). During UIT, we argue that it is not enough to simply translate images. It is desirable for such distributional semantics to be maintained, *e.g.* red high-collar high heels map to black purses with handles, while white low-collar sneakers map to blue backpacks. This is exactly the goal of this research.

To maintain distributional semantics during UIT, two critical problems must be addressed. The first question is what semantics should be maintained. As we are considering unsupervised learning, no labeling should be required, which means that the data variations (the distributional semantics) should be characterized without prior knowledge, yet be interpretable by humans. Secondly, distributional semantics are rarely considered in general in deep learning, because data is usually transformed numerous times in the model. To maintain such semantics, we need a mechanism to ensure that the data distribution remains as undistorted as possible, especially in the dimensions of the semantics of interest, during transformations mapping between two domains.

In this paper, we propose a novel deep learning method called *distributional semantics mapping* (DSM). Given two image datasets $\mathbf{A} = \{x_i\}$ and $\mathbf{B} = \{y_j\}$, and a desire 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: Härkönen *et al.* [10] have already demonstrated that PCA applied in feature space can produce interpretable controls for image synthesis. Our approach is agnostic to specific network architectures and consists of three key modules. The first is a semantics-preserving transformation $\mathbf{e}$ where the variations of xs in a direction (*e.g.* the first principal component of $\mathbf{A}$) must be consistent with the variations of latent vectors $z_x = \mathbf{e}(x)$ in 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 by $y_{\text{recon}} = \mathbf{g}(\mathbf{e_B}(y))$ and translates x by $y_{\text{trans}} = \mathbf{g}(\mathbf{e_A}(x))$.

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

2 Related work

2.1 Generative adversarial networks

Generative adversarial networks (GANs) [9] have achieved great success for a fast-growing number of computer vision tasks, including image generation [2, 16], image colorization [15], image inpainting [13], and image super-resolution [22]. Conditional GANs [32] can be used to perform image-to-image translation [5, 18, 37, 40]. Recently, some interactive systems have been proposed using GANs for real-time portrait image editing [4, 23]. Our work also utilizes GANs conditioned on an input image, but it does not rely on any specific GAN model. To validate its generality, we employ two widely used GAN models: LSGAN [30] and NSGAN [9].

2.2 Image-to-image translation

To perform image-to-image translation, early methods such as pix2pix [15, 36] often required the networks to be trained with paired training data. Recently, a variety of approaches [18, 37] has been proposed to learn the image translation from unpaired data. For example, CycleGAN [40] leverages cycle consistency to constrain the mapping. Lu *et al.* [28, 29] show that optimal transport costs can improve the generative network. UNIT [26] assumes that 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 [12] and DRIT [24] can synthesize diverse outputs from an input image. Mejjati *et al.* [31] and Kim *et al.* [17] improve translation results using an attention mechanism. Choi *et al.* [6] propose StarGAN which can perform image translation for multiple domains using a single GAN. More recently, DRIT++ [25] extends DRIT to support multiple domains, while StarGAN v2 [7] extends StarGAN to generate diverse images across multiple domains. FUNIT [27] can work on previously unseen target classes given only 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.* [8] employ other geometric constraints (*e.g.* orientation). 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

3.1 Approach

Given two image datasets $\mathbf{A} = \{x_i\}$ and $\mathbf{B} = \{y_j\}$, we aim to compute a mapping M: $\mathbf{A} \to \mathbf{B}$, so that the distributional semantics of $\mathbf{A}$ are aligned with those of $\mathbf{B}$; we use principal components (PCs) to describe the semantics. First, we project $\mathbf{A}$ using an encoder $\mathbf{e_A}$ to $\mathbf{z_A} = \{z_{x_i}\}$ where $z_{x_i} = \mathbf{e_A}(x_i)$ is a latent vector. We keep the distributional semantics of $\mathbf{A}$ and $\mathbf{z_A}$ aligned during the encoding process. A similar projection is applied to $\mathbf{B}$ using an encoder $\mathbf{e_B}$ to get $\mathbf{z_B} = \{z_{y_i}\}$. We denote the PCs of the two latent distributions as $V_A \in \mathbb{R}^{P\times P}$ and $V_B \in \mathbb{R}^{P\times P}$ where P is the dimensionality of the latent vectors. Next we ensure the top k PCs of $\mathbf{z_A}$ are aligned with those of $\mathbf{z_B}$. Finally, we use a generative network $\mathbf{g}$ to reconstruct $\mathbf{B}$ using $y_{\text{recon}} = \mathbf{g}(\mathbf{e_B}(y))$ and translate $\mathbf{A}$ by $y_{\text{trans}} = \mathbf{g}(\mathbf{e_A}(x))$. The model is shown in Fig. 1. Below, we give the details of

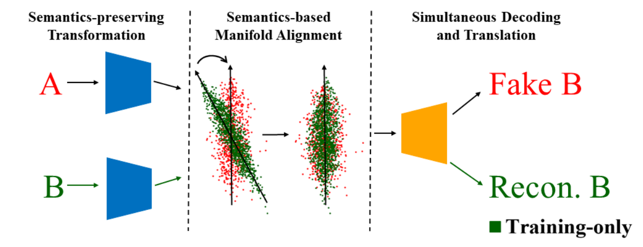
Semantics-preserving Transformation
Semantics-based Manifold Alignment
Simultaneous Decoding and Translation
A
B
Fake B
Recon. B
■ Training-only

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

the components.

3.2 Semantics-preserving Transformation

The images need to go through a sequence of transformations during translation, which in deep learning are usually some encoding processes such as convolutions. However, the shape of the latent distribution seems to be rarely considered in terms of its semantic consistency with the data distribution itself. Existing efforts such as imposing a prior distribution [20] or geometric constraints [3] are mostly aimed at encouraging the latent distribution to behave well, rather than to match the data distribution. As a result, current encoders may not be able to preserve the distributional semantics, as we confirm in experiments. We, therefore, introduce a new general autoencoding scheme to preserve the distributional semantics:

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

where x is a data sample, and $\mathbf{e}$ and $\mathbf{d}$ are 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 $\sum ||x - x'||_2^2$. Eq. (1) states the key difference between our autoencoder and existing autoencoders: it requires the projections of x on the data-space PCs U to be 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 Eq. (1), when $V = U$ and $\mathbf{e}(\mathbf{A}) = V\mathbf{A}$. Eq. (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. Eq. (1) only requires the covariance structure to persist when encoding along the first K PCs in both the data and latent space. One key question is why we do 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.

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

$$\begin{aligned} L_{\text{localAlign}} &= \frac{1}{N}\sum_{i=0}^{N-2}(L_{\text{angle}} + L_{\text{norm}}) \\ L_{\text{angle}} &= \left|\cos(x_i, x_{i+1}) - \cos(z_{x_i}, z_{x_{i+1}})\right| \quad (2) \\ L_{\text{norm}} &= \frac{||x_i||_2}{||x_{i+1}||_2} - \frac{||z_{x_i}||_2}{||z_{x_{i+1}}||_2} \end{aligned}$$

where cos() is the cosine distance and N is the batch size. L_{angle} and L_{norm} are computed on vectorized data samples and corresponding latent vectors. The goal of $L_{\text{localAlign}}$ is that, given any two data samples, their length ratio and angle should remain the same after projection into the latent space. L_{angle} aims to keep the overall shape of the distribution when the data is projected into the latent space, while L_{norm} allows scaling but prefers uniform scaling. $L_{\text{localAlign}}$ has the effect of preserving the covariance structure of the data during transformation. While we only apply local alignment constraints between a certain number of pairs in each batch, we randomly sample various batches in each training epoch, constraining further distinct pairs in the process. Furthermore, the locality of this approach, only considering two samples a time, essentially ensures that the covariance structure is invariant under homogeneous transformations of the basis in the latent space, which allows the autoencoder to choose the optimal V that aids reconstruction while preserving the covariance structure.

3.3 Semantics-based Manifold Alignment

Given $\mathbf{z_A}$ and $\mathbf{z_B}$, we have two latent distributions with their respective covariance structures characterized by their latent space PCs $V_A \in \mathbb{R}^{P\times P}$ and $V_B \in \mathbb{R}^{P\times P}$. To ensure that the UIT maintains the distributional semantics, we need to align $\mathbf{z_A}$ and $\mathbf{z_B}$, *e.g.* by aligning the direction in which $\mathbf{z_A}$ shows the biggest variation with that of $\mathbf{z_B}$ by aligning their first PCs. Further, to maintain visual semantics in multiple dimensions, we should align the top K PCs in V_A and V_B. Several alternative methods are possible, such as aligning V_A and V_B directly, or fixing one and aligning the other to it. Since both V_A and V_B are unknown, directly aligning V_A and V_B corresponds to minimizing

$$L_{\text{latentA}} + L_{\text{latentB}} + L_{\text{align}},$$

where

$$L_{\text{latentA}} = \frac{1}{N_A}\sum_{i=0}^{N_A-1}||z_{x_i} - V_A V_A^T z_{x_i}||_2^2$$
$$L_{\text{latentB}} = \frac{1}{N_B}\sum_{j=0}^{N_B-1}||z_{y_j} - V_B V_B^T z_{y_j}||_2^2 \quad (3)$$
$$L_{\text{align}} = \frac{1}{K}\sum_{k=0}^{K-1}||V_A^k - V_B^k||_2^2$$

N_A and N_B are the numbers 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. Experimentally, we have found that allowing both V_A and V_B to change causes the optimization to settle into local minima, so we fix V_A and align V_B with it. This also means that $\mathbf{e_A}$ can be pre-trained and we can compute V_A from $\mathbf{z_A}$ using PCA.

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 of B. This is a similar difficulty to the one in Sec. 3.2. Again, we operate only on a batch of N samples to ensure the global alignment of V_A and V_B:

$$L_{\text{maniAlign}} = \quad (4)$$
$$\frac{1}{N}\sum_{j=0}^{N-1}||z_{y_j} - V_{A[0:K-1]}V_{A[0:K-1]}^T z_{y_j}||_2^2 + ke^{-\alpha||z_{y_j}||_2^2}$$

where $z_{y_j} \in \mathbf{z_B}$ and $V_{A[0:K-1]} \in \mathbb{R}^{P\times K}$ contains the first K PCs of interest of $\mathbf{z_A}$. $L_{\text{maniAlign}}$ 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. The term $k\exp(-\alpha||z_{y_i}||_2^2)$ prevents z_{y_i} from shrinking, leading to a trivial solution of Eq. (4). Since the covariance structures of $\mathbf{A}$ and $\mathbf{B}$ are kept in $\mathbf{z_A}$ and $\mathbf{z_B}$ via Eq. (2), $L_{\text{maniAlign}}$ completes the semantics-based alignment of two domains.

3.4 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 using a single network $\mathbf{g}$ which serves both as a decoder and a translator: the general shapes of the two latent distributions are similar after alignment. When the decoder has been trained with the reconstruction loss on $\mathbf{z_B}$, it has already to some extent learnt to translate $\mathbf{z_A}$. We reconstruct $\mathbf{B}$ using $y_{\text{recon}} = \mathbf{g}(z_y)$. Meanwhile, we treat $\mathbf{g}$ as a generator in a generative adversarial network using $y_{\text{trans}} = \mathbf{g}(z_x)$ and use a discriminator network $\mathbf{h}(y_{\text{trans}}, y) = [0, 1]$ to further improve the translation.

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

$$L = \omega_1\frac{1}{N_B}\sum_{i=1}^{N_B-1}||y_i - y_{\text{recon}}||_2^2 + \omega_2 L_g + \omega_3 L_d + \omega_4 L_{\text{localAlign}}^B + \left(1 - \sum_{i=1}^{4}\omega_i\right)L_{\text{maniAlign}} \quad (5)$$

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 the ω_i are weights. In $L_{\text{localAlign}}^B$, we apply Eq. (2) to both the y-to-z_y, and z_x-to-y_{trans} mappings. Further details are in the appendix.

4 Implementation Details

We pre-train an autoencoder for dataset A with $L_{\text{localAlign}}$ to get $\mathbf{e_A}$ and calculate V_A from $\mathbf{z_A}$ using PCA. 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 according to the experiment. See the appendix for details. $\mathbf{e_A}$ is trained for 100 epochs and the remainder are trained for 300 epochs on all datasets, using Adam [19] 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 experiments were conducted using 2 NVIDIA GTX 1080 Ti GPUs, and PyTorch. Training took 6–18 hours.

5 Experiments

5.1 Data

We employed several benchmark datasets to validate our method, including SummerWinter [40], CatDog [24] and ShoeHandbag [38, 39]. We also built a very challenging dataset, MMISTHandbag, using hand-drawn digits from MNIST [21] and handbags randomly sampled from [39]; it has two distinctive distributions and distributional semantics. All comparative results in this section were computed using LSGAN. See the appendix for details and further results.

5.2 Evaluation Metrics

In addition to visual evaluation, we employ the Frechet inception distance (FID) [11] 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 the *ordering-tolerance curve* (OTC). Given images $x \in \mathbf{A}$ and their translations $y_{\text{trans}} \in \mathbf{B}$, we define the OTC as:

$$c = \frac{1}{N_A}\mathbf{card}(\{x|d(x, y_{\text{trans}})/N_A \le \beta,\ x \in \mathbf{A}\}), \quad (6)$$

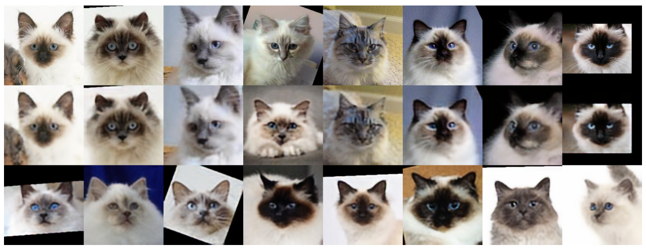
Fig. 2 Top: original data. Mid: latent samples of DSM. Bottom: latent samples of a standard autoencoder. Left to right: images ranked 4th, 19th, 24th, 37th, 44th, 52nd, 94th and 97th on the first PCs of the data (top) and latent space (center, bottom).

where $\beta \in [0,1]$, $\mathbf{card}(\cdot)$ is the cardinality of a set, $d(x, y_{\text{trans}}) = |\text{rank}(y_{\text{trans}}) - \text{rank}(x)|$, $\text{rank}(x)$ 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 $\mathbf{A}$, and $\text{rank}(y_{\text{trans}})$ is the rank of y_{trans} along a chosen PC in $\mathbf{B}$. $d(x, y_{\text{trans}})$ is equal to zero if the rank of x is retained during translation, and non-zero otherwise (the larger the worse). c is the *percentage of correctly ordered* x values 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 distributional semantics. To show the necessity of such transformations in UIT, we trained an autoencoder on cat images in CatDog then computed the 1st PCs of A and $\mathbf{z_A}$. The autoencoder is based on the encoder and decoder of UNIT. Please see the appendix for further details. We then ranked all images along the two PCs: see Fig. 2.

In the original data, the variation on the 1st PC is mainly a color transition from light to dark (see Fig. 2(top)). However, without semantics-preserving transformations, the shape of the latent distribution is changed, and unable to preserve the visual semantics (Fig. 2(bottom)). In contrast, DSM preserves the distributional semantics (Fig. 2(center)). We also show the OTCs in Fig. 3(left): DSM can contain the rank error to under 3% while a standard autoencoder fails systematically. Although we only show the results from a specific autoencoder, our preliminary experiments showed this to be a common problem of autoencoders.

5.4 Architectural Studies

Two main adjustable components of DSM are the GAN architecture and the number of PCs, K. We first evaluated DSM on the first K PCs on all datasets; we present the FID scores in Tab. 1. While we expected

Tab. 1 FID scores on four datasets with the first K PCs aligned. When $K = 0$, DSM is trained without alignment loss.

Dataset	$K=0$	$K=1$	$K=2$	$K=3$
CatDog	64.5	**58.0**	96.2	84.5
SummerWinter	100.9	**90.4**	107.6	99.5
ShoeHandbag	**123.6**	129.7	155.5	128.9
MNISTHandbag	172.7	180.6	**149.3**	171.8

that a larger K would result in a harder optimization problem and hence lower quality, the results show that it depends on the dataset. After considering the data, this is understandable because different datasets have different variance distributions over the PCs: some have large variance on the 1st PC while others have variances spread over the first K PCs, which affects the behavior of Eq. (4). Although control can be added, *e.g.* by adding weights to different PCs, we choose not to do so and let DSM adapt to the data. We only tested DSM up to $K = 3$: while DSM can work for $K \geq 3$, it is typically hard for humans to visually understand the semantics in PCs where $K \geq 3$. We also compared two GAN architectures LSGAN and NSGAN on ShoeHandbags and show the OTCs in Fig. 3(center). Both keep the rank error to within around 20%, showing that alignment constraints can improve translation ability visibly for differing GAN architectures. Further 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 value of K depends on the dataset. The first PC is almost always visually interpretable across a dataset, and some datasets have meaningful variations on other PCs. We show the images from the four datasets ranked along their respective meaningful PCs in Fig. 4. For CatDog, SummerWinter and ShoeHandbags, the first PC (PC0) shows color variations from light to dark, for MNIST, shape varies from slim to round, while for ShoeHandbags, the second PC (PC1) shows shoe collar height variation for shoes, and handle length variation for handbags.

In Fig. 5, we show our results for mappings of cat-to-dog, summer-to-winter, shoes-to-handbags, and digits-to-handbags. PC0s of CatDog and SummerWinter are color variations (light to dark). The major difference is that 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

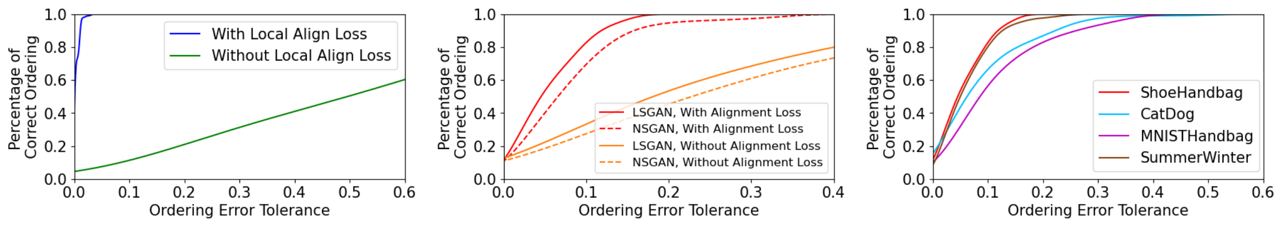

Fig. 3 OTCs in various experiments. Left: OTC along PC0 with/without semantics-preserving transformation on CatDog. Center: LSGAN vs NSGAN along PC0 on ShoeHandbags. Right: OTC along PC0 for four datasets.

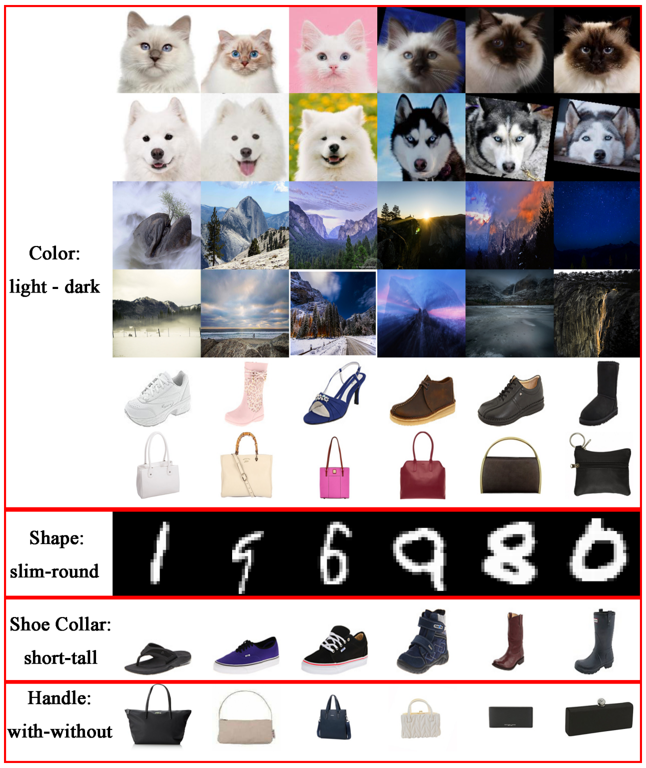

Fig. 4 Images ranked according to PCs. Top to bottom: CatDog (PC0), SummerWinter (PC0), ShoeHandbags (PC0), MNIST (PC0), and ShoeHandbags (PC1).

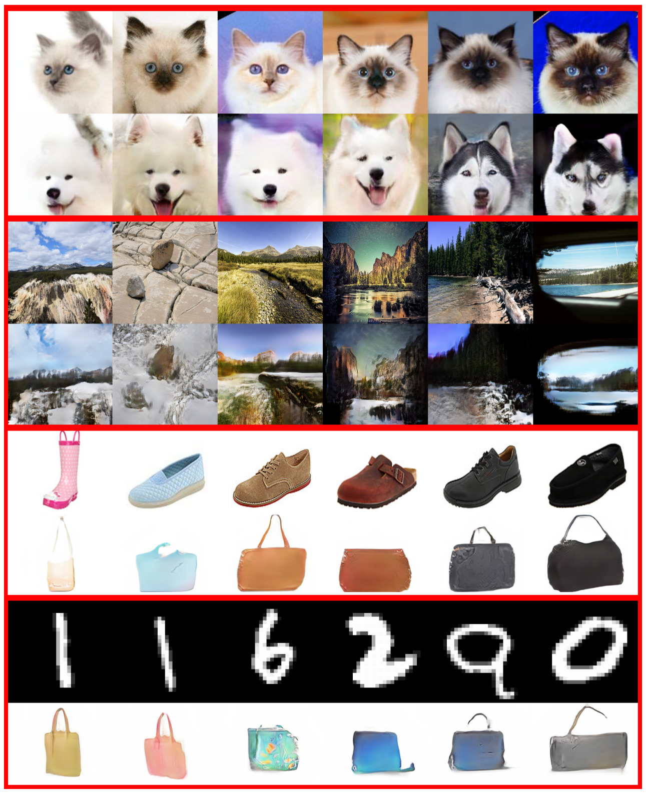

Fig. 5 Mapping results. Top to bottom: CatDog, SummerWinter, ShoeHandbag and MNISTHandbag, ordered along PC0. In each pair of rows, the upper row shows the input images and the lower row shows the translated images.

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; DSM lays snow on different landscapes. In shoes-to-handbags, unlike CatDog and SummerWinter, the color variation is restricted to the object itself, and DSM faithfully keeps the semantics. Finally, to push DSM further, we tested a digit-to-handbag 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 their respective PC0s in Fig. 4. We also show the OTCs in Fig. 3(right).

5.6 Comparisons

We compare our model to UNIT, CycleGAN, DistanceGAN and DRIT++ on the CatDog and ShoeHandbag datasets, by using the public code shared by the authors of these methods. We first give FID scores for all methods in Tab. 2 and OTCs in Fig. 6. By aligning two data manifolds based on their semantics, DSM is able to improve the translation quality for both datasets. The OTCs show clearly that DSM can keep the semantics on PC0 better

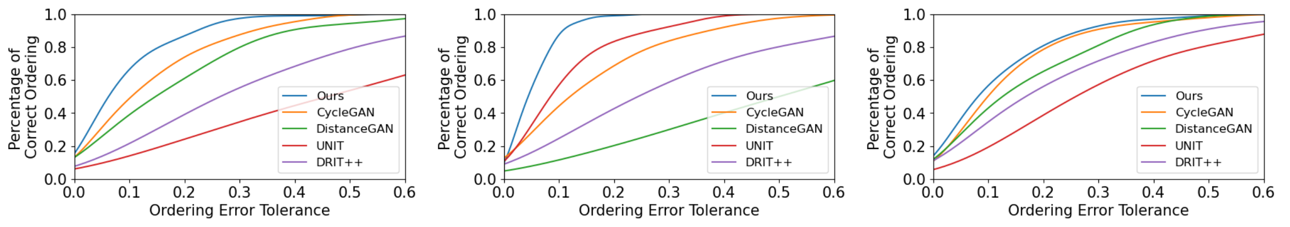

Fig. 6 OTC scores for various methods. Left: CatDog. Center: Shoe-to-Handbags on PC0 of Handbags. Right: Shoe-to-Handbags on PC1 of Handbags.

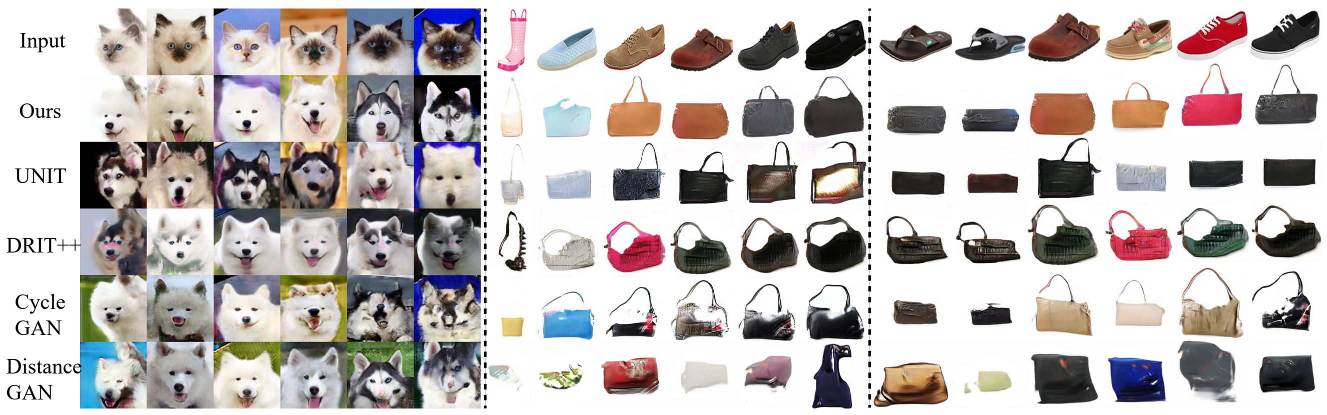

Fig. 7 Left, center: ordering along PC0 of Dogs, Handbags. Right: ordering along PC1 of Handbags.

Tab. 2 FID scores on CatDog and ShoeHandbag datasets for various methods.

Method	CatDog	ShoeHandbag
Ours	**58.00**	**128.93**
CycleGAN	95.12	135.55
DistanceGAN	63.60	185.28
DRIT++	61.95	185.01
UNIT	64.62	140.88

than the other methods by containing the rank error to within around 30% and 18% (see Fig. 6). The second best methods contain it to roughly only 50% and 40%. On PC1 of ShoeHandbag (see Fig. 6(right), CycleGAN is close to DSM. However, we argue that its behavior is inconsistent: see Fig. 6(left, mid) where the variance is large and contains large amounts of semantic information.

Visual comparisons can be found in Fig. 7. Overall, DSM generates images of higher visual quality. Additionally, other methods are incapable of preserving or matching the distributional semantics. In Fig. 7(left), the major variation of the input images from left-to-right is a color variation, light-to-dark. While DSM obviously keeps the same variations during UIT, it is hard to find similar effects in other methods. Similar observations can also be made in Fig. 7(centre).

To further explore semantics in other PCs, we show Fig. 7(right). While the input varies from slippers to sneakers, our results generates handbags varying from those without handles to those with handles. In contrast, other methods struggle to generate consistent semantic variation. Further results can be found in appendix.

6 Discussions and Conclusion

While PCA is a straightforward way of characterizing distributional semantics, our method can incorporate alternative techniques such as kernel PCA, multidimensional scaling, and others that can define the covariance structure by ‘flattening’ the data manifold before performing DSM.

We have only evaluated DSM up to $K = 3$ because the semantics start to lack visual meaning for $K > 3$. However, we argue that DSM is effective and useful for two reasons. Firstly, for almost all datasets, the first PC bears the majority of the variance, and the distributional semantics captured by the variance are always visually interpretable. Secondly, aligning the PCs of the two distributions during translation increases image quality.

In summary, we have proposed DSM, the first UIT method which preserves and matches the distributional semantics of two image domains. It is straightforward and effective, as demonstrated on multiple datasets, and capable of improving translation quality compared to the state-of-the-art. DSM is also general in its capacity to incorporate any GAN and autoencoder model. In future, we will incorporate human-labelling in 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 up to many other applications.

Acknowledgements

We thank the anonymous reviewers for their valuable comments. The work was supported by NSF China (Grant 61772462), and the 100 Talents Program of Zhejiang University.

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

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.

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

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

[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.

[5] S.-Y. Chen, W. Su, L. Gao, S. Xia, and H. Fu. Deepfacedrawing: Deep generation of face images from sketches. *ACM Transactions on Graphics (TOG)*, 39(4):72–1, 2020.

[6] 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.

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

[8] 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.

[9] 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.

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

[11] 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.

[12] 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.

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

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

[15] 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.

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

[17] 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.

[18] 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.

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

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

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

[22] 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.

[23] 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.

[24] 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.

[25] 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.

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

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

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

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

[34] 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.

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

[36] 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.

[37] 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.

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

[39] 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.

[40] 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.

Appendix

A Implementation Details

A.1 Loss functions

Eq. (5) contains a local alignment loss $L^B_{\text{localAlign}}$, which applies Eq. (2) to both y-to-z_y, and z_x-to-y_{trans}. Applying Eq. (2) to y-to-z_y is straightforward, and ensures semantics-preserving transformation between $\mathbf{B}$ and $\mathbf{z_B}$, just as does the encoder of $\mathbf{A}$. Here we explain why applying Eq. (2) to z_x-to-y_{trans} is essential. As Sec. 3.4 explains, we align $\mathbf{z_A}$ and $\mathbf{z_B}$, and use a single network $\mathbf{g}$ to serve as both the decoder for $\mathbf{z_B}$ and the generator for $\mathbf{z_A}$. This mechanism provides an implicit constraint on the translated images $\{y_{\text{trans}}\}$ from $\mathbf{z_A}$ so that it has a similar distribution to the reconstructed images $\{y_{\text{recon}}\}$ from $\mathbf{z_B}$. However, as there is an additional GAN loss which modifies the distribution of the translated images, the alignment of $\{y_{\text{trans}}\}$ and $\{y_{\text{recon}}\}$ may be affected, compromising the distributional semantics matching. Thus, we apply Eq. (2) to z_x-to-$\{y_{\text{trans}}\}$ to explicitly preserve the semantics:

$$\omega_4 L^B_{\text{localAlign}} = \omega^{\mathbf{e}}_4 L^{\mathbf{e}}_{\text{localAlign}} + \omega^{\mathbf{g}}_4 L^{\mathbf{g}}_{\text{localAlign}} \quad (7)$$

where $L^{\mathbf{e}}_{\text{localAlign}}$ and $L^{\mathbf{g}}_{\text{localAlign}}$ are the loss terms for y-to-z_y and z_x-to-y_{trans}. All experiments set $\omega_1 = 0.033, \omega_2 = \omega_3 = 0.333, \omega^{\mathbf{g}}_4 = 0.167$ in Eq. (5) of the paper. We set $\omega^{\mathbf{e}}_4 = 0.066$ in ShoeHandbag align 3PCs experiments and $\omega^{\mathbf{e}}_4 = 0.05$ in all other experiments. We set $k = 15, \alpha = 10^{-6}$ for the regularization term in Eq. (4).

A.2 Network Architecture

Our image translation network architecture is based on the one from UNIT [26]. 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 further convolutional layer (see Tab. 3 for details). Furthermore, instead of using instance normalization (IN) [35], we utilize batch normalization (BN) [14] in the encoder and generator. For the discriminator

Tab. 3 Network architecture for the distributional semantics mapping experiments

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	

Tab. 4 Network architecture for the autoencoder

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

network, we employ spectral normalization [33], and multi-scale discriminators at 3 scales. The network architecture is given in Tab. 3. We use the following abbreviation for ease of presentation: N: number of kernels number, K: kernel size, S: stride. UP indicates a 2 nearest-neighbor upsampling layer and RESBLK, a residual basic block. The detailed network architecture of the pre-trained autoencoder for **A** is given in Tab. 4.

B Further Experimental Results

B.1 Data Details

For all datasets, images were resized to 256 × 256. In CatDog, 871 cat (birman) and 1364 dog (husky, samoyed) images were randomly divided into 771 (cat) and 1264 (dog) for training and the remainder used for testing. SummerWinter comprises 1540 summer photos and 1200 winter photos, which were randomly divided into 1231 (summer) and 962 (winter) for training and the remainder used for testing. For ShoeHandbag, we randomly sampled images from edges2shoes and edges2handbags, using 3726 (shoe) and 3822 (handbag) for training, and 101 (shoe) and 178 (handbag) for testing. For MNISTHandbag, 1600 MNIST images and

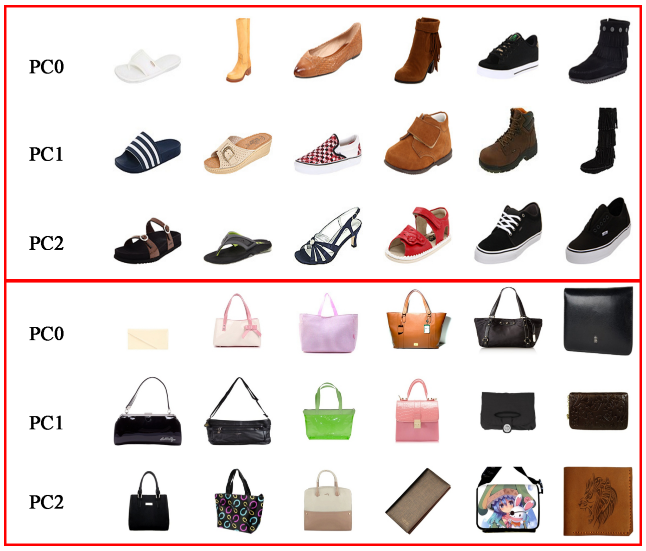

Fig. 8 Top: Shoe dataset. Bottom: Handbag dataset. From left to right: the images ordered along the corresponding PC

1600 handbag images were randomly selected from MNIST and edges2handbags, with 1500 of each for training and 100 for testing. We show images from the ShoeHandbag dataset along the first 3 PCs in Fig. 8. This dataset is very challenging, as the distributions of MNIST and handbags are very different. The top five variances ratios along PCs in

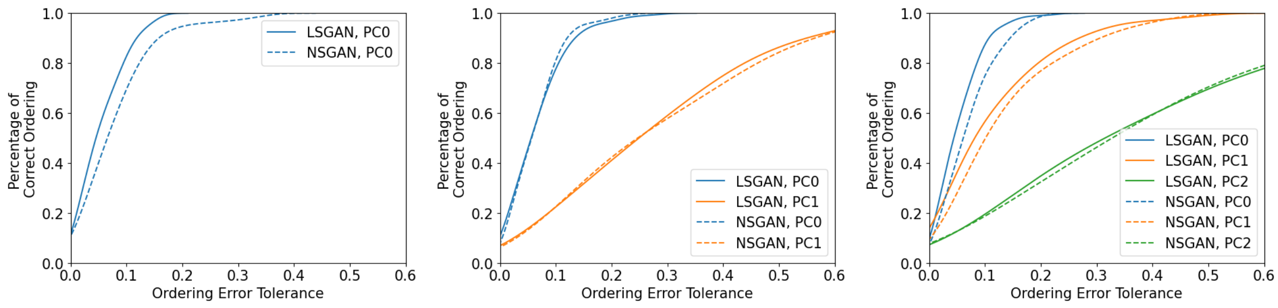

Fig. 9 LSGAN/NSGAN

MNIST are [0.095, 0.071, 0.066, 0.052, 0.047], while in handbags these ratios are [0.274, 0.115, 0.065, 0.054, 0.026].

B.2 Our OTCs

Fig. 10 shows OTCs for our method on the CatDog, MNISTHandbag and SummerWinter datasets, with up to 3 PCs aligned. We can see that our method always keeps semantics best for PC0, and the rank errors become larger on PC1 and PC2. This is mainly because the variance ratio along PC0 is always much larger than along other PCs. For example, in the Handbag dataset, the ratios of variances along the first 3 PCs are about 4:2:1. As a result, the network prefers to perform alignment along PC0 in preference to minimize the total loss. We also note that semantics preservation varies largely across different datasets. In the challenging MNISTHandbag dataset which has two distinctive distributions, our method preserves semantics well along PC0, while in the SummerWinter dataset, our method is capable of preserving semantics on the first 3 PCs. Although control could be used e.g. via weights 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 which should be respected during translation.

We evaluate semantics preservation only for the first 3 PCs for two main reasons. Firstly, 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 directions with very small variations adds complexity to the optimization while decreasing the explicability of the mapping. Secondly, by investigating various popular datasets (e.g. Shoes, Handbags, Cars, Animals, Faces, Art works), we discovered that while people can easily perceive semantics on the first PC in all datasets, they can only do so on the second PC in the Shoe, Handbags, and MNIST datasets. People cannot perceive any semantics on the 4th and subsequent PCs. Hence we focus on the first 3 PCs.

Tab. 5 FID scores on ShoeHandbag with the first K PCs aligned

	$K=1$	$K=2$	$K=3$
LSGAN	129.7	155.6	**128.9**
NSGAN	172.8	187.0	**154.9**

B.3 Comparison of LSGAN and NSGAN

Fig. 9 and Tab. 5 compare using LSGAN and NSGAN for image translation on the ShoeHandbag dataset when aligning the first 1–3 PCs. The OTCs show that the two GAN models result in very similar semantics preservation in all cases, demonstrating that our method does not rely on a specific GAN model and can preserve semantics using different GAN models. We also note that the FIDs of NSGAN are higher than that for LSGAN. This is mainly because the image generation capability of NSGAN is weaker than that of LSGAN. Employing other GAN models such as StyleGAN [16] can improve the FID scores.

B.4 Alignment of PCs

Eq. (4) in the paper requires the first K PCs of two domains 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 should be aligned with PC1 of the second domain, etc. We have two choices. The first is to enforce order, PC0-

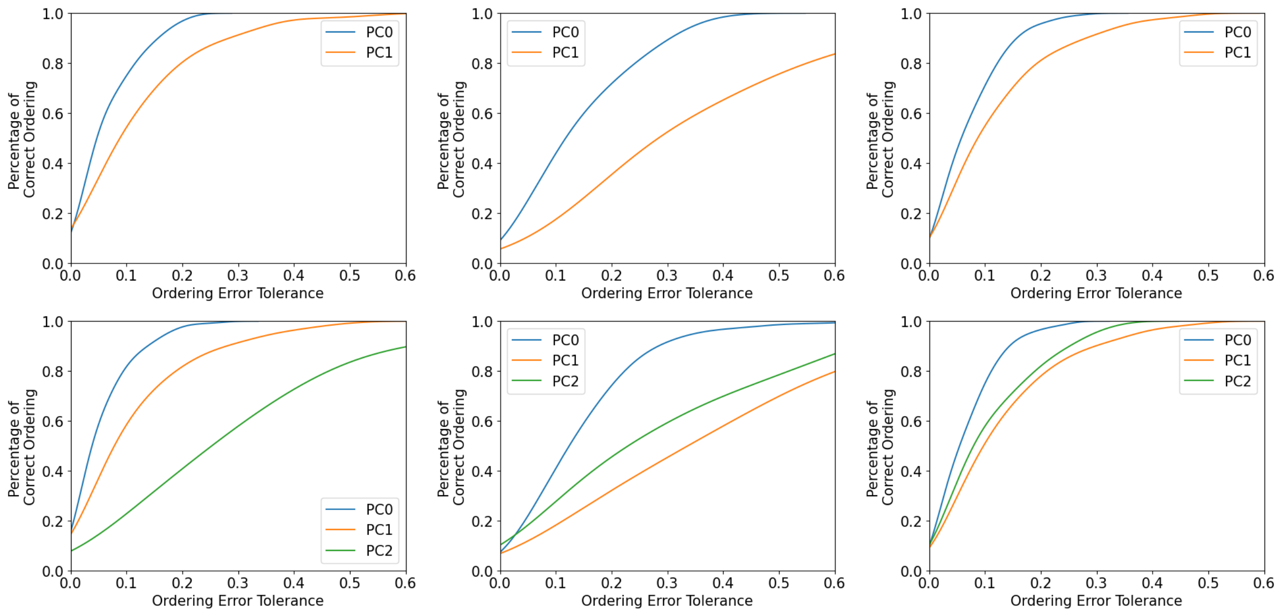

Fig. 10 Above: Aligning PC0 and PC1. Below: Aligning PC0-2. Left to Right: CatDog, MNISTHandbag and SummerWinter.

to-PC0, PC1-to-PC1, and so on. However, we find this to be sub-optimal in the sense that Eq. (4) is affected by distribution of variances across PCs. For a dataset with a majority of variance on PC0, the alignment forces due to Eq. (4) have little effect on PC1 and subsequent PCs. We argue that they should be small because the dataset's variance on PC1 and higher PCs is less explicable. Hence, the force provided by Eq. (4) 4 should naturally follow variance distribution. We therefore do not enforce order of alignment of PCs for two datasets. As a result, while PC0 is always mapped to PC0 in all experiments, sometimes PC1 of one dataset can be mapped to PC2 of another. We argue that such a mapping is still valid because it is explicable and reflects the distributional semantics.

Zhexi Peng received his B.S. degree in information and computing sciences from Beijing University of Posts and Telecommunications. He is currently a Ph.D. student at Zhejiang University. His research interests cover computer vision, SLAM and machine learning.

He Wang is an Associate Professor at the School of Computing, University of Leeds, UK. He is the Director of High-Performance Graphics and Game Engineering and the Academic Lead of the Centre for Immersive Technology. His current research interests are mainly in computer graphics, vision, and machine learning and applications.

Yanlin Weng is an associate professor in the School of Computer Science and Technology at Zhejiang University. She got her Ph.D degree in Computer Science from the University of Wisconsin-Milwaukee, and her master's and bachelor's degrees in Control Science and Engineering from Zhejiang University. Her research interests include computer graphics and multimedia.

Yin Yang is an associate professor in the School of Computing, Clemson University. Previously, he was a faculty member of the University of New Mexico. He received his Ph.D. from the University of Texas at Dallas (with a David Daniel fellowship). He is a recipient of a NSF CRII award (2015) and a CAREER award (2019). His research aims to develop efficient and customized computing methods for challenging problems in graphics, simulation, machine learning, vision, visualization, robotics, medicine, and many other applied areas.

Tianjia Shao is a ZJU100 Young Professor in the State Key Laboratory of CAD&CG, Zhejiang University. Previously He was a Lecturer in the School of Computing, University of Leeds. He received his Ph.D. in Computer Science from the Institute for Advanced Study, Tsinghua University, and his B.S. from the Department of Automation, Tsinghua University. During his Ph.D., he spent one year as a visiting researcher in University College London and four years as a research intern in the Internet Graphics Group, Microsoft Research Asia before becoming an Assistant Researcher in the State Key Laboratory of CAD&CG, Zhejiang University.